Fields group specification document

General architecture

This unit of the system has the duty to mantain and manage the type system of the elements that have to be used inside user-created forms.
This unit is entirely located on the server side of the architecture, interacting directly with units Workflow (server side), GUI and Workflow Editor (client side).

The comunication interface with client units is implemented using ASP.NET and the library has been implementated using C# language.

The library is structured in three logical elements:

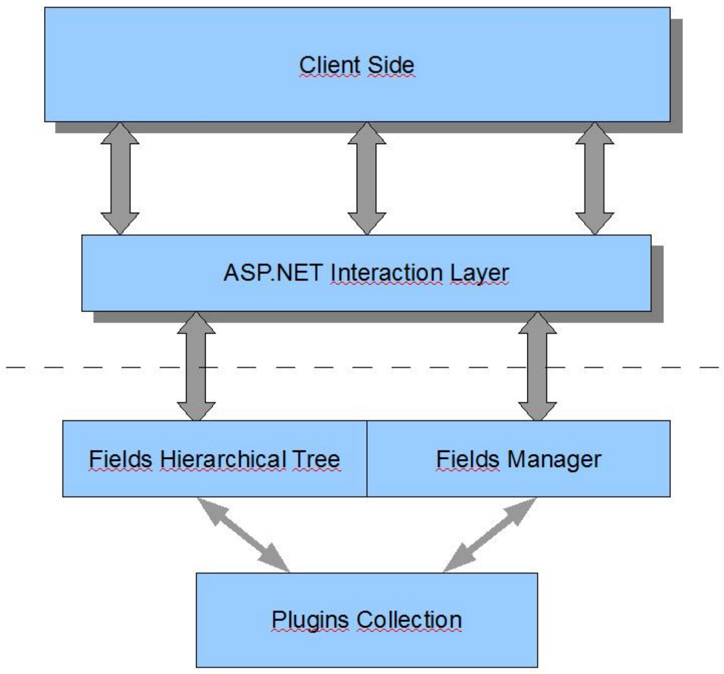
-Fields Hierarchical Type Tree
-Fields Manager
-Plugins Collection

The Fields hierarchical type tree describes the basic types that can be used to compone a form. All elements that have to be used in the client application implement a common interface ("IField") , and must extend the System.Web.UI.WebControls.WebControl class to give the possibility o be rendered.

The Fields Manager section is used to negotiate the specification of instances of the types defined in the type tree and to collect the data contained in the plugin collection.

Plugin collection consists of a set of dynamically linked libraries which define the data types that can by used by the Workflow Editor unit to compose forms.

Next image shows how is composed the Fields unit and how it relates to the client side of the application which is composed by the GUI and WorkFlow Editor units.
The middle line separetes the part of the architecture composed by the Fields unit from the "rest of the world".



The Client-side interaction style does not have common points with the way WorkFlow unit communicates with Field. WorkFlow queries directly the Fields Manager section of the unit to discover which "predicates" and "operations" can be used with each Field element (predicates and operations are discussed later in the document.

Fields Section

Below is given a representation of the Hierarchical Type Tree, which is defined in its leaves by the content of the dynamically loaded plugins.
All these elements inherit directly from the base interface IField and abstract class WebControl which define the methods to manipulate and validate the components of user-defined forms.
For rendering purposes, each element is exposed as a WebContol element, which is offered to the GUIs in two different versions: a Preview Mode, used inside WorkflowEditor to give a visual evaluation of the element in junction with the look-and-feel defined by Theme Editor unit, and an Edit Mode which permits to edit the value contained inside the element and represents the default modality used by the Client GUI to present the form content to the end user.
Each Field must give the clients the ability to validate the content of the filled form on the GUI module, using the constraints specified by the designer of the form via Workflow

Editor, comparing the values contained in a XMLNode delivered by the client.

For the validation phase of the form-filling process, it has been decided to rely to the functionalities of the Validators elements that can be associated to a WebControl.

Fields Manager

The Fields Manager section gives to the client side of the application the possibilty to find which data types are defined in the Fields Tree and to discover constraints, operations, predicates available, offering the possibility to use more complex transition conditions to connect various states of the form.
To discover the types defined inside the plugin collection, the user can call the GetTypes() method which retrieves a list of Type variables which, via reflection, can be used in junction with GetInstance() to create a new instance of a specific Field type (See "Use Cases" section to find out how).
To find which predicates, operations and constraint are defined on each IField implementation

----

USE CASES

Workflow Editor

-> Definition of a new element of a Form (base type)

Request of usable data types ==> Returns the whole list of defined types

List<Type> lt=FieldsManager.GetTypes();
//esempio d'uso

Choose the type to use ==> Returns an instance of the requested type

IField f = FieldsManager.GetInstance(t);

-> Association of a constraint to a form element

Request for the defined constraints for a specific instance ==> Returns a list which contains the defined constraint for the type of the instance

List<MethodInfo> lm=FieldsManager.GetConstraints();

Choose of the needed constraint (with requested parameters) ==> Adds inside the instance the chosen constraint

f.Invoke(lm[2],args);//invokes the third method of the list with args arguments on the IField f instance

-> XSD structure generation from an element

Generation request of the XML structure of the field==> Gives the complete XML schema of the field which describes the structure of the element and the constraints defined on the instance

```
XmlSchemaComplexType xsd=f.GetSchema();
```

GUI

-> Element rendering request

Richiesta di un'istanza del tipo a partire da un elemento complesso XSD ==> Restituzione di un'istanza che rappresenta l'elemento da rappresentare con i vincoli associati

```
XmlSchemaComplexType schema;
//GUI retrieves the XML schema from the structure of the form
IField f=FieldsManager.GetInstance(schema);
```

Invocazione dei metodi di rendering dell'istanza che rappresenta l'elemento ==> Restituzione del codice Javascript da inserire all'interno della pagina da presentare all'utente

```
//Not needed!!, ASP.Net and WebControl take care of everything, simply do
((WebControl)f) //now you can use it as a WebControl, with standard method calls
```

-> Set an instance value

Chiamata di un metodo sull'istanza fornita per impostare il valore a partire dalla struttura ==> viene inserita nell'istanza il valore richiesto

```
XMLNode xmln;
f.Value=xmln;
```

-> Element validation request

Chiamata della funzione della validazione ==> Viene effettuato un controllo sui valori settati all'interno dell'istanza e sui vincoli imposti

```
bool flag=((IField)f).IsValid();
```

------This was the initial version of the document, it may be subject to changes, especially after some more feature trading with other groups----