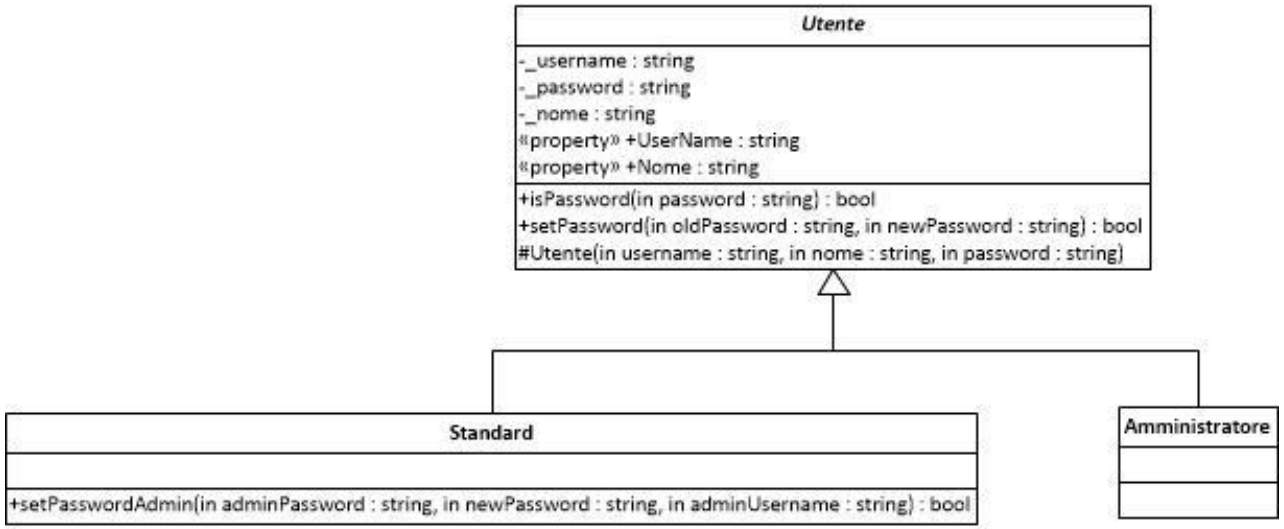
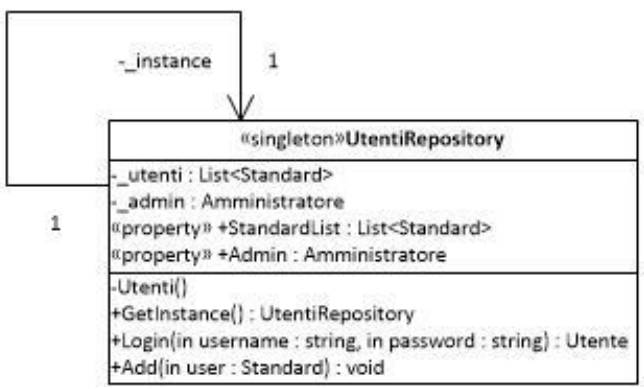
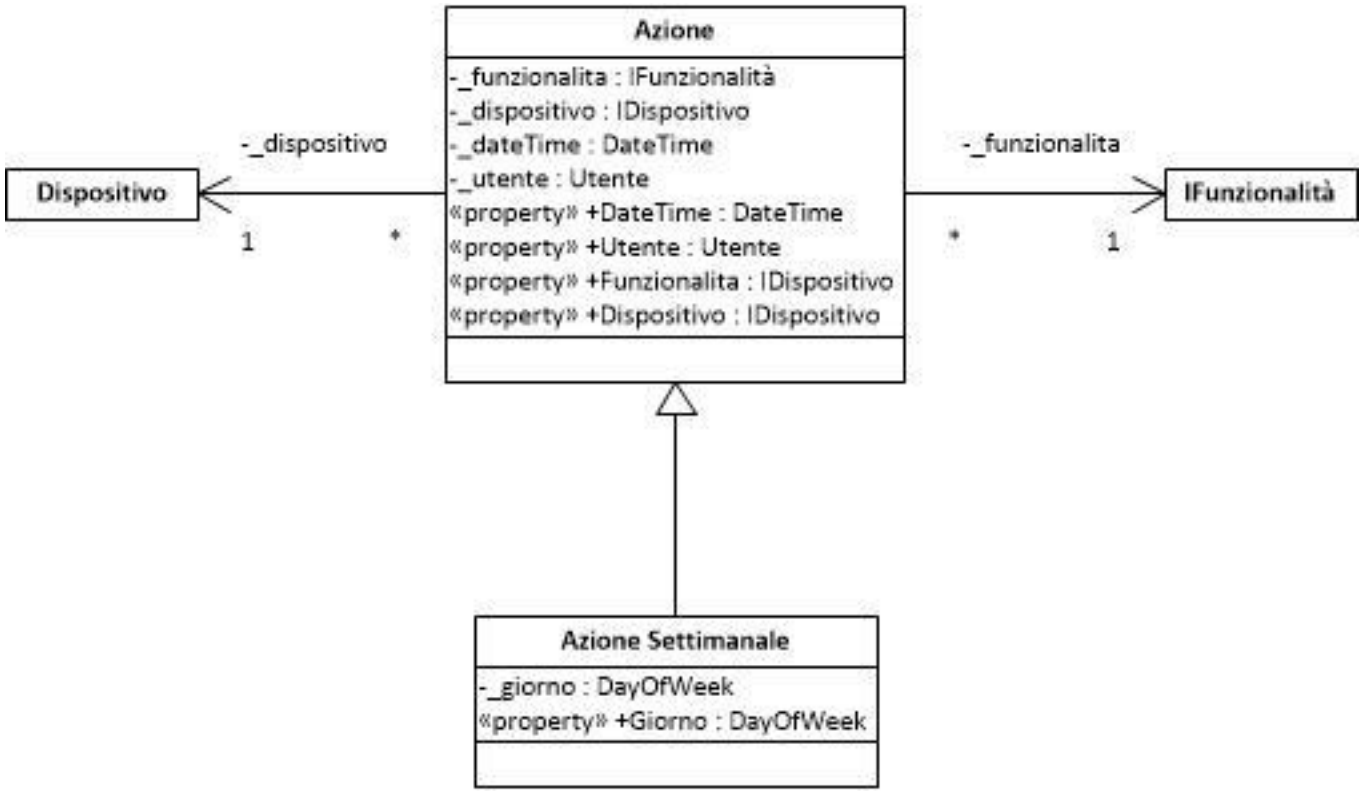
Progettazione

Utenti

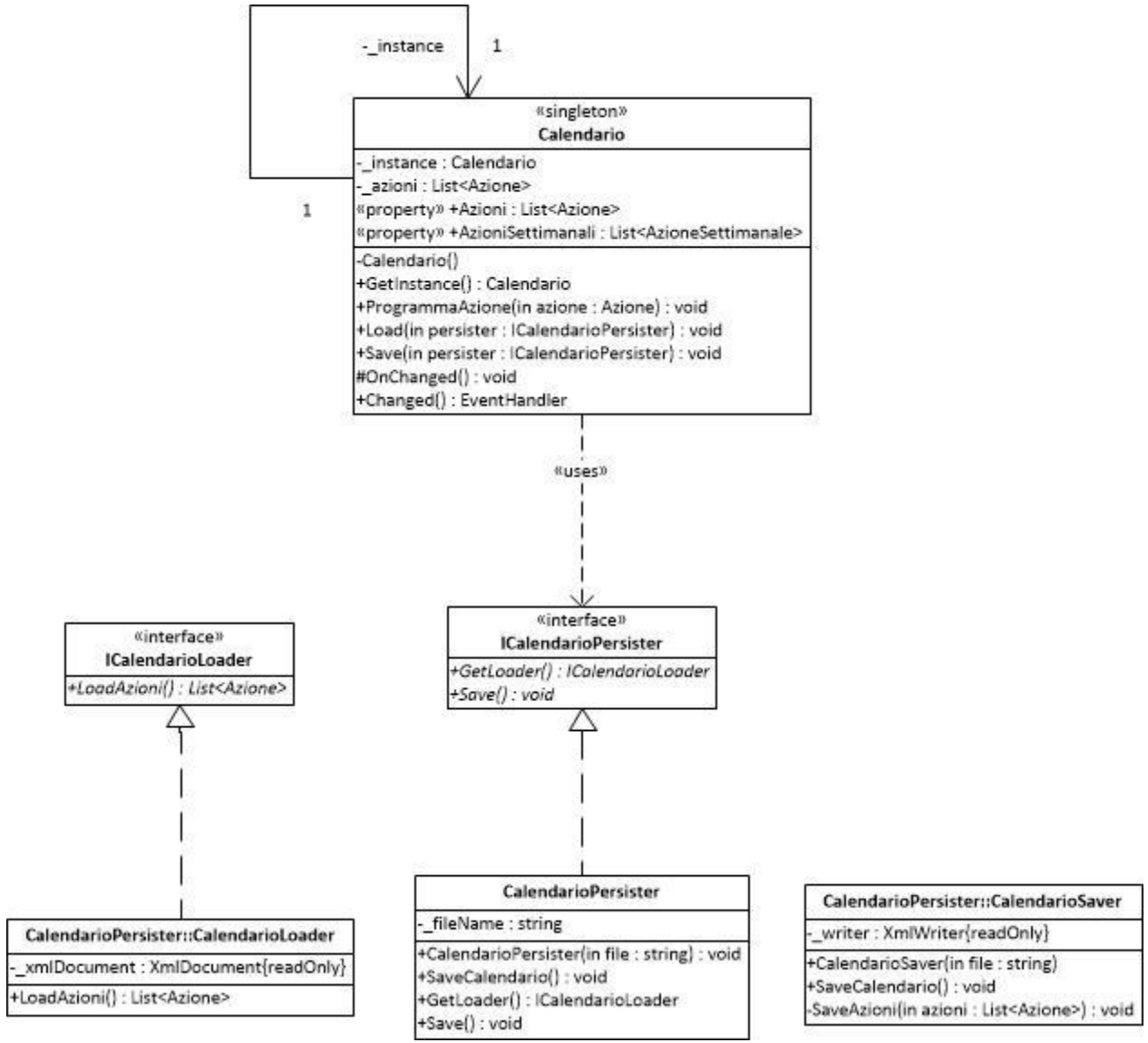

Utente
-_username : string
-_password : string
-_nome : string
«property» +UserName : string
«property» +Nome : string
+isPassword(in password : string) : bool
+setPassword(in oldPassword : string, in newPassword : string) : bool
#Utente(in username : string, in nome : string, in password : string)
Standard
+setPasswordAdmin(in adminPassword : string, in newPassword : string, in adminUsername : string) : bool
Amministratore


-_instance
1
1
«singleton»UtentiRepository
-_utenti : List<Standard>
-_admin : Amministratore
«property» +StandardList : List<Standard>
«property» +Admin : Amministratore
-Utenti()
+GetInstance() : UtentiRepository
+Login(in username : string, in password : string) : Utente
+Add(in user : Standard) : void

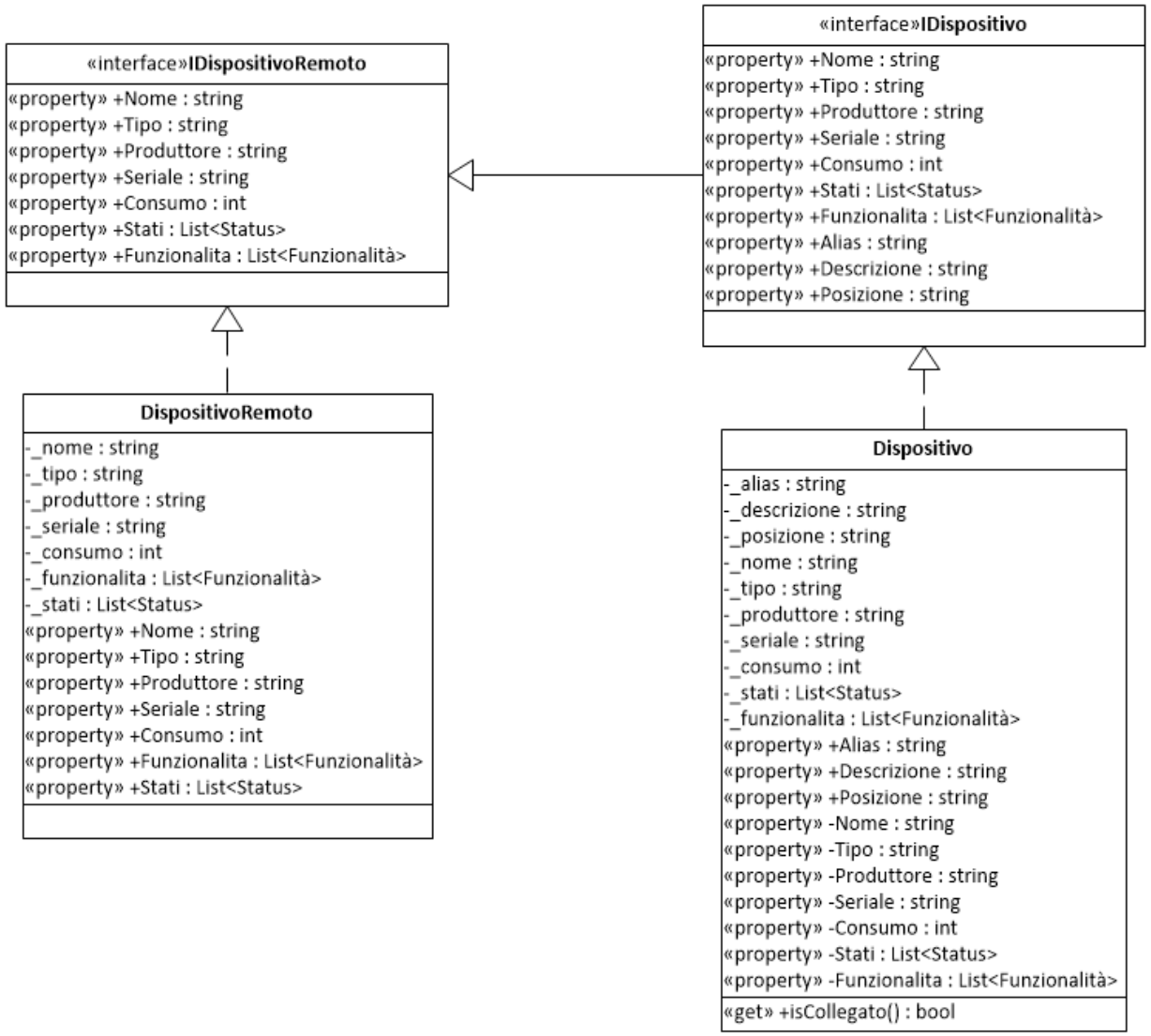
Azione


Azione
-_funzionalita : IFunzionalità
-_dispositivo : IDispositivo
-_dateTime : DateTime
-_utente : Utente
«property» +DateTime : DateTime
«property» +Utente : Utente
«property» +Funzionalita : IDispositivo
«property» +Dispositivo : IDispositivo
-_dispositivo
Dispositivo
1
*
-_funzionalita
IFunzionalità
*
1
Azione Settimanale
-_giorno : DayOfWeek
«property» +Giorno : DayOfWeek

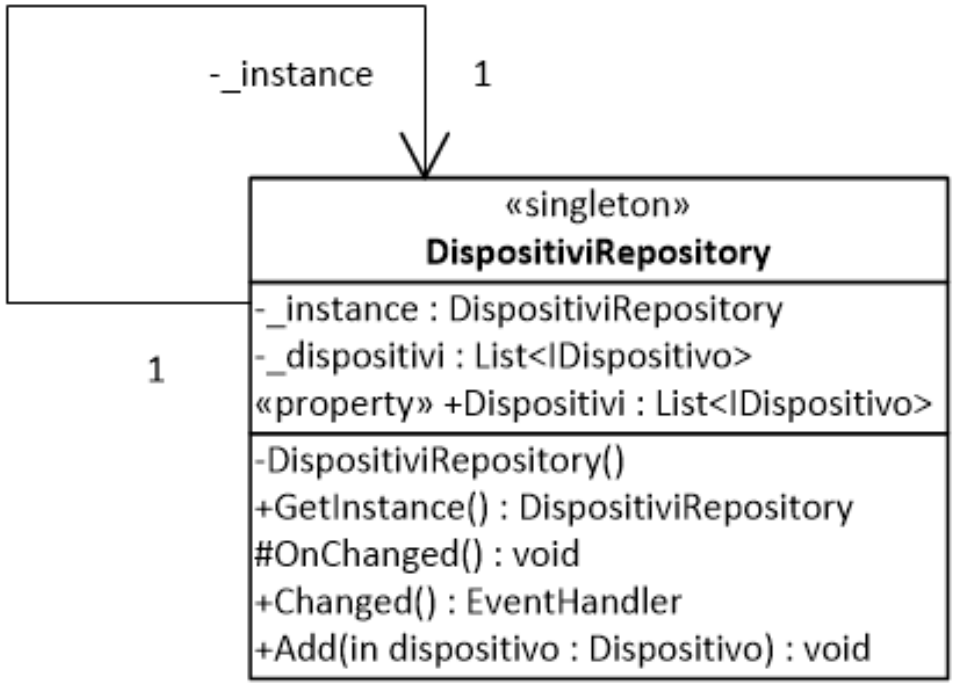
Calendario


-_instance
1
1
«singleton»
Calendario
-_instance : Calendario
-_azioni : List<Azione>
«property» +Azioni : List<Azione>
«property» +AzioniSettimanali : List<AzioneSettimanale>
-Calendario()
+GetInstance() : Calendario
+ProgrammaAzione(in azione : Azione) : void
+Load(in persister : ICalendarioPersister) : void
+Save(in persister : ICalendarioPersister) : void
#OnChanged() : void
+Changed() : EventHandler
«uses»
«interface»
ICalendarioLoader
+LoadAzioni() : List<Azione>
«interface»
ICalendarioPersister
+GetLoader() : ICalendarioLoader
+Save() : void
CalendarioPersister::CalendarioLoader
-_xmlDocument : XmlDocument{readOnly}
+LoadAzioni() : List<Azione>
CalendarioPersister
-_fileName : string
+CalendarioPersister(in file : string) : void
+SaveCalendario() : void
+GetLoader() : ICalendarioLoader
+Save() : void
CalendarioPersister::CalendarioSaver
-_writer : XmlWriter{readOnly}
+CalendarioSaver(in file : string)
+SaveCalendario() : void
-SaveAzioni(in azioni : List<Azione>) : void

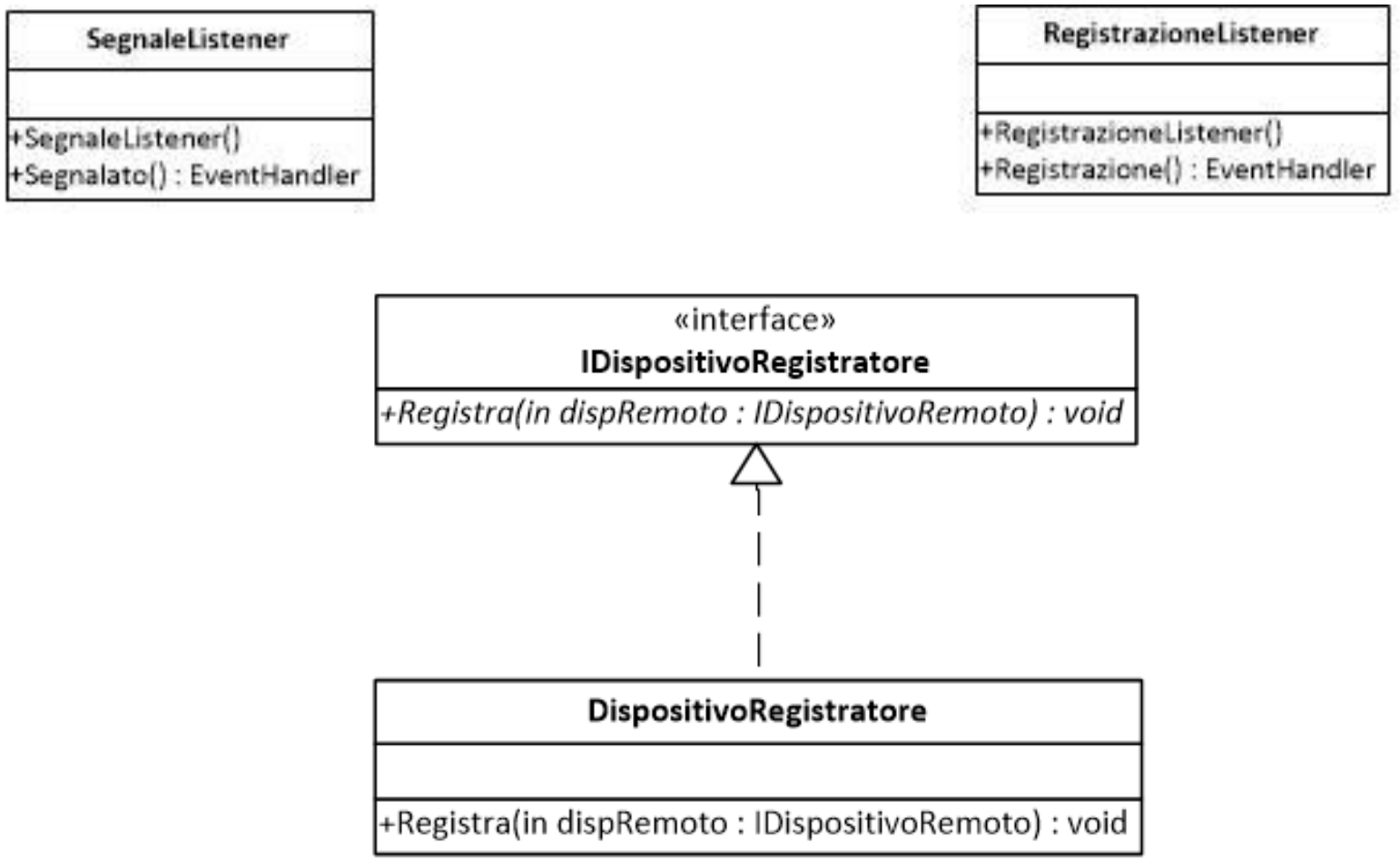
Dispositivi



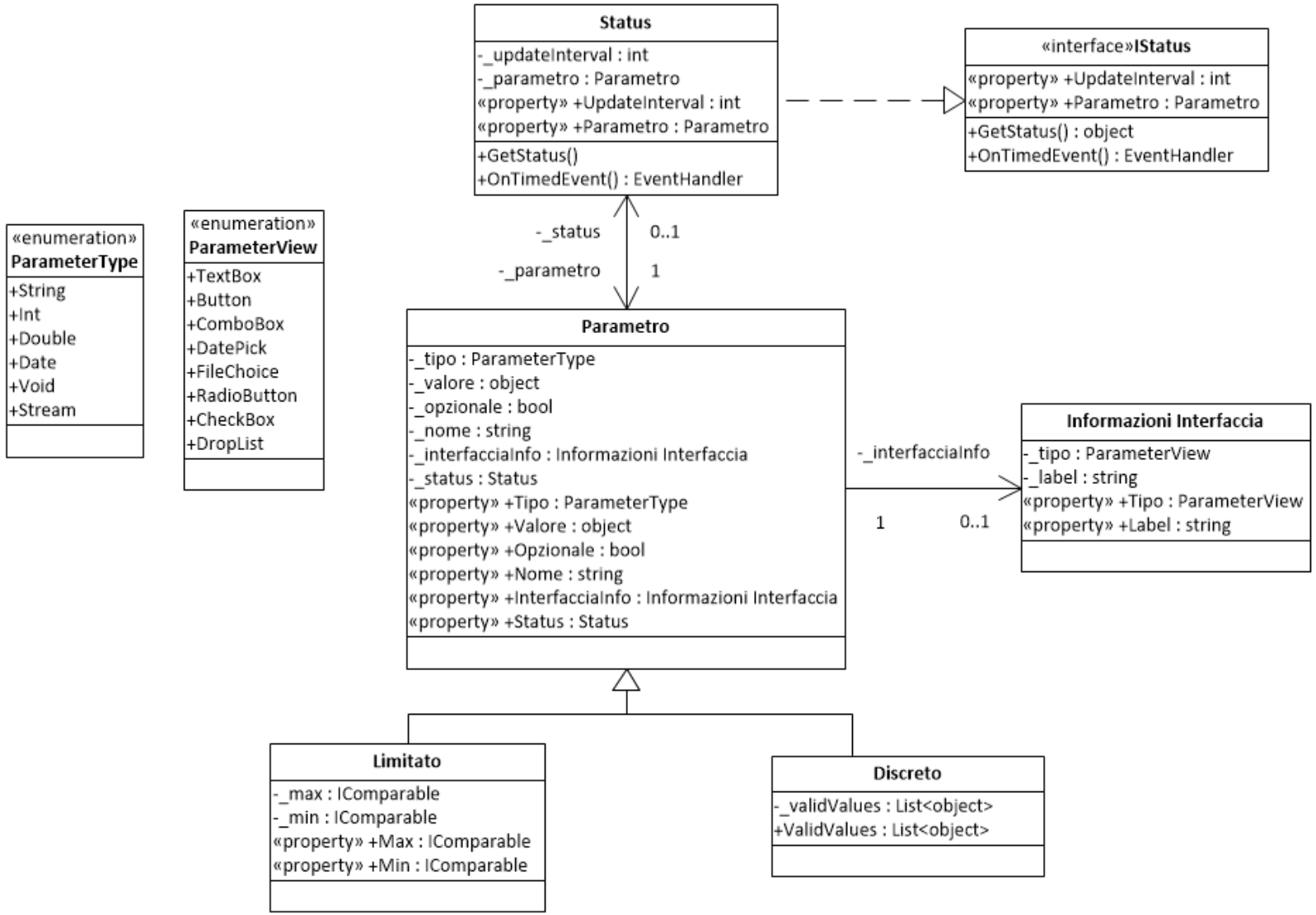
Dispositivi



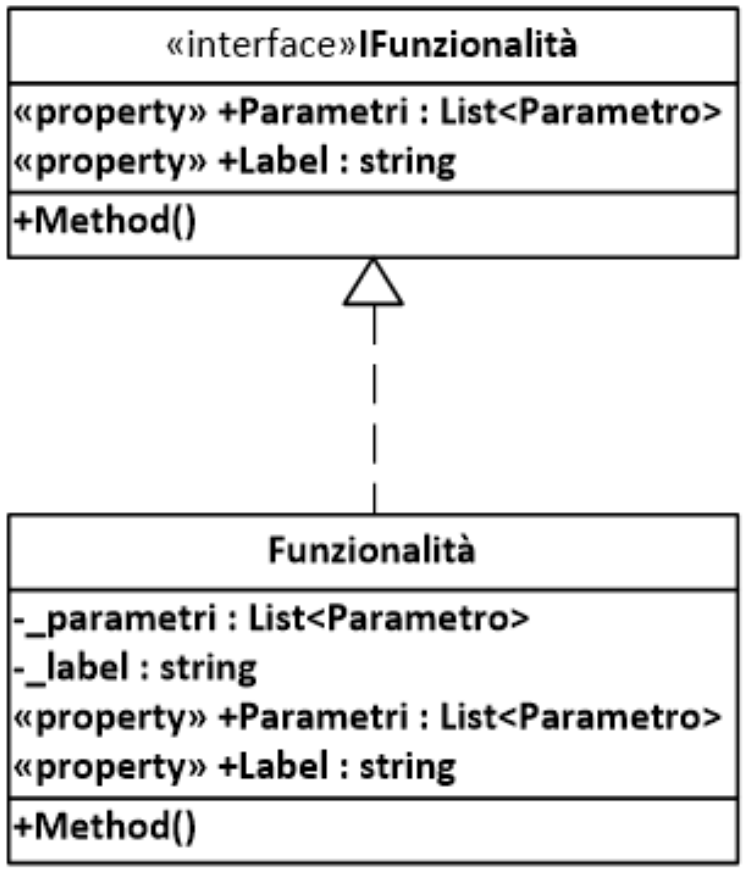
Dispositivi



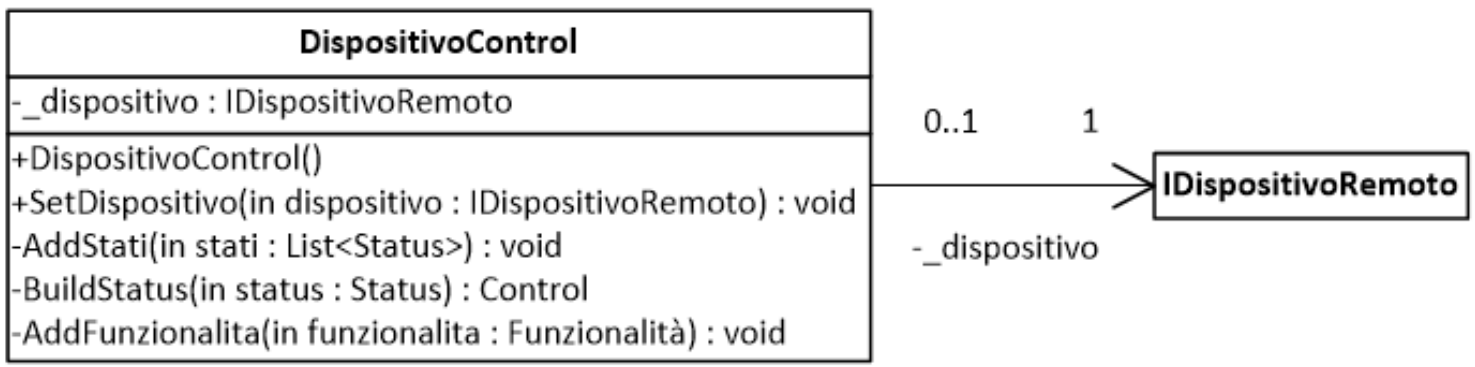
Parametri


Status
-_updateInterval : int
-_parametro : Parametro
«property» +UpdateInterval : int
«property» +Parametro : Parametro
+GetStatus()
+OnTimedEvent() : EventHandler
«interface»IStatus
«property» +UpdateInterval : int
«property» +Parametro : Parametro
+GetStatus() : object
+OnTimedEvent() : EventHandler
-_status
0..1
-_parametro
1
«enumeration»
ParameterType
+String
+Int
+Double
+Date
+Void
+Stream
«enumeration»
ParameterView
+TextBox
+Button
+ComboBox
+DatePick
+FileChoice
+RadioButton
+CheckBox
+DropList
Parametro
-_tipo : ParameterType
-_valore : object
-_opzionale : bool
-_nome : string
-_interfacciaInfo : Informazioni Interfaccia
-_status : Status
«property» +Tipo : ParameterType
«property» +Valore : object
«property» +Opzionale : bool
«property» +Nome : string
«property» +InterfacciaInfo : Informazioni Interfaccia
«property» +Status : Status
-_interfacciaInfo
1
0..1
Informazioni Interfaccia
-_tipo : ParameterView
-_label : string
«property» +Tipo : ParameterView
«property» +Label : string
Limitato
-_max : IComparable
-_min : IComparable
«property» +Max : IComparable
«property» +Min : IComparable
Discreto
-_validValues : List<object>
+ValidValues : List<object>

Parametri


«interface»IFunzionalità
«property» +Parametri : List<Parametro>
«property» +Label : string
+Method()
Funzionalità
-_parametri : List<Parametro>
-_label : string
«property» +Parametri : List<Parametro>
«property» +Label : string
+Method()

Parametri


DispositivoControl
-_dispositivo : IDispositivoRemoto
+DispositivoControl()
+SetDispositivo(in dispositivo : IDispositivoRemoto) : void
-AddStati(in stati : List<Status>) : void
-BuildStatus(in status : Status) : Control
-AddFunzionalita(in funzionalita : Funzionalità) : void
0..1
1
IDispositivoRemoto
-_dispositivo

Principi di Progettazione

In progettazione per molte delle classi del sistema si è applicato il principio di singola responsabilità, ad esempio separando la classe che carica il Calendario dal singleton che ne conserva le informazioni a runtime: per tale classe inoltre vale il principio **ISP**, o di segregazione delle interfacce, essendo infatti divise tra loro quelle responsabili per caricamento e salvataggio delle informazioni. Si è cercato inoltre di astrarre il più possibile dall'implementazione dei dispositivi, per eliminare le dipendenze dai loro meccanismi e rendere la comunicazione il più possibile vicina alle dinamiche tipiche di **RPC** e **RMI**.

Pattern di Progettazione

Oltre al pattern **Singleton**, utilizzato per gestire i vari repository del sistema (utenti, dispositivi, calendario), e il **Visitor**, utilizzato per caricamento e salvataggio del Calendario, per registrazioni e segnali è stato scelto, per via della natura degli eventi stessi, il pattern **Observer**. Una classe è infatti in attesa di richieste di registrazione, un'altra in attesa di segnali da dispositivi: quando rilevano tali eventi interpretandoli secondo la logica della comunicazione tra server e dispositivi, scatenano eventi veri e propri invocando i delegati in ascolto.

Si è cercato di separare il più possibile le informazioni sulla vista (logica) dalla vista effettiva (implementazione), per rispettare la separazione del model dalla view dell'applicazione. Il pattern **MVP** è rispettato in quanto spetta al sistema, più precisamente nella logica di applicazione(Controller), la decisione finale sulla view.