Interface Homme-Machine
et ergonomie
EG23

Projet
Second rapport

Introduction

Le projet Crisegest prend place dans le cadre du cours d'Interface Homme-Machine. Il s'agit donc ici de se concentrer sur ce qui fait qu'une interface va être agréable à utiliser par les futurs utilisateurs du logiciel. Ainsi, ce projet implique de choisir différents types d'éléments visuels en fonction de différentes utilisations. Le but étant d'arriver à une interface claire, compréhensible du premier coup d'œil, instinctive, pratique et surtout rapide à utiliser. En effet, si l'accent est mis sur la rapidité d'utilisation, c'est bien parce que ce projet prend place dans un contexte de crise potentielle, et nécessite donc une absolue rapidité d'utilisation par des membres du milieu médical.

Ce second rapport se concentrera bien davantage sur les choix de conception, les choix de réalisation et le résultat final ; là où le premier concernait plutôt la présentation visuelle de l'application et les choix pris pour répondre au mieux aux différentes demandes du client. Le second rapport se situe alors dans la continuité du premier rapport. Il en concerne l'après-réalisation et la prise en compte des possibilités et des limites de Delphi, alors que le premier rapport était l'avant-réalisation et une première vision de l'application, forcément un peu utopique puisque ne prenant pas en compte les possibilités et les limites de Delphi.

Durant ce rapport, nous attacherons alors à expliquer les différentes parties et choix réalisés au niveau de la conception, via les diagrammes du Schéma Navigationnel d'Interactions et du Schéma d'Enchaînement des Fenêtres. Ce n'est qu'ensuite que nous aborderons les principales difficultés rencontrées –sous une forme brève–, puis les décisions prises pour les résoudre –sous une forme déjà moins brève–. Enfin viendra une présentation du résultat final et de ce qui pourrait encore être amélioré.

I) La conception : deux diagrammes, le SNI et le SEF

À faire. Partie Julien.

II) La réalisation : difficultés rencontrées

Les difficultés rencontrées et problèmes à résoudre furent nombreux lors de l'implémentation en Delphi. En effet, il nous a fallu réfléchir à la mise en place de plusieurs aspects, principalement au niveau de la programmation mais pas uniquement. Elles concernent chacun des différents TPanel/TPageControl et peuvent se résumer sous la forme d'une question par difficulté :

- **TPanel « Carte » :**
  - Comment réaliser la carte ?
  - Comment réaliser la structure « drag & drop » des objets plaçables sur la carte ?
  - Comment afficher la météo d'un lieu visible sur la carte ?

- **TPanel « Filtres » :**
  - Comment filtrer les différents types d'objets plaçables sur la carte en fonction de nos préférences ?
  - Comment respecter les conventions des TCheckBox ? (Par exemple, cocher une TCheckbox « générale » comme « Véhicules » fait cocher automatiquement toutes les TCheckBox « particulières » du même groupe comme « Police » et « SAMU ».)

- **TPageControl « Informations générales et spécifiques » :**
  - Comment représenter les deux types d'informations spécifiques ?
  - Comment modifier les informations, qu'elles soient générales ou spécifiques ?

- **TPanel « Tâches à réaliser » :**
  - Comment ajouter des tâches sur l'interface sans risquer de déborder du cadre visuel de l'objet ?

- **TPanel « Chat » :**
  - Comment réaliser un chat ?

La plupart de ces difficultés rencontrées ont abouti à plusieurs solutions possibles. Il a alors fallu effectuer un choix, une décision, envers la solution qui constituera une partie de notre interface future. Cela nous amène à aborder ces différentes possibilités, et ce que nous avons finalement décidé de conserver.

III) La réalisation : plusieurs possibilités, une seule décision

À plusieurs reprises, le projet nous a amené à devoir prendre des décisions lors de sa réalisation. Entre autres face aux problèmes rencontrés lors de l'implémentation en Delphi et décrits précédemment. De nombreux choix ont alors été réalisés pour aboutir à l'interface telle qu'elle l'est aujourd'hui. Certains peuvent sembler ne concerner que des détails, mais c'est justement la prise en compte des détails qui distingue une interface claire et intuitive d'une interface peu ergonomique. Réfléchir à la solution la plus adaptée dans les différentes situations auquel ce projet nous a confronté permet d'aboutir à un résultat qui sera plaisant à utiliser pour l'utilisateur.

A) Choix généraux

Nous nous sommes demandés si l'ouverture d'une nouvelle fenêtre devait pouvoir permettre à l'utilisateur de rebasculer sur la fenêtre principale avant la fermeture de la fenêtre annexe, ou si l'ouverture de la fenêtre annexe verrouillerait, jusqu'à sa fermeture, la possibilité de revenir à la fenêtre principale. Pour plus de praticité d'utilisation, nous avons préféré opter pour la première solution. En effet, rien ne nous l'empêche dans notre système. Par exemple, aucun accès particulier à une base de données ne nécessite ce verrouillage.

De plus, toutes nos fenêtres ne sont pas redimensionnables. Nous avons en effet désactivé le carré d'agrandissement, sur le bord de la barre en haut de la fenêtre, ainsi que le redimensionnement via les bords de la fenêtre. Cela permet de ne pas perdre la disposition originale de l'interface, celle-ci n'étant pas adaptable selon les dimensions de la fenêtre. De cette manière, ni un agrandissement qui ajouterait des emplacements vides, ni un rapetissement qui cacherait certains champs, n'est possible.

L'aide est accessible aussi bien via le TMainMenu de la fenêtre principale qu'un TButton présent sur chacune des fenêtres annexes. Cela permet à l'utilisateur de ne jamais être perdu et d'avoir toujours une aide sous la main si jamais il ne trouve pas une information dans le logiciel ou n'arrive pas à utiliser ce dernier. Nous avons donc préféré en intégrer une et la rendre disponible quel que soit notre localisation dans le logiciel.

L'ordre des tabulations a été mis de haut en bas pour les TPanel, puis en fonction de la solution la plus pratique et la plus courante selon les types d'objets à l'intérieur de chacun des Tpanel. Nous avons préféré respecter les conventions à ce niveau et être le plus logique possible, pour ne pas rendre le logiciel perturbant à utiliser.

Tous les champs de l'interface, comme la validation de la ville saisie pour la carte ou l'envoi d'un message dans le chat, peuvent être validés grâce à une pression de la touche « Entrée » du clavier. Effectivement, cela permet plus de rapidité d'utilisation que le clic d'une souris.

B) Choix du TPanel « Carte »

La première difficulté à résoudre et choix à prendre était bien sûr l'affichage de la carte. Cela pouvait se faire par plusieurs moyens. Après plusieurs recherches, nous avons décidé d'utiliser l'API Google Map, pour sa facilité de mise en œuvre et sa praticité. En effet, elle nous permettait de faire tout ce que nous voulions à travers des fonctions déjà programmées et une base de données complète. Ainsi, ce choix nous aura beaucoup facilité l'ajout d'options qui peuvent se révéler utiles en crises, tels que la couche météorologique, la couche du trafic ou l'affichage particulier de la carte permis par la Street View.

Nous nous sommes également posés la question de la pertinence de l'utilisation d'onglets pour l'affichage de la carte sous une forme normale, et ce même-affichage sous une forme indiquant la météorologie des différentes régions. Nous avons finalement décidé d'opter pour une interface plus pratique à utiliser, plus cohérente et plus pertinente, sous la forme d'une TCheckBox qui active ou désactive l'affiche de la météorologie des différentes régions. En effet, cet affichage n'est finalement qu'une couche supplémentaire, au même titre que l'affichage du trafic. La météorologie comprend l'affichage d'une couche indiquant le temps et la température de différentes régions en fonction du zoom ; ainsi que l'affichage des nuages si la carte est suffisamment dézoomée par rapport au zoom par défaut.

Des TSpeedButton ont été utilisés pour permettre l'affichage de personnes ou d'objets particuliers sur la carte. Cette forme nous paraissait la plus appropriée pour représenter ces différents types d'objets. En effet, les TSpeedButton, petits mais lisibles par leurs icônes, permettaient d'en utiliser une quantité assez importante –ici, dix–, sans qu'ils ne prennent trop de place. De plus, leur taille et leur alignement laissent aussi deviner leur utilisation. Enfin, il leur a été ajouté une possibilité. En effet, ces derniers affichent le mot leur correspondant si on laisse la souris les survoler pendant quelques dixièmes de seconde, grâce à leur propriété « Hint ». Cela permet à l'utilisateur de comprendre malgré tout à quoi se rapporte le TSpeedButton s'il ne l'avais compris avant, soit de part le système des TSpeedButton (incompréhension globale), soit de part une image de TSpeedButton qu'il ne trouverait pas assez explicite (incompréhension particulière/précise).

Le TMemo servant à la saisie de la ville pour la carte comporte, par défaut, un passage à la ligne obligatoire si l'utilisateur appuie sur la touche « Entrée ». Ce passage à la la ligne a été désactivé pour correspondre aux conventions, donc ce qui est habituellement utilisé, et ne pas perturber l'utilisateur.

Ce n'est pas un hasard si les TCheckBox sont regroupées dans un coin du TPanel « Carte » et l'une au-dessus des deux autres. Nous avons préféré cette utilisation de l'espace du TPanel à celle qui consistait à mettre les trois TCheckBox sur une même ligne visuelle horizontale. En effet, l'interface doit être rapide à utiliser, et si nous avions aligné les trois TCheckBox, l'utilisateur se retrouvait dans l'obligation de bouger conséquemment la souris. L'interface actuelle permet au contraire à l'utilisateur de ne pas avoir à déplacer beaucoup la souris pour cocher l'une ou l'autre des trois TCheckBox, ce qui rend l'interface plus rapide à utiliser.

C) Choix du TPanel « Filtres »

Il a été choisi que la sélection d'une TCheckBox « générale » comme « Véhicules » provoquerait la sélection de toutes les TCheckBox « particulières » du même groupe, c'est-à-dire « Pompier », « SAMU », « Police » et « Accidenté » dans notre exemple. À l'inverse, la désélection de la TCheckBox « générale » provoque la désélection de toutes les TCheckBox « particulières ». Cette possibilité respecte les conventions utilisés en termes de cases à cocher. Nous avons décidé de respecter ces conventions, déjà parce qu'étant souvent utilisées, un utilisateur saura s'y retrouver et ne sera pas perturbé par de nouvelles règles ; aussi parce que nous trouvions ces conventions logiques et intuitives, et que nous ne voyons pas de raison d'en changer.

D) Choix du TPageControl « Informations générales et spécifiques »

Les informations inscrites dans l'espace réservé aux informations générales et spécifiques sont affichées via un TLabel, puis peuvent être modifiées grâce à un TButton placé juste à la droite de l'information qui transformera alors le TLabel en TEdit (ou du moins, en donnera l'illusion visuellement). La validation du contenu du TEdit retransformera alors le TEdit en TLabel. Ce choix d'interface a été préféré à une solution où le TButton de modification ouvrait directement une nouvelle fenêtre, avec une fenêtre différente par information. Cette solution, en plus d'être laborieuse à mettre en place, manquait à la fois de praticité et de rapidité d'utilisation. Nous avions également pensé à mettre des TEdit dont la modification aurait été permise ou non selon le fait d'avoir cliqué sur le TButton de modification ou non. Cette solution avait le défaut de ne pas rendre instantanément visible le fait qu'il soit permis de modifier le champs ou non. Seul l'inscription sur le TButton de faire la distinction.

Dans l'interface finale, on peut également choisir de modifier toutes les informations d'un coup grâce à un TButton, ce qui ouvrira une nouvelle fenêtre. À l'inverse du choix précédent, avec une grosse quantité d'informations à saisir, il nous a semblé préférable de créer une nouvelle fenêtre. En effet, le nombre de clics est alors drastiquement réduit, étant donné qu'en utilisant la solution, il aurait fallu cliquer sur tous les TButton de modification puis tous ceux de validation, au lieu de l'unique TButton de modification et de l'unique TButton de validation tels qu'ils le sont dans l'interface actuelle.

Également, tant que les informations générales n'ont pas été modifiées une première fois via la fenêtre annexe qui permet de toutes les modifier en une fois, il est impossible de modifier une seule et unique information. Cela nous a semblé le choix le plus logique. En effet, quand l'utilisateur va découvrir l'interface, il va probablement commencer par ajouter la plupart des informations si ce ne sont toutes… Le faire au cas par cas prendra plus de temps que l'ouvrir dans une fenêtre annexe, pour des raisons évoquées dans le précédent paragraphe. D'où l'intérêt de ne laisser, dans un premier, l'accès qu'au TButton qui permet de modifier toutes les informations.

Aussi, le TTabSheet concernant les informations spécifiques n'est pas visible tant que les informations générales n'ont pas été modifiées une première fois. En effet, la fenêtre annexe permettant de modifier toutes les informations générales oblige l'utilisateur à sélectionner un type de crise. C'est alors ce type de crise qui décidera du contenu du TTabSheet concernant les informations spécifiques. Sans type de crise, il n'est pas possible d'afficher quelconque contenu au TTabSheet concernant les informations spécifiques, ce qui explique son absence avant la première sélection du type de crise.

E) Choix du TPanel « Chat »

Le contenu du chat est stocké dans un fichier, lui-même lu par un TMemo. Cela a l'avantage de permettre une persistance des données lors d'une discussion avec d'autres personnes. C'est pourquoi nous avons préféré cette solution de mise en œuvre du chat à celle qui consistait à envoyer directement les informations dans le TMemo. D'autant qu'en situation réelle (ici, le chat est simulé, nous ne pouvons pas communiquer entre deux ordinateurs séparés mais seulement sur un seul), un fichier commun peut stocker les différents envois de messages. Cette mise en œuvre s'approche donc davantage d'une mise en situation réelle que l'envoi direct dans le TMemo.

IV) Le résultat final

Le logiciel est divisé en plusieurs parties bien distinctes à travers ses quatre TPanel et son TPageControl. Chacun de ces TPanel/TPageControl a une fonction. Il est plus simple de vérifier si le logiciel tient ses promesses en parlant de chacun de ces TPanel/TPageControl un à un :

- **TPanel « Carte » :**
  - TSpeedButton déplaçables sur la carte : fonctionne.
  - Affichage de la carte : fonctionne, mais nécessite Internet, puisque l'API Google Map est utilisée.
  - Changement de lieu affiché par la carte suite à la saisie d'une adresse : fonctionne. Peut être réalisé par un clic de souris sur le TButton de validation ou en appuyant sur la touche « Entrée ».
  - Couches trafic et météorologique rajoutées sur la carte, et mode Street View disponible : fonctionne.

- **TPanel « Filtres » :**
  - Filtrage des TSpeedButton en fonction de nos sélections : fonctionne.

- **TPanel « Informations générales et spécifiques » :**
  - Transformation du JLabel en JEdit après clic sur le TButton de modification, avec réutilisation de la donnée précédemment saisie : fonctionne.
  - Transformation du JEdit en JLabel après clic sur le TButton de validation, avec affichage de la donnée validée : fonctionne.
  - Modification de toutes les données générales : fonctionne.
  - Modification de toutes les données spécifiques : fonctionne.
  - Désactivation des TButton de modification d'un unique champ et TTabSheet « Informations spécifiques » invisible tant qu'il n'y a pas eu validation des données saisies via la fenêtre s'ouvrant grâce à un clic sur le TButton « Tout modifier » : fonctionne.
  - Adaptation du TTabSheet « Informations spécifiques » en fonction du type de crise (cas 1 : accident de la route ; cas 2 : autres crises) : fonctionne.

- **TPanel « Tâches à réaliser » :**
  - Ajout d'une tâche : ne fonctionne pas, seule la fenêtre d'ajout a été réalisée.
  - Affichage des tâches sous forme d'une liste : ne fonctionne pas.
  - Suspension d'une tâche : ne fonctionne pas.

- **TPanel « Chat » :**
  - Envoi de message : fonctionne. Peut être réalisé par un clic de souris sur le TButton d'envoi ou en appuyant sur la touche « Entrée ».
  - Sauvegarde des messages dans un fichier : fonctionne.
  - Lecture des messages du fichier pour les afficher dans le chat : fonctionne.

Pour mieux illustrer les fonctionnements de l'interface, voici une capture d'écran présentant l'ensemble des quatre TPanel et le TPageControl :

Crisegest - Fenêtre principale

Fichier Aide À propos de...

Carte

Saisissez ici l'adresse où se déroule la crise...

Valider

Street View

Trafic

Météo

Filtres

Véhicules

- Pompiers
- SAMU
- Police
- Accidentés

Bâtiments

- Avant-postes
- Hôpitaux
- Bases arrières

Autres

- Barrières
- Dangers
- Personnes

Infos générales | Infos spécifiques

Type crise : Accident de la route — Modifier

Gravité : Grave — Valider

Effet associé : Autre accident — Valider

Homogénéité : Très hétérogène — Modifier

Localisation :

Victimes...

... légères : 5 — Modifier

... graves : 42 — Modifier

... mortes : 17 — Modifier

Tout modifier

Tâches à réaliser

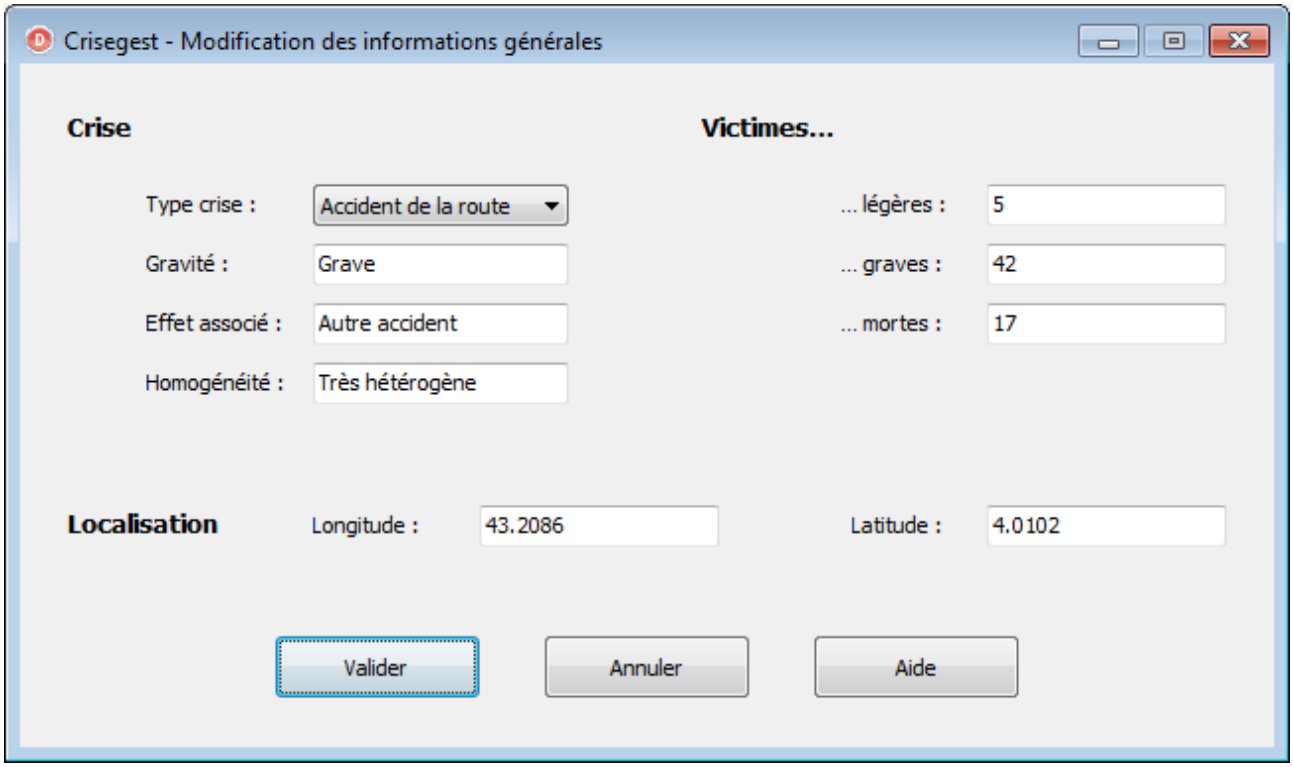
Suspendre Ajouter

Chat

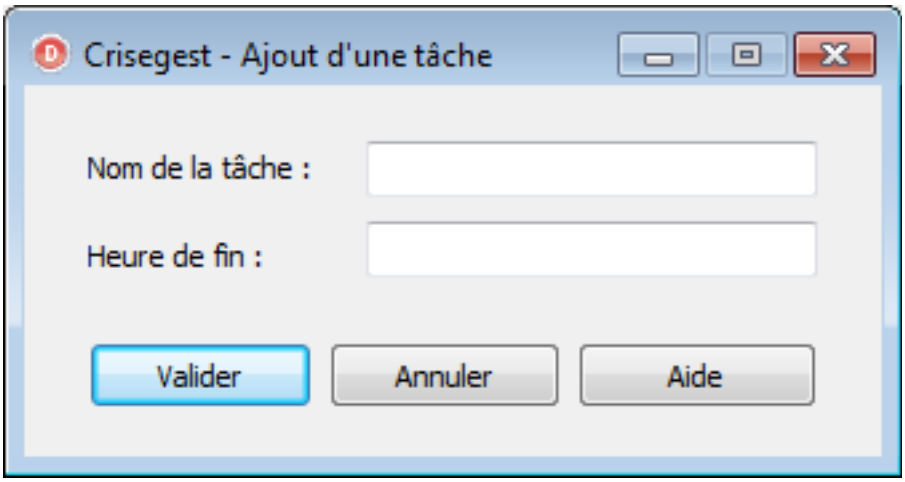
Police --> Attention !
Pompier --> Il y a le feu !
SAMU --> Où ça ?!
Police --> Partout !

Nom | Message | Envoyer

À titre d'exemple pour les fenêtres annexes s'ouvrant avec un clic sur les TButton « Tout modifier », dans le TPanel « Informations générales et spécifiques », voici la fenêtre annexe du TTabSheet « Informations générales » :



Enfin, voici la fenêtre annexe d'ajout d'une tâche qui a été développée pour le TPanel « Tâches à réaliser » :



V) Ce qui peut être amélioré

Tout ce qui avait été prévu d'être fait au niveau du TPanel « Tâches à réaliser » n'a pas été fait par manque de temps. Cette partie est donc le principal point qui reste à améliorer à l'avenir. Cependant, la fenêtre d'ajout d'une tâche a quand même été réalisée. Ce n'est donc que la partie propre à la fenêtre principale qui reste à développer.

La fonction d'impression, initialement prévue, n'a pas été développée. Cette option pourrait s'avérer très utile pour le personnel du SAMU qui pourrait alors conserver des archives papiers de ces crises.

On pourrait aussi améliorer le logiciel du point de vue de la persistance des données, et particulièrement pour les informations générales et spécifiques, en ajoutant la possibilité d'enregistrer ces données dans un fichier (il serait alors nécessaire de programmer également la lecture du fichier) et/ou dans une base de données.

Au niveau des conventions des TCheckBox, au sein du TPanel « Filtres », une convention consiste à cocher automatiquement la TCheckBox « générale » quand toutes les TCheckBox « particulières » lui correspondant ont été cochés ; et également à décocher la TCheckBox « générale » quand l'une des TcheckBox « particulières » en vient à être décochée. Cette convention n'a pas été mise en œuvre dans le projet, et pourrait l'être à l'avenir pour mieux correspondre à ce qui se fait dans ce domaine.

Conclusion

Ce projet a été l'occasion pour nous d'appliquer les notions vues en cours et en TD, et tout particulièrement le SNI, le SEF, l'utilisation de Delphi et du langage Pascal. Il nous aura aussi permis de voir davantage la continuité qu'il existe entre la conception d'une interface homme-machine via le SNI et le SEF ; et la réalisation d'une interface homme-machine via Delphi et le langage Pascal. Ce qui y a également été intéressant d'observer, c'est de voir la différence qu'il y a entre penser une interface visuelle libre de toute contrainte logicielle –premier rapport– et réaliser cette même-interface avec les possibilités et limites propres à un logiciel tel que Delphi –deuxième rapport–. Ce projet aura aussi été complet au niveau des notions abordées : création d'une carte, utilisation du drag & drop sur des éléments plaçables sur la carte, utilisation de filtres pour les éléments plaçables sur la carte, ajout et conservation de nombreuses informations et même création d'un chat.

De plus, ce projet ne nous aura pas seulement servi à utiliser les notions vues dans ce cours d'Interface Homme-Machine (ce qui est déjà pas mal, admettons-le). Il nous aura également permis de mettre un pied dans le monde professionnel, avec des conditions dignes de bons projets de stage : les délais à tenir, les joies de la communication et du travail d'équipe, la division du travail en deux grandes étapes (conception et réalisation), l'interface à réfléchir en profondeur et dans ses moindres détails, les besoins du client à garder en tête de la première à la dernière seconde, la mise en liaison des principaux avantages que doit avoir notre interface avec l'utilisation qu'en fera notre client et même la question à se poser sur qui seront (sera ?) les utilisateurs (l'utilisateur ?) de notre logiciel. Que de points communs entre un projet étudiant et un projet professionnel. On comprend tout de suite l'intérêt d'un tel projet qui, loin de se limiter à un seul cours, concerne carrément des méthodes de travail, une façon de procéder et de communiquer, et une manière de se confronter à des difficultés et de chercher à les résoudre.

Que retiendra-t-on finalement de ce projet ? Déjà, son SNI et son SEF, qui peuvent servir de très bons outils à plus d'une occasion de conception logicielle. Faire ressortir la navigation à travers les interactions et l'enchaînement des fenêtres est une excellente idée pour voir, avant même d'attaquer la réalisation, ce dont on aura besoin comme éléments visuels et de quelle manière va-t-on les placer pour qu'ils soient instinctifs et pratiques à utiliser. Pour voir la cohérence globale et l'architecture de notre logiciel, également. Nous retiendrons aussi l'importance qu'il y a à bien réfléchir chacun des éléments de notre interface pour qu'ils soient les plus adaptés possibles. Mettons un champ de texte là où une liste déroulante pourrait suffire, et nous nous retrouvons à laisser trop de liberté à l'utilisateur. Mettons des boutons radios à un endroit où nous souhaitons que l'utilisateur puisse sélectionner plusieurs réponses à la fois, et nous nous retrouvons à ne pas laisser assez de liberté à l'utilisateur. L'interface homme-machine ne fonctionne pas sur de l'à peu près, et notre client sera le premier à remarquer les défauts de notre interface si nous n'y réfléchissons pas assez. Nous retiendrons finalement l'utilisation qu'on peut avoir de logiciels comme Delphi ou même Access : la réalisation de logiciels impliquant la réalisation d'une certaine quantité d'éléments et de nombreux placements grâce à une très grande praticité et facilité de mise en œuvre des différents éléments, de leur placement et de leur personnalisation. De plus, information non négligeable, ces deux logiciels comprennent des outils permettant de manipuler avec aisance les bases de données. Cette possibilité, loin d'être anecdotique, ne peut qu'être très intéressante et utile dans un monde où l'on stocke de plus en plus de données et où la persistance de celles-ci devient primordiale au sein-même des entreprises.