Epitech

Document technique

Document réservé au développement de l'application FReg.NET

Lago_a, schehl_c, narcis_m, clique_x, tran-p_n
1/11/2010

Sommaire

Document technique

1 Rappel du contexte général

1.1 Objectif de l'EIP

Ce projet consiste en la création d'un logiciel de gestion de régate pour remplacer le logiciel existant FREG. Bien que le logiciel FREG actuel satisfasse les besoins de ses utilisateurs, la FFV souhaite disposer d'un logiciel qui lui appartienne et développé dans des langages actuels pour en permettre une meilleure évolutivité. C'est pourquoi la Fédération a fait appel à Epitech et à ses étudiants pour mener à bien le développement d'une nouvelle version de leur logiciel ; en utilisant les dernières technologies et en adoptant une architecture modulaire. La Fédération compte utiliser ce nouveau produit sur la scène internationale.

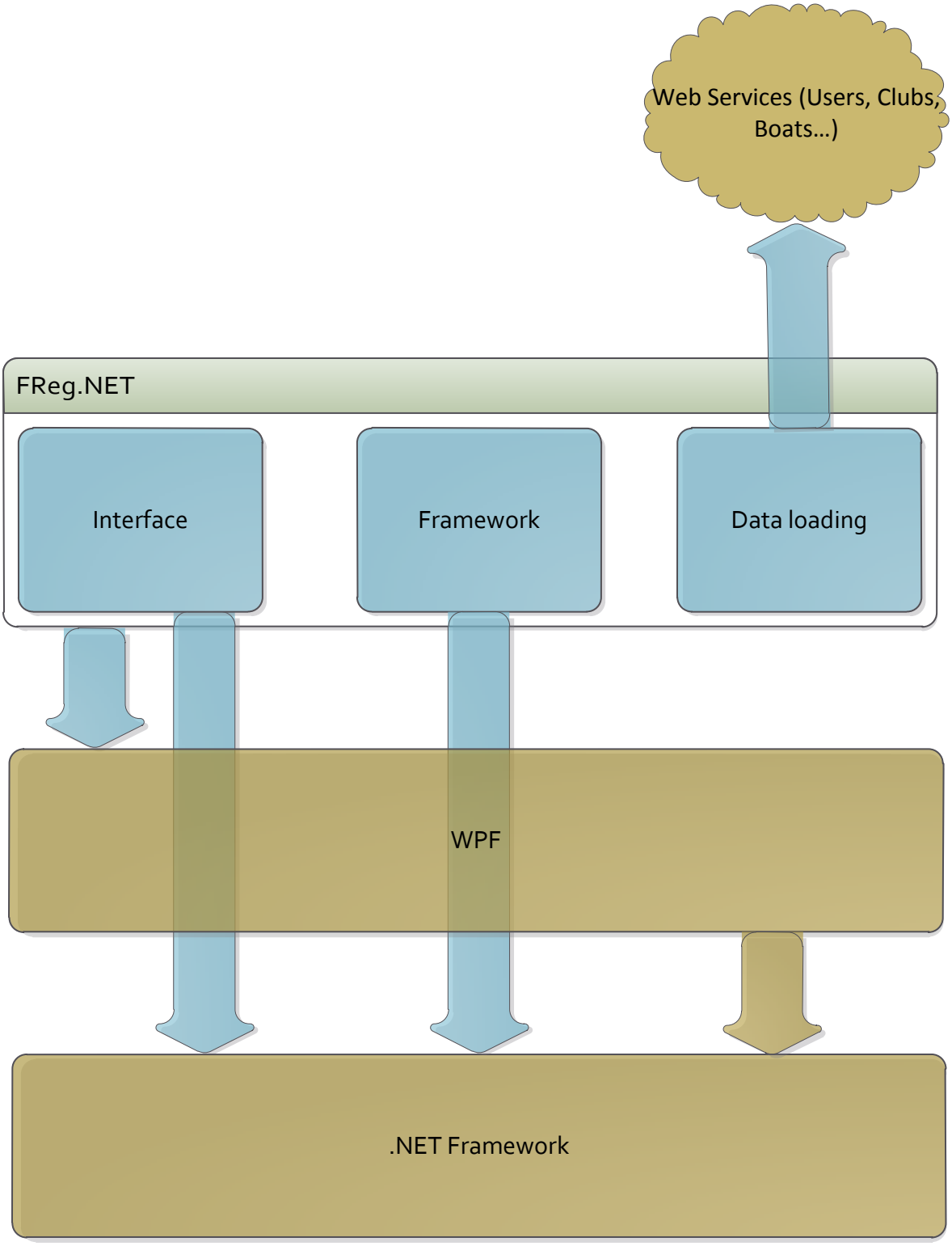
1.2 Résumé du document

Ce document s'adresse principalement au futur développeur du projet. Il s'agit de présenter dans les détails les différents modules développés afin de bien prendre en compte les besoins des utilisateurs et de ne pas les perdre de vue. Il regroupe également tous les diagrammes représentants l'architecture du logiciel.

Ce document est également utile si un développeur souhaite continuer le projet. Il peut voir le projet dans sa globalité, son architecture, ses modules en détail et les choix technologiques fait et avoir un aperçu des prévisions de développement grâce au diagramme de GANTT.

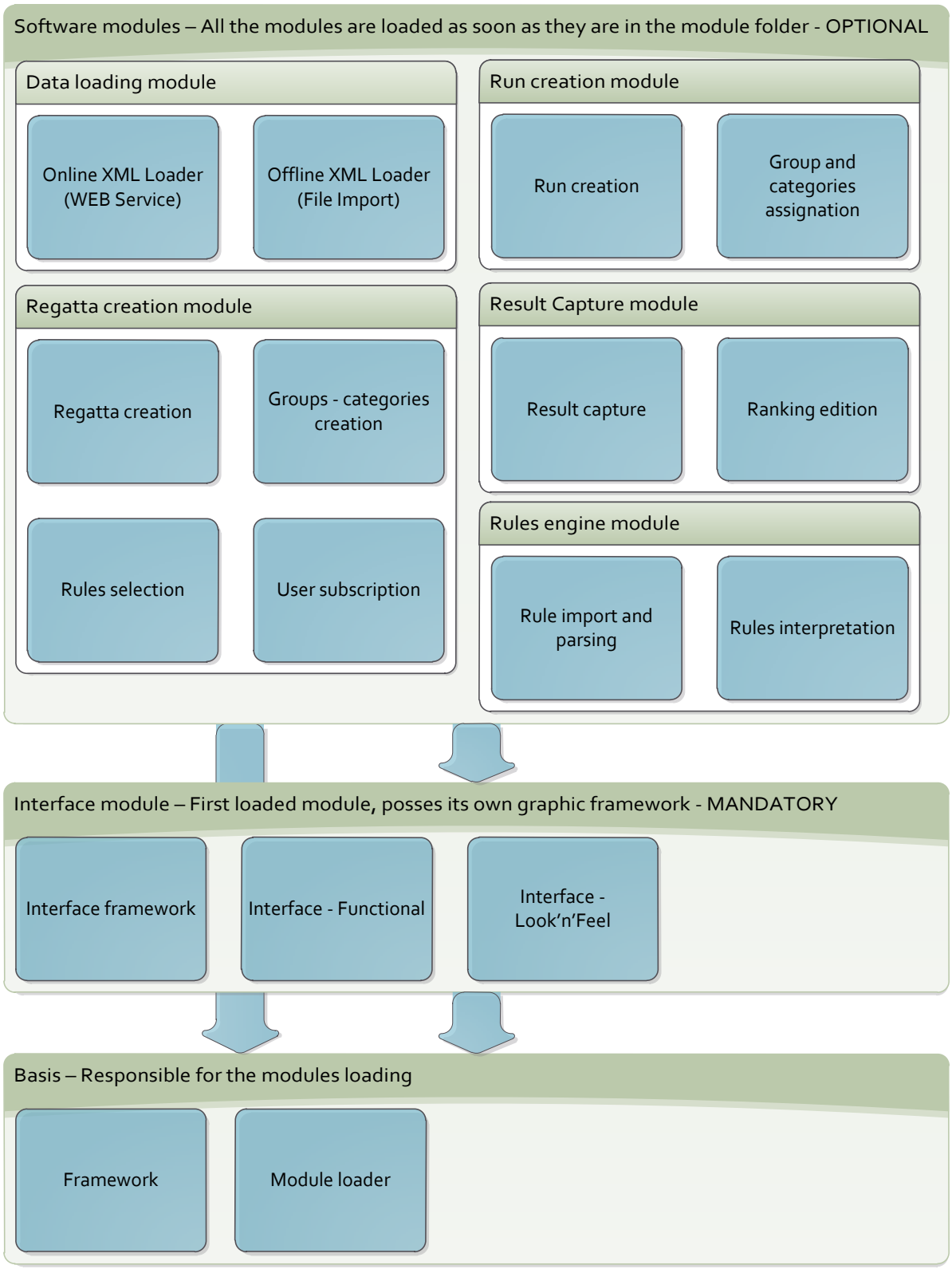
2 Diagrammes

2.1 Diagramme global du projet



2.2 Diagramme détaillé

2.2.1 Vue générale du projet

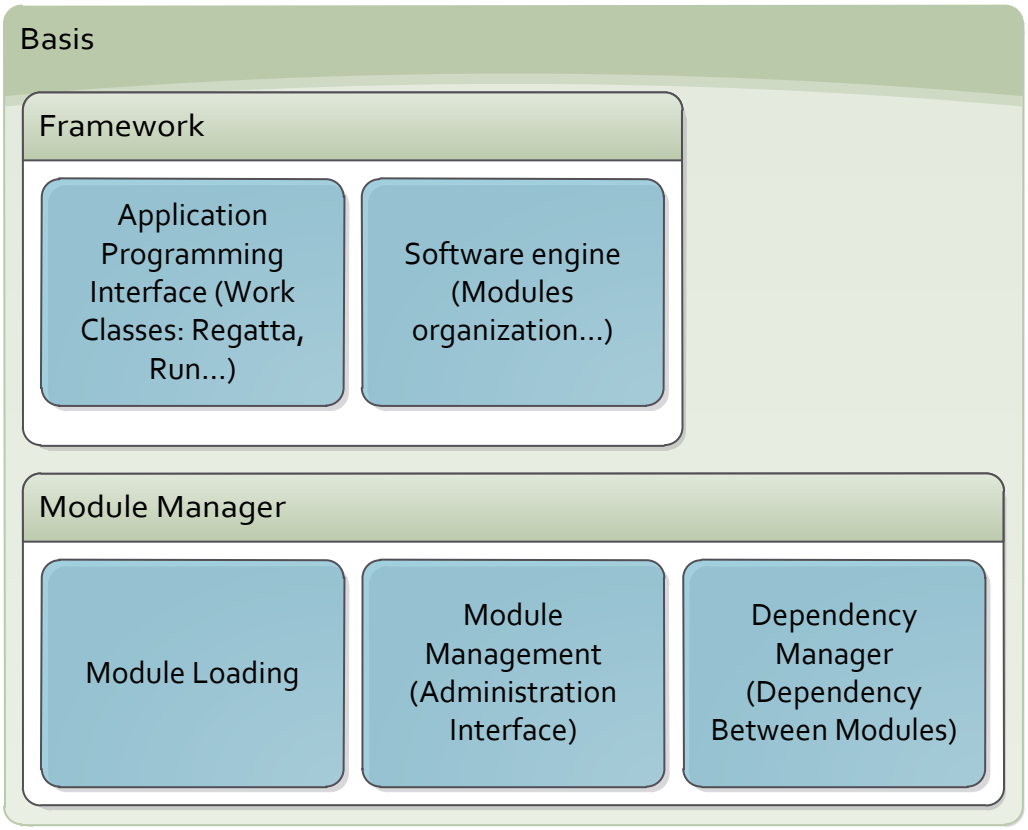


Tout le projet sera découpé en plusieurs modules. Permettant ainsi de s'assurer de la pérennité du projet malgré la fin de l'EIP, cela dans l'objectif de fournir au client une base solide et évolutive.

2.2.2 Vue détaillée de la base

La base du logiciel se chargera de s'assurer du bon fonctionnement des modules. Ces modules pourront communiquer ensembles à l'aide de l'API.

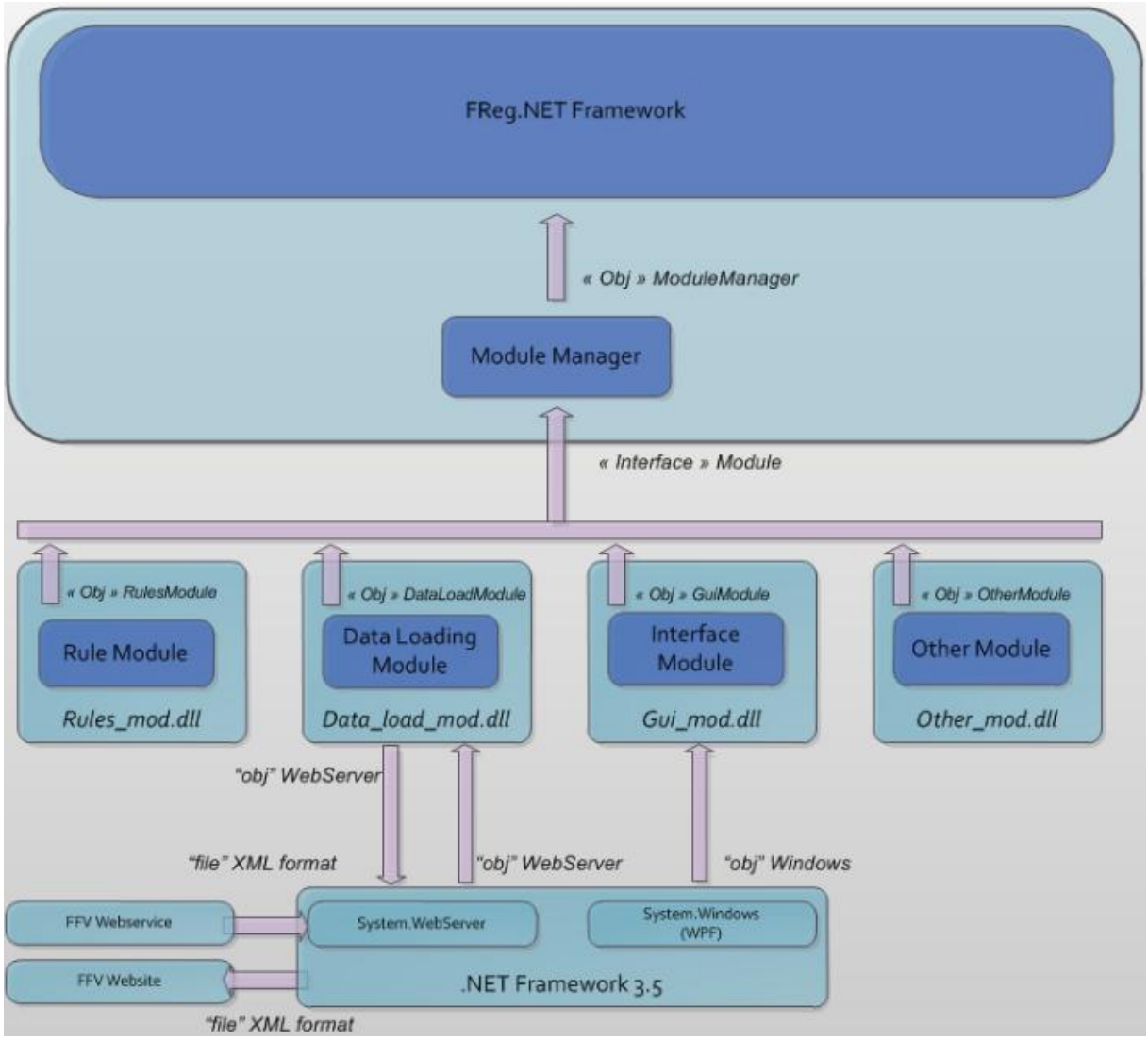
Le module Manager sera responsable du chargement des modules ajoutés dans l'application, il vérifiera aussi les dépendances inter modules.



2.3 Diagramme de communication

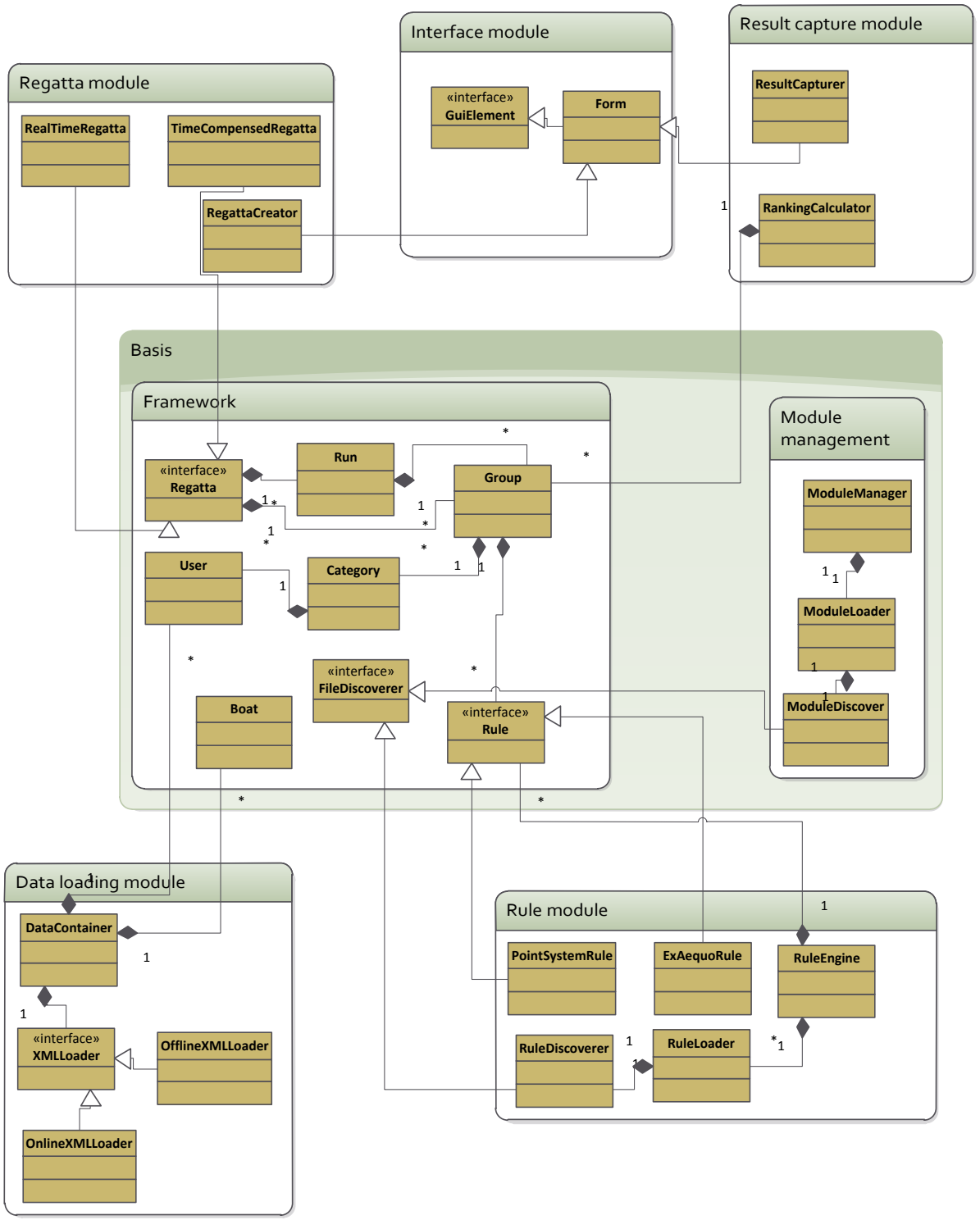
FReg.NET utilisera principalement les bibliothèques incluse dans le Framework .NET dont Windows Presentation Fundation qui permet de réaliser des interfaces utilisateurs intuitive et ergonomique qui a été introduit dans la version 3 du Framework.

De plus le module de chargement des données sera en charge de communiquer avec les web services mis en places par la FFV pour la récupération des différentes données telles que les Clubs, Coureurs, Bateaux…



2.4 Architecture des classes

Le cœur du programme contiendra des interfaces permettant aux modules de spécialiser leur comportement et de communiquer ensembles. Ces modules, hormis le module d'interface qui est chargé automatiquement devront rester indépendant dans leur fonctionnement pour permettre a un utilisateur de n'utiliser qu'un seul s'il le souhaite.



3 Choix des technologies

Le programme est réalisé à l'aide du Framework .NET technologie très fortement supportée par Microsoft. De plus nous utilisons la bibliothèque WPF inclue dans le Framework, qui permet de réaliser rapidement des applications soignée est très fonctionnelles. La bibliothèque Composite Application Library est elle-même utilise pour réaliser l'architecture en module du logiciel ; cette dernière prend en charge le chargement des modules et leur organisation de façon logique a l'intérieur du programme.

3.1 Le MVVM

Le programme est développé en respectant le Design Pattern MVVM (Model View ViewMode) qui permet une séparation totale du code et de l'interface. Le programme s'axe atour de Vues qui utilisent des VueModel pour leur donner accès aux données de l'application. Les VuesModels utilisent le Model pour donner accès aux données aux vues.

3.2 Les avantages

La technologie .NET présente le premier avantage d'être simple à mettre en place et déployer dans le contexte de la fédération française de voile du fait des accords entre cette dernière et Microsoft.

De plus le développement est facilité grâce à l'utilisation de bibliothèques telle que Windows Presentation Fundation qui permet de réaliser rapidement des interfaces utilisateurs conviviales et ergonomiques.

Avec l'utilisation du Framework .NET, nous respecterons les conventions de codage établies par Microsoft, permettant une reprise ultérieur de l'application facilitée.

Grâce à la gestion automatique des modules de la Composite Application Library nous pouvons gagner du temps sur la gestion des modules et mettre à disposition des développeurs tiers une API fonctionnel et efficace.

3.3 Les inconvénients

En contrepartie, due au fait des accords entre la fédération et Microsoft, il serait très compliqué de changer de technologie en cas de besoin.

Le deuxième problème majeur que nous rencontrerons avec cette technologie et que nous ne la connaissons que très peu et qu'un temps d'apprentissage sera nécessaire.

Pour finir le programme du fait de sa réalisation dans cette technologie ne sera que peut portable et sera principalement utilisable sur les plateformes Windows équipées du Framework .NET ; cependant pour la FFV, le cœur des utilisateurs est sur Windows et cette contrepartie est compensée par la facilité de développement et les performances du Framework.

4 Modules développés

4.1 Présentation générale

FReg.Net reprendra la majeure partie des fonctionnalités de son prédécesseur FReg à la différence que FReg.Net utilisera les technologies .NET qui permettront d'améliorer l'utilisation du logiciel aussi bien ergonomiquement que fonctionnellement.

FReg.Net se composera de quatre modules principaux :

- Un module de chargement des données, récupérant toutes les données nécessaires au fonctionnement du logiciel (adhérents, jurys, …)
- Un module de création de régates, utilisé pour regrouper toutes les données de la Régate (inscrits, règles, groupe de classement, …)
- Un module de capture des résultats, qui permettra à l'utilisateur d'ajouter les arrivées des participants
- Un module de génération des classements et rapports

Une interface permettra à l'utilisateur de naviguer à travers ces différents modules. Elle respectera la démarche d'ergonomie actuelle pour les logiciels. FReg.Net se devra d'être facile et agréable à utiliser. Pour cela les modules principaux seront présents dès l'ouverture du logiciel, avec une explication partielle de chacun afin que l'utilisateur puisse rapidement repérer le module qui lui correspond.

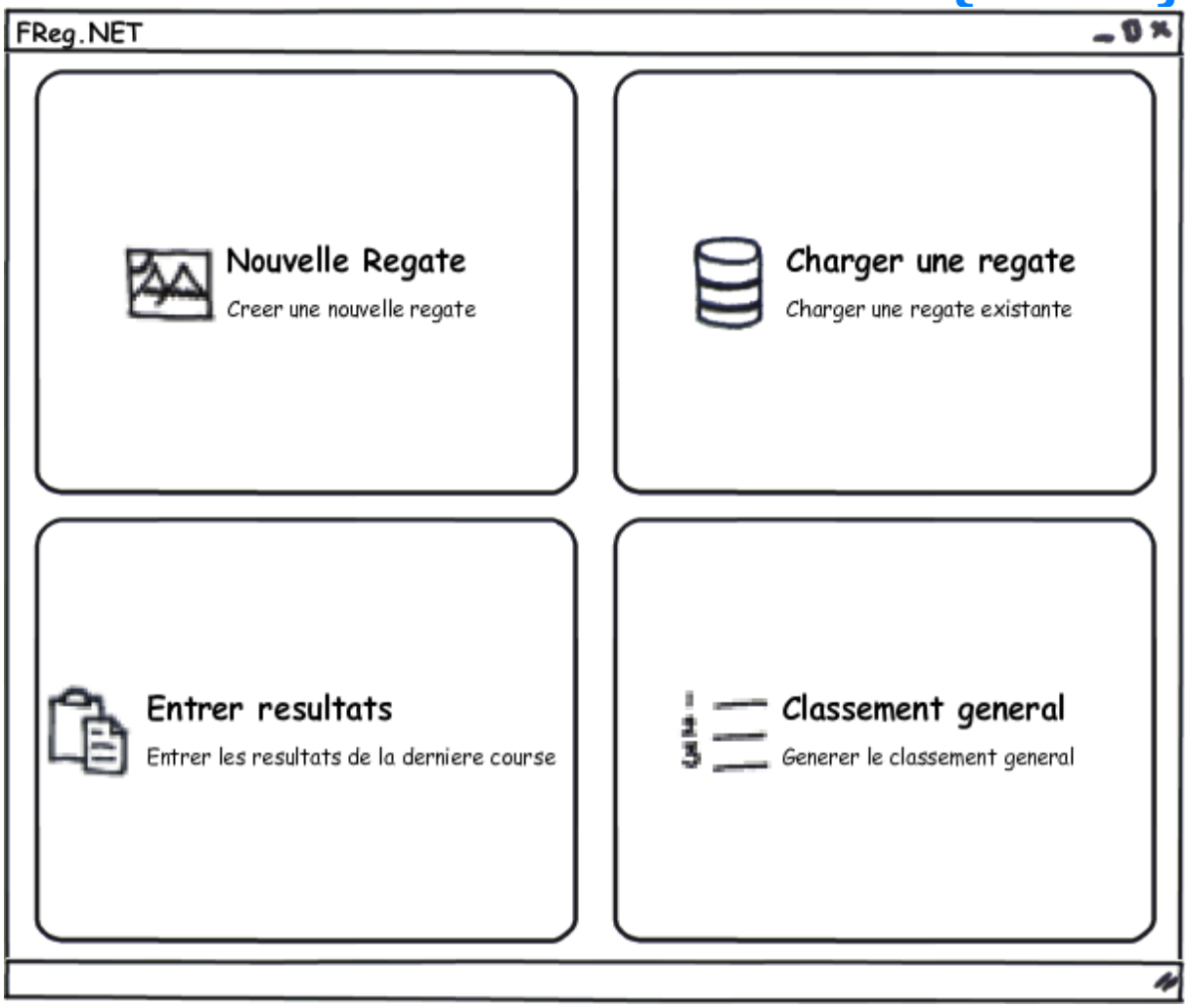


Figure 1 : Image mockup - Ouverture du logiciel FReg.Net - Créée par Matthieu NARCISI

A chaque formulaire rencontré l'utilisateur aura également accès à un assistant qu'il pourra, s'il le décide, cacher.

Il y aura également un Framework qui va référencer toutes les classes (club, coureur, …) qui seront utilisées par les modules afin d'harmoniser le fonctionnement du logiciel.

Et enfin, un moteur de règle sera mis en place pour que l'utilisateur puisse gérer les règles présentes dans le logiciel comme il le souhaite.

4.2 Chargement des données

Le module de chargement des données utilisera un format XML pour la récupération des données sur les serveurs de la fédération. Ces données seront disponibles :

- En WebService : lorsque le logiciel est connecté à Internet, il sera capable d'aller vérifier automatiquement et périodiquement la présence de mises à jour.
- En fichiers XML : lorsque le logiciel est en mode déconnecté, il sera capable de recevoir des fichiers XML téléchargés sur le site de la fédération pour mettre à jour ses données.

Ce module sera responsable de la vérification de la date de la base de données ; en effet, si celle-ci se révèle trop ancienne (2-3 jours à définir), il bloquera le logiciel empêchant ainsi des régates de se faire avec une base de données non à jour.

Il utilisera une interface de façon à abstraire totalement le fonctionnement du chargement des données, il sera ainsi très facile de rajouter une interface permettant le chargement des données dans un autre format que le XML (YAML, JSon, SQL…).

Ce module sera administrable dans un onglet propre du panneau d'administration de FReg.NET : il sera par exemple possible de charger un nouveau fichier de données ou de provoquer manuellement la mise à jour de la base depuis internet.

4.3 Création de régates

Le module représentera deux des boutons présents dans l'univers d'accueil ; l'un permettant de créer une nouvelle régate et l'autre permettant d'en charger une existante.

Une fois que l'utilisateur entre en mode création ou édition de régate, il entre dans mode « wizard » lui permettant de configurer tous les aspects de la régate pas à pas. A chaque passage à l'étape suivante avec le bouton suivant, les données de la régate actuelle seront automatiquement sauvegardées.

NOTE : En mode édition l'utilisateur pourra passer directement d'une étape à l'autre pour éditer rapidement les groupes de classement ou les inscrits. 6 | FReg.NET

Les étapes du Wizard reprenne dans leur ordre les étapes de création d'une régate dans FReg 2009, et ce dans le but de ne pas perturber les utilisateurs du logiciel existant.

4.3.1 Première étape : création d'une régate

Sur cet écran, l'utilisateur pourra entrer les informations générales de la régate, celle-ci serviront à l'édition des résultats par exemple ou pour l'envoi des données à la presse (avec par exemple la présence d'un champ dédié aux sponsors).

Cette étape servira aussi à l'utilisateur pour définir les dates de début et de fin de la régate.

4.3.2 Deuxième étape : définition des règles

Dans cette étape il sera possible de définir les règles de comptage de points qui s'appliqueront à la régate. Ces règles sont définies dans les instructions de course et dans l'avis de course.

Suivant le système choisit les champs et écrans suivants seront différents, par conséquent la modification de certains paramètres de cet écran entraineront la perte d'une parties des données de la régate si l'utilisateur édite une régate existante.

Il sera ensuite possible d'imprimer ces règles pour qu'elles soient vérifiées par le Président du Comité de Course et par le Président du comité de réclamation, puis affichées au tableau officiel.

4.3.3 Troisième étape : définition des groupe de classement

Le principe des groupes de classement sera le même que FReg 2009. L'utilisateur aura la possibilité de créer plusieurs niveaux de classement (Scratch, par groupe, par Classe/Catégorie).

4.4 Capture des résultats

Le module de capture des résultats permettra à un membre du comité de course de saisir l'ordre d'arrivée des concurrents. C'est à ce moment-là que l'utilisateur peut ajouter les pénalités signalées, mais aussi lister les participants qui ne sont pas encore arrivée.

A la fin de la saisie une vérification est effectuée par le logiciel au niveau des éventuels doublons ou anomalies. L'utilisateur est alors averti que sa saisie n'est peut-être pas juste.

S'il s'agit d'une course en temps il faut pour cela, indiquer l'heure de départ des différents concurrents. La saisie des résultats est alors un peu différentes car il s'agit de mentionner le jour et l'heure d'arrivée pour chaque concurrent. Les rangs sont alors calculés automatiquement à chaque ajout. A la fin de la saisie comme pour une course normale, s'il existe des erreurs l'utilisateur sera prévenu.

A la fin d'une course des pénalités pourront être appliquées (après décision du jury), l'utilisateur pourra alors choisir dans une liste de pénalité qui aura des influences sur le classement qui sera recalculé automatiquement. Il aura également la possibilité de reclasser un concurrent suivant les différents événements qui se sont produits au cours de la course (réparation, rectification de temps, …).

4.5 Générations des classements et rapports

4.5.1 Général

A chaque formulaire rencontré par l'utilisateur, celui-ci se verra la possibilité d'imprimer les informations qu'il vient de saisir. Ceci permettre alors de générer les listes de participants, mais aussi les règles avant le début de la course, …

Plusieurs formats d'impressions seront alors disponibles :

- HTML, pour le site internet
- Export en PDF
- RTF (pour permettre le copier/coller dans Microsoft Word)

L'utilisateur pourra choisir dans la liste des champs d'informations saisies celles qu'il souhaite éditer ou non.

4.5.2 Courses

Pour chaque type de saisie il sera possible d'imprimer les ordres d'arrivées, même si la saisie n'est pas complète. Une liste des concurrents toujours pas arrivés pourra également être faite pour rechercher ces derniers (il s'agira des concurrents étiquetés « absent à l'ordre d'arrivée »).

Si les classements sont validés ils pourront être édités, sinon l'impression sera bloquée. Les formats d'impressions seront alors les mêmes que pour les formulaires généraux.

L'utilisateur pourra également sélectionner les champs qu'il souhaite éditer (club, temps, …).

4.6 Le Framework

Le Framework contient des classes simples permettant aux modules de manipuler les objets concernes par FReg.NET.

Ces classes sont situées dans le répertoire Core/Framework du projet. Actuellement les classes développées sont :

- Runner (Courreur) : représente un membre d'un club par lequel il est rattaché par le nom de club. Une méthode permet dans cette classe de récupérer le club associe.
- Club : représente un club qui est rattaché à un plus haut niveau à une ligue. Contient en plus de ses attributs basiques, une liste de Runner qui représente tous les membres du club.
- Regatta (Régate) : représente une régate, cet objet est instancie lors de l'entrée dans le module Regatta. Il contient toutes les informations de la régate (nom, date de début, date de fin…).

4.7 Le Kernel

Le Kernel est la base du logiciel, il est la première partie du programme exécutée et est responsable de démarrer tout le reste du programme. Le Kernel se base sur la bibliothèque Composite Application Library, celle-ci se charge de charger tous les modules présents dans le répertoire Modules.

Le Kernel contient actuellement les classes suivantes :

- Bootstrapper: première classe du programme instancié, elle se charge d'ouvrir la vue principal du programme et de l'afficher.
- Le Shell : Ce composant se compose de deux classes ; la vue et le vue model, ce dernier se charge de récupérer les évènements qui adviennent sur le Hub pour ensuite charger le module correspondant.
- Le Hub : Cette vue contient une liste de boutons que les modules peuvent utiliser pour ajouter un lien vers leurs fonctionnalités à l'accueil du logiciel.
- Le FormView : cette classe est la vue par défaut de tous les formulaires de FReg.NET, elle contient toutes les régions necessaire à la création d'un formulaire :

- Le Menu : une fois le formulaire chargé, le Hub disparait, laissant alors la possibilité aux modules d'afficher une version plus petit de leur bouton.
- La Région Principale : dans cette région, les modules peuvent placer leur formulaire à proprement parler. C'est ici que peuvent être placés tous les champs des formulaires.
- L'Assistant qui contient trois régions :
  - L'aide : lorsqu'un champ gagne le focus, l'aide correspondant est affiché dans cette case.
  - Les erreurs : lorsqu'un champ d'un formulaire perd le focus, le Moteur d'Erreurs vérifie automatiquement les règles de validations associées au champ pour ensuite les afficher.
  - La progression : cette région présentera l'avancement actuel dans le formulaire.

4.8 L'Infrastructure

Ce projet, compilé sous la forme d'une DLL commentée contient toutes les classes qui sont mises à la disposition des développeurs souhaitant réaliser leur propres modules. Elle contient notamment les classes suivantes :

4.8.1 Evénements

Ces classes représentent les événements que le développeur peut publier ou bien souscrire ses propres méthodes. Ils sont automatiquement publiés par le Kernel ou récupéré par certaines parties de l'interface.

4.8.1.1 ErrorsChangedEvent

Cet événement qui transporte un ErrorReport (rapport d'erreur) est récupéré par l'assistant pour afficher dans la région consacrée aux erreurs, l'erreur contenue dans le rapport, l'assistant vérifie avant d'ajouter l'erreur a la liste, si celle-ci est déjà présente, auquel cas il ne l'ajoutera pas.

4.8.1.2 HelpChangedEvent

Cet événement qui transporte une chaine de caractères est récupéré par l'assistant qui affiche à la réception le contenu du message dans sa zone d'aide.

4.8.1.3 HubComponentChoosedEvent

Cet événement qui transporte une chaine de caractères est récupéré par le Kernel qui va alors décharger le Hub de la vue principale et charger la vue de formulaire pour laisser au module la possibilité de charger ses propres formulaires. Le module qui envoi cet événement doit préciser dans la chaine de caractère en paramètre le nom de son module.

4.8.1.4 FormLoadedEvent

Cet événement qui transporte une chaine de caractère est envoyé par le Kernel lorsqu'il a fini de décharger le Hub et de charger la vue Formulaire. La chaine de caractère représente le nom du qui a envoyé l'événement HubComponentChoosedEvent, les modules abonnés à cet événement

peuvent alors vérifier s'il s'agit d'eux et ainsi afficher leur formulaire ou à défaut le bouton dans la région du menu pour continuer à être accessible.

4.8.2 Model

Ces classes représente les objets auquel les modules ont accès pour communiquer avec l'Infrastructure ou les autres modules ; principalement à l'aide des événements.

4.8.2.1 AssistantErrors

Cet objet est utilisé pour stocker une règle de validation pour l'assistant, elle contient le nom du champ à laquelle elle s'applique et le message à afficher en cas d'erreur. Elle contient de plus le nom de la règle à appliquer (taille maximale, minimale, comparaison de date…) ainsi que la valeur de comparaison (une valeur numérique dans le cas d'une taille, le nom d'un autre champ dans le cas d'une comparaison de date…)

4.8.2.2 AssistantHelp

Cet objet contient les informations nécessaires à l'affichage d'un message d'aide pour un champ donné, il ne contient donc que le nom du champ et le message à afficher.

4.8.3 ViewModel

Toutes le classes contenues dans ce dossier sont des classes abstraites qui peuvent être étendue pour bénéficier de comportement par défaut dans les modules crées par les développeurs.

4.8.3.1 ViewModelForm

Cette classe propose une implémentation de base pour une VueModel de formulaire, mettant à la disposition du développeur une gestion de l'assistant avec les erreurs et les l'aide déjà complète.

4.8.4 Autres clases

Ces classes font partie du namespace de base de l'Infrastructure et mettent différentes fonctionnalités à la disponibilité des développeurs.

4.8.4.1 RegionNames

Cette classe contient une série de chaînes de caractères statiques pour donner la possibilité au développeur d'avoir accès à toutes les régions de l'interface. Il peut, en utilisant la Composite Application Library, spécifier dans quelles régions il veut placer ses vues.

4.9 Le Module Regatta

Ce module est le premier développé, il contient les formulaires de création de régate. Il contient aussi sa propre implémentation des régions d'aide et d'erreur dans l'assistant.

5 L'interface

L'interface respectera une charte graphique préalablement établie.

5.1 Ouverture du logiciel

A l'ouverture du logiciel, un écran d'accueil apparait avec les 4 modules importants de saisie :

- Création d'une régate

- Charger une régate existante
- Entrer résultats
- Classement général

Une explication sous chaque module et des noms explicites permettent à l'utilisateur de comprendre rapidement l'utilité du module.

Une fois un module choisit, l'utilisateur peut choisir à tout moment d'abandonner ce qu'il faisait et revenir soit à l'accueil soit cliquer sur un des autres modules qui seront présents dans le haut de la fenêtre. Les informations alors saisies ne seront pas sauvegarder automatiquement et l'utilisateur devra faire le choix de les conserver et d'y revenir plus tard. 8 | FReg.NET

5.2 Formulaires

Pour la présentation d'un formulaire, s'il se compose en plusieurs parties (feuillets), des boutons suivants et précédents permettront à l'utilisateur de naviguer comme il le souhaite à travers sa saisie. Il devra néanmoins faire attention aux éventuelles erreurs mentionnées par l'assistant (cf. Assistant).

Une barre de progression sera visible et l'utilisateur pourra également cliquer sur une étape précédente (ou suivante, s'il y est déjà allé) pour y revenir. Ceci permettra à l'utilisateur de se situer plus facilement.

5.3 Assistant

Dans toutes les étapes sera présent, sur le côté de l'écran, un assistant composé de trois sections, ce panneau pourra être refermé afin que l'utilisateur gagne de la place sur son interface :

- L'aide : la première section présentera des informations sur le champ que l'utilisateur sera en train de remplir. Ceci évitant aux nouveaux utilisateurs d'avoir la documentation ouverte en permanence à côté du logiciel.
- Les erreurs : en plus d'une information visuelle sur les champs (contour en rouge), ce champ répertoriera toutes les erreurs de formulaire présent sur la page dans laquelle l'utilisateur travail. Permettant ainsi à l'utilisateur de mieux les comprendre et de revenir sur celle-ci avant de valider la page. Suivant ce procédé, le bouton permettant de passer à la page suivante ne sera pas accessible tant que toutes les erreurs n'auront pas été corrigées.
- L'avancement : cette section présentera à l'utilisateur son avancement dans le processus de création d'une régate et une description plus complète de l'étape à laquelle il se trouve actuellement. Cela lui permettra d'anticiper les formulaires à remplir à tout moment.

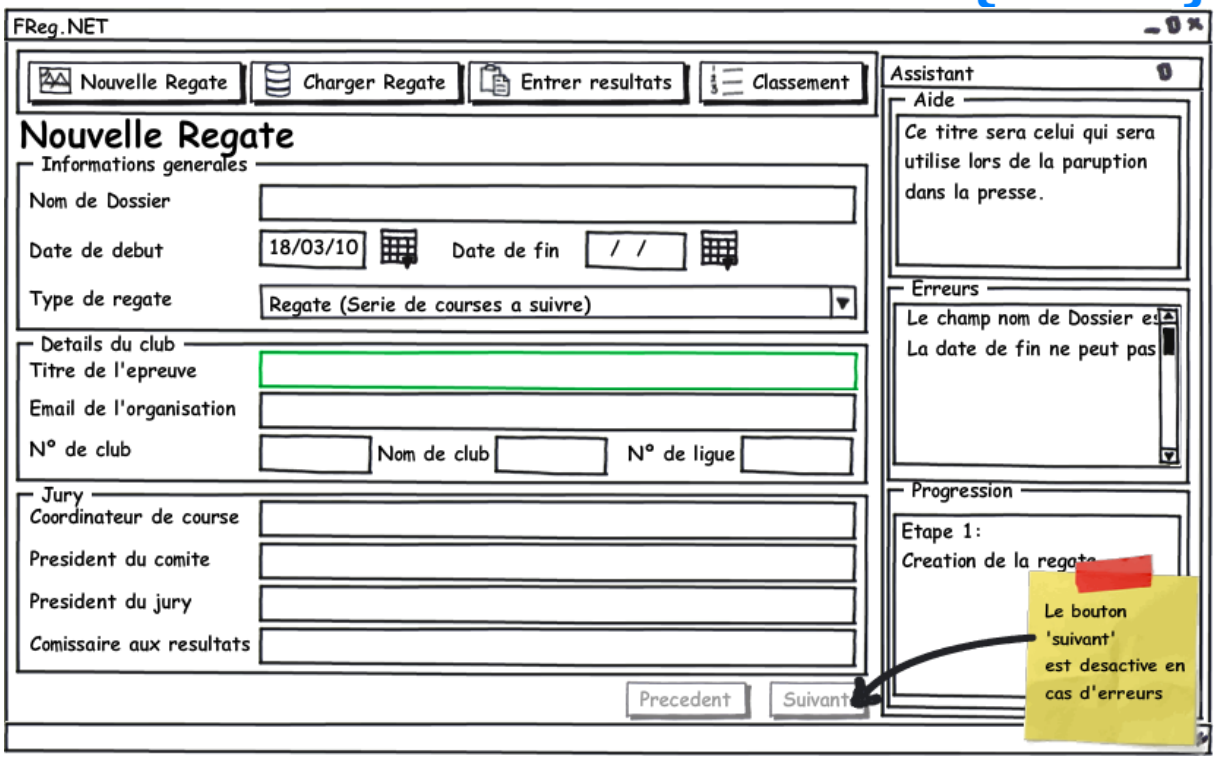


Figure 2 : Image mockup – Formulaire et Fenêtre Assistant – Créée par Matthieu Narcisi

6 Moteur de règle

Les règles de voiles sont complexes et souvent difficiles à interpréter. Toutes les règles nécessitant d'être ajoutées au logiciel auront une influence sur les points et donc sur le classement. Il peut s'agir d'une pénalité de temps (suivant la faute commise par le concurrent), mais aussi un reclassement après acceptation d'une réparation, …

Après ajout d'une règle le classement sera recalculé automatiquement en fonction des caractéristiques de la règle choisie.

Un moteur de règle sera alors implémenté afin de permettre à l'utilisateur de créer, mettre à jour ou encore supprimer une règle.

L'utilisateur devra renseigner les différents changements sur les points ou le classement qu'impliquera la règle.

7 Bugs connus

Voici la liste des bugs connus lors de la rédaction de ce document, pour une liste actualisée, consultez le Trac du projet : https://labeip.epitech.eu/2011/ffv/, ces bugs seront corrigés dans les prochaines révisions du logiciel.

- Lors du passage du formulaire de création de régate à la deuxième page, la partie réservée à l'aide dans l'Assistant reste sur le dernier champ sélectionné.

- Le nom des champs affichés dans la partie erreurs de l'assistant est le nom du champ dans le code, celui-ci n'est pas très « User Friendly » il faudrait afficher la version française du champ.

8 Diagramme de GANTT

ID	Component type	Task Name	Resource Names	Start	Finish	Duration
1	BASIS	UML conception (basis)	Narcis_m (all group)	08/03/2010	27/03/2010	3W
2	BASIS	Basis implementation	Tran-p_n (narcis_m)	29/03/2010	10/04/2010	2W
3	ANALYSIS	Rules analysis	Lago_a	08/03/2010	31/07/2010	21W
4	MODULE	Data format definition	Narcis_m (lago_a)	29/03/2010	10/04/2010	2W
5	MODULE	Data import Module	Tran-p_n (narcis_m)	12/04/2010	24/04/2010	2W
6	INTERFACE	Interface	Schehl_c (clique_x)	29/03/2010	17/04/2010	3W
7	DOCUMENTATION	API and interface documentation	Narcis_m	19/04/2010	01/05/2010	2W
8	MODULE	Regatta creation conception	Lago_a (schehl_c)	19/04/2010	01/05/2010	2W
9	MODULE	Rules modification	Clique_x (tran-p_n)	19/04/2010	01/05/2010	2W
10	TESTS	Functionnal tests	Lago_a	03/05/2010	08/05/2010	1W
11	DOCUMENTATION	Modules documentation	Lago_a	10/05/2010	15/05/2010	1W
12	MODULE	Group creation/User subscription	Schehl_c (narcis_m)	03/05/2010	15/05/2010	2W
13	MODULE	Run creation	Tran-p_n (clique_x)	03/05/2010	15/05/2010	2W
14	TESTS	Functionnal tests	Lago_a	17/05/2010	22/05/2010	1W
15	MODULE	Result capture	Clique_x (lago_a)	17/05/2010	29/05/2010	2W
16	MODULE	Rule engine	Narcis_m (schehl_c, tran-p_n)	17/05/2010	12/06/2010	4W
17	MODULE	Ranking edition	Schehl_c (clique_x)	14/06/2010	19/06/2010	1W
18	MODULE	Ranking printing	Tran-p_n	14/06/2010	19/06/2010	1W
19	DOCUMENTATION	Rules script documentation	Lago_a (narcis_m)	14/06/2010	26/06/2010	2W
20	TESTS	Functionnal tests	Lago_a	21/06/2010	26/06/2010	1W