



RESCUEME

SISTEMA GLOBALE PER LA GESTIONE DELLE CATASTROFI NATURALI

SPECIFICA TECNICA

Ingegneria del Software
A.A. 2010/2011

INFORMAZIONI DOCUMENTO

Titolo documento:	Specifica Tecnica
Data creazione:	2011/01/16
Versione attuale:	2.0.0
Stato del documento:	Formale a uso esterno
Nome file:	Specifica_tecnica_2.0.0.pdf
Redazione:	Luca Fongaro Stefano Tonello Alice Zerbaro
Revisione:	Andrea Rampado
Approvazione:	Anna Marin
Distribuito da:	Zeta Solutions
Lista di distribuzione:	Prof. Tullio Vardanega Dott. Riccardo Cardin Zeta Solutions

REGISTRO MODIFICHE

DATA	VERSIONE	REDATTORI	MOTIVO DELLA MODIFICA
2011/02/21	2.0.0	Anna Marin	APPROVAZIONE. Approvazione del documento, cambio di stato in "Formale a uso esterno" e avanzamento di versione.
2011/02/21	1.3.1	Luca Fongaro	REVISIONE. Revisione a seguito delle segnalazioni effettuate dal verificatore Andrea Rampado in data 2011/02/19.
2011/02/19	1.3.0	Stefano Tonello Alice Zerbaro	REVISIONE. Revisione a seguito delle segnalazioni effettuate dal verificatore Andrea Rampado in data 2011/02/18.
2011/02/17	1.2.0	Stefano Tonello	AGGIUNTA CONTENUTI. Modificato il capitolo dei diagrammi delle attività rendendolo più specifico.
2011/02/15	1.1.0	Alice Zerbaro	AGGIUNTA CONTENUTI. Aggiunta descrizione diagrammi View e Controller, rivista numerazione capitolo 5.
2011/02/14	1.0.1	Luca Fongaro	CORREZIONE. Correzione a seguito delle segnalazioni effettuate in sede di RP dal prof. Tullio Vardanega in data 2011/02/07.
2011/02/01	1.0.0	Stefano Tonello	APPROVAZIONE. Approvazione del documento, cambio di stato in "Formale a uso esterno" e avanzamento di versione.
2011/01/31	0.6.1	Luca Fongaro Massimo Lana	REVISIONE. Revisione a seguito delle segnalazioni effettuate dal verificatore Alice Zerbaro in data 2011/01/29.
2011/01/21	0.6.0	Gianluca Carlesso Anna Marin Andrea Rampado	AGGIUNTA CONTENUTI. Aggiunta capitoli "Stime di Fattibilità" e "Tracciamento della relazione componenti - requisiti".
2011/01/20	0.5.1	Gianluca Carlesso Luca Fongaro	CORREZIONE. Reimpostazione generale dei capitoli e correzione errori ortografici e sintattici.
2011/01/20	0.5.0	Massimo Lana Anna Marin	AGGIUNTA CONTENUTI. aggiunta capitolo "Interfacce Utenti"
2011/01/19	0.4.0	Gianluca Carlesso Luca Fongaro Andrea Rampado	AGGIUNTA CONTENUTI. Completato capitolo "Descrizione dei singoli componenti" e aggiunto capitolo "Design Pattern".
2011/01/18	0.3.1	Gianluca Carlesso	CORREZIONE. Correzione di vari errori ortografici e sintattici.
2011/01/18	0.3.0	Massimo Lana Anna Marin Andrea Rampado	AGGIUNTA CONTENUTI. Inizio stesura capitolo "Descrizione dei singoli componenti".
2011/01/17	0.2.0	Luca Fongaro Massimo Lana Andrea Rampado	AGGIUNTA CONTENUTI. Aggiunti capitoli "Diagramma dei Package" e "Diagramma di Attività".
2011/01/16	0.1.0	Gianluca Carlesso Anna Marin	PRIMA STESURA. Creazione documento e stesura capitoli "Introduzione" e "Definizione del prodotto".

STORICO

RR->RP

Versione 1.0.0	Nominativo
Redazione	Gianluca Carlesso, Luca Fongaro, Massimo Lana, Anna Marin, Andrea Rampado
Verifica	Alice Zerbaro
Approvazione	Stefano Tonello

RP->RQ

Versione 2.0.0	Nominativo
Redazione	Luca Fongaro, Stefano Tonello, Alice Zerbaro
Verifica	Andrea Rampado
Approvazione	Anna Marin

Sommario

Il presente documento ha l'intento di fornire una descrizione architetturale del sistema software RescueMe. Attraverso un'insieme di diagrammi UML, ottenuti a partire dai requisiti, verranno descritte le attività, i componenti e le loro interazioni. Partendo da una visione complessiva del sistema si andrà progressivamente nel dettaglio.

Indice

1	Introduzione	1
1.1	Scopo del documento	1
1.2	Scopo del prodotto	1
1.3	Glossario	1
1.4	Riferimenti	1
1.4.1	Normativi	1
1.4.2	Informativi	2
2	Descrizione delle tecnologie utilizzate	3
2.1	Google App Engine	3
2.2	Spring MVC	4
2.3	Spring Security	4
2.4	Persistenza dei dati	4
3	Definizione del prodotto	5
3.1	Metodo e formalismo di specifica	5
3.2	Presentazione dell'architettura generale del sistema e identificazione dei componenti architetturali di alto livello	5
3.2.1	Architettura alto livello lato server	5
3.2.2	Architettura alto livello lato client	6
3.3	Diagramma dei package	7
3.3.1	Struttura dei package	7
3.3.1.1	Diagramma dei package applicazione Android	8
4	Diagrammi di attività	10
4.1	Segnalante	10
4.1.1	Segnalante Web	10
4.1.1.1	Registrazione	10
4.1.1.2	Segnalazione anonima	10
4.1.1.3	Login	10
4.1.2	Segnalante Android	11
4.1.2.1	Invia segnalazione	11
4.1.2.2	Visualizza stato segnalazione	11
4.2	Amministratore	12
4.2.1	Amministratore d'autorità	12
4.2.1.1	Autenticazione	12
4.2.1.2	Gestione catastrofe	12
4.2.1.3	Gestione operatore	13
4.2.1.4	Gestione richieste	13
4.2.2	Amministratore di sistema	13
4.2.2.1	Autenticazione	13
4.2.2.2	Registra nuovo utente di tipo autorità	13
4.2.2.3	Crea amministratore	14
4.2.2.4	Gestione autorità	14
4.3	Operatore	14
4.3.1	Autenticazione	14

4.3.2	Gestione richieste	14
4.4	Sottoattività	15
4.4.1	Autenticazione	15
4.4.2	Crea nuovo amministratore	15
4.4.3	Gestione Autorità	16
4.4.3.1	Associa operatore	16
4.4.3.2	Modifica dati	16
4.4.4	Gestione Catastrofe	17
4.4.5	Gestione Operatore	17
4.4.6	Gestione Richieste	18
4.4.7	Gestione Segnalazione	18
4.4.8	Invia Segnalazione	19
4.4.9	Modifica dati utente	19
4.4.10	Registrazione	20
5	Descrizione dei singoli componenti	21
5.1	Model	21
5.1.1	Descrizione del diagramma	21
5.1.2	Presentazione delle classi	21
5.1.2.1	Address	21
5.1.2.2	GenericUser	22
5.1.2.3	Authority	22
5.1.2.4	OperatorDisaster	22
5.1.2.5	Disaster	22
5.1.2.6	Request	23
5.1.2.7	Comment	23
5.1.2.8	PersistenceMgr	23
5.1.2.9	HistoricalChanges	23
5.1.2.10	OperatorAuthority	23
5.1.2.11	AuthorityDisaster	24
5.2	View	24
5.2.1	Descrizione del diagramma	24
5.2.2	Presentazione delle classi	25
5.2.2.1	RescueMeShell	25
5.2.2.2	MainMenuTreeViewModel	25
5.2.2.3	ContentWidget	25
5.2.2.4	Widget	25
5.3	Controller	26
5.3.1	Descrizione del diagramma	26
5.3.2	Presentazione delle classi	26
5.3.2.1	ControllerUtilities	26
5.3.2.2	IAuthService	27
5.3.2.3	IAuthServiceAsync	27
5.3.2.4	AuthServiceImpl	27
5.3.2.5	IWebService	27
5.3.2.6	IWebServiceAsync	28
5.3.2.7	WebServiceImpl	28
5.3.2.8	IValidatorService	28
5.3.2.9	IValidatorServiceAsync	28
5.3.2.10	ValidatorServiceImpl	29
5.3.2.11	IStatistics	29
5.3.2.12	IStatisticsAsync	29
5.3.2.13	Statistic	29
5.3.2.14	CommentController	29

5.3.2.15	DisasterController	30
5.3.2.16	GenericUserController	30
5.3.2.17	RequestController	30
5.3.2.18	CustomAuthenticationProvider	30
5.3.2.19	CustomAuthListener	31
5.3.2.20	CustomUserAuthentication	31
5.3.2.21	GwtRpcController	31
6	Applicazione Android	32
6.1	Descrizione dei singoli componenti	32
6.1.1	Model	32
6.1.1.1	Presentazione delle classi	33
6.1.1.1.1	Request	33
6.1.1.1.2	Comment	33
6.1.1.1.3	User	33
6.1.1.1.4	PersistenceMgrAndroid	34
6.1.2	View	34
6.1.2.1	Presentazione delle classi	34
6.1.2.1.1	Registration	34
6.1.2.1.2	NewRequestForm	35
6.1.2.1.3	ViewRequest	35
6.1.2.1.4	ViewListRequests	35
6.1.2.1.5	ModifyInformationUser	36
6.1.3	Controller	36
6.1.3.1	Presentazione delle classi	36
6.1.3.1.1	InsertNewRequest	36
6.1.3.1.2	UpdateRequest	36
6.1.3.1.3	UserRegistration	37
6.2	Comunicazione con il sistema RescueMe	37
6.2.1	Protezione URI API RESTful	37
6.2.2	Persistenza dati	37
7	Design pattern	38
7.1	Singleton	38
7.2	MVC	38
8	Interfacce utenti	40
8.1	GUI1: Pagina principale	40
8.1.1	GUI1.1: Autenticazione nel sistema	40
8.1.2	GUI1.2: Invio richiesta	41
8.1.2.1	GUI1.2.1: Form di inserimento richiesta di soccorso	41
8.1.3	GUI1.3: Form di registrazione	42
8.2	GUI2: Pannello segnalante	43
8.3	GUI3: Pannello amministratore di sistema	43
8.4	GUI4: Pannello operatore, operatore avanzato e amministratore d'autorità	44
8.5	GUI5: Applicazione Android	45
9	Stime di fattibilità	46
10	Tracciamento della relazione componenti - requisiti	47

Elenco delle figure

1	Google App Engine	3
2	Processo di elaborazione di una richiesta con Spring MVC	4
3	DataStore di Google App Engine	5
4	Schema generale applicazione Client - Server	6
5	Schema generale interazione client-server dell'applicazione	6
6	Specifica dei dispositivi con cui può interagire il segnalante	7
7	DP1 - Diagramma dei package e loro relazioni	8
8	DP1.1 - Diagramma dei package applicazione Android	9
9	DA1 - Diagramma di attività del segnalante per l'interfaccia Web	11
10	DA2 - Diagramma di attività del segnalante per l'applicazione Android	12
11	DA3 - Diagramma di attività dell'amministratore di autorità	13
12	DA4 - Diagramma di attività dell'amministratore di sistema	14
13	DA5 - Diagramma di attività dell'operatore	15
14	DA6 - Diagramma di attività per l'autenticazione	15
15	DA7 - Diagramma di attività della creazione di un nuovo amministratore	16
16	DA8 - Diagramma di attività della gestione di un'autorità	17
17	DA9 - Diagramma di attività della gestione catastrofe	17
18	DA10 - Diagramma di attività della gestione operatore	18
19	DA11 - Diagramma di attività della gestione richieste	18
20	DA12 - Diagramma di attività della segnalazione	18
21	DA13 - Diagramma di attività dell'invio della segnalazione	19
22	DA14 - Diagramma di attività della modifica dei dati di un utente	20
23	DA15 - Diagramma di attività della registrazione di un utente	20
24	DC1 - Diagramma delle classi che illustra il Model	21
25	DC2 - Diagramma delle classi che illustra la View	24
26	DC3 - Diagramma delle classi che illustra il Controller	26
27	DC4 - Diagramma delle classi per applicazione Android	32
28	Design Pattern Singleton in UML _[g]	38
29	Pattern architetturale MVC	39
30	GUI1 - Home Page di RescueMe	40
31	GUI1.1 - Pagina Autenticazione	41
32	GUI1.2 - Pagina per l'invio della richiesta, primo passo	41
33	GUI1.2.1 - Pagina per l'invio della richiesta, secondo passo	42
34	GUI1.3 - Pagina per la registrazione	42
35	GUI2 - Interfaccia web segnalante	43
36	GUI3 - Interfaccia web amministratore di sistema	44
37	GUI4 - Interfaccia web operatore, operatore avanzato e amministratore d'autorità	44
38	GUI5 - Interfaccia per l'applicazione smartphone	45

Elenco delle tabelle

1	Tracciamento package - componenti	48
2	Raggruppamento componenti atomici	48
3	Tracciamento Componenti - Requisiti	49
4	Tracciamento Requisiti - Componenti	51

1 Introduzione

1.1 Scopo del documento

Il presente documento ha lo scopo di definire l'architettura del prodotto software RescueMe. Tale definizione inizia descrivendo il prodotto ad alto livello, dopo la quale segue un'analisi a basso livello, applicando quindi un approccio top-down_[g].

1.2 Scopo del prodotto

Il prodotto ha lo scopo di realizzare un sistema di gestione delle catastrofi, sviluppato all'interno dell'infrastruttura di *cloud computing*_[g] fornita da Google e denominata *Google App Engine*_[g], d'ora in avanti *GAE*. Con gestione delle catastrofi è intesa l'attività di richiesta di aiuto da parte di persone colpite dalle catastrofi, il monitoraggio della situazione globale riguardante gli interventi di salvataggio eseguiti, di quelli rifiutati, di quelli ancora in attesa e infine l'organizzazione delle risorse per far fronte nel miglior modo possibile alle richieste d'aiuto da parte della popolazione colpita. Essendo le tecnologie proposte dal capitolato in oggetto sconosciute alla totalità dei membri del gruppo, Zeta Solutions vuole cogliere l'occasione per esplorare e conoscere queste nuove soluzioni tecnologiche. I prodotti sviluppati utilizzando questo tipo di infrastruttura presentano, inoltre, alcuni evidenti vantaggi:

- Scalabilità automatica dell'applicazione;
- Creazione di interfacce Web_[g] scrivendo un normale programma *Java*_[g];
- *Framework*_[g] di sviluppo *opensource*_[g];
- Utilizzo di un database distribuito.

1.3 Glossario

Tutti gli acronimi, le abbreviazioni e i termini ambigui o di uso non comune menzionati in questo documento vengono definiti nel documento esterno Glossario 2.0.0 che si allega alla presente documentazione. Tali termini saranno evidenziati solo alla prima occorrenza e verrà utilizzata la seguente nomenclatura: parola_[g].

1.4 Riferimenti

1.4.1 Normativi

- Capitolato d'appalto del committente all'indirizzo
<http://www.math.unipd.it/~tullio/IS-1/2010/Progetto/RescueMe.pdf>;
- Analisi dei requisiti 2.0.0;
- Norme di progetto 2.0.0.

1.4.2 Informativi

- Registro comunicazioni proponente 1.0.0;
- Registro comunicazioni committente 1.0.0;
- Piano di progetto 2.0.0;
- Piano di qualifica 2.0.0;
- Glossario 2.0.0;
- Slides del corso di Ingegneria del Software (a.a. 2010/2011) presenti all'indirizzo <http://www.math.unipd.it/~tullio/IS-1/2010>.
- Swebok - Guide to the Software Engineering Body of Knowledge <http://www.swebok.org>.
- Testo di consultazione

Software Engineering (8th edition)
Ian Sommerville
Pearson Education, Addison-Wesley

- Testo di consultazione

Design Patterns - Elements of Reusable Object-Oriented Software
GoF, 1995, Addison-Wesley

- Documentazione riguardante le tecnologie da utilizzare:
 - piattaforma GAE:
<http://code.google.com/appengine/articles/>.
 - tecnologia DataStore_[g]:
<http://code.google.com/appengine/docs/java/datastore>.
 - piattaforma Google Web Toolkit_[g], d'ora in poi indicato solo come GWT:
<http://code.google.com/intl/it/webtoolkit/doc/latest/DevGuide.html>.
 - Android_[g]SDK_[g]:
<http://developer.android.com/index.html>.
 - Android Designing for Performance:
<http://developer.android.com/guide/practices/design/performance.html>.
 - Android Best Practises:
<http://developer.android.com/guide/webapps/best-practices.html>.
- Design pattern Singleton:
http://sourcemaking.com/design_patterns/singleton.

2 Descrizione delle tecnologie utilizzate

2.1 Google App Engine

L'applicazione RescueMe risiede nell'infrastruttura cloud GAE, essa verrà così eseguita in un ambiente, detto Sandbox_[g] (vedi figura 1 a pagina 3), che permette di distribuire le richieste su più server e di avviare e fermare tali server per soddisfare le esigenze di traffico. La Sandbox isola l'applicazione in un ambiente sicuro e affidabile che è indipendente dall'hardware, dal sistema operativo e dalla posizione fisica del server Web.

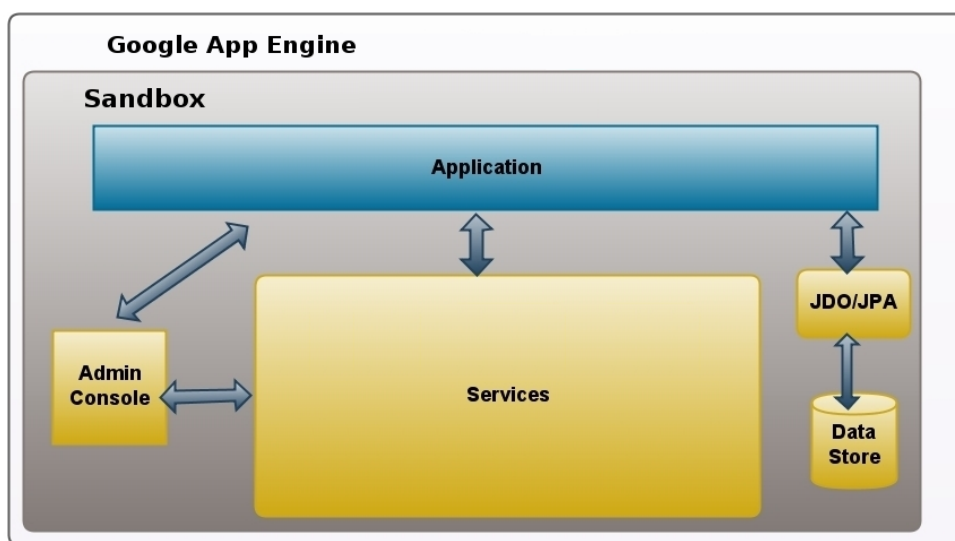


Figura 1: Google App Engine

In particolare l'architettura GAE offre le seguenti caratteristiche:

Scalabilità automatica: le risorse disponibili all'esecuzione dell'applicazione aumentano linearmente con l'aumentare delle richieste al fine di garantire performance costanti;

Performance: il tempo di elaborazione delle richieste ricevute rimangono costanti al crescere del carico di lavoro.

La comunicazione tra client e server avviene attraverso il servizio Remote Procedure Call_[g], d'ora in avanti RPC, il quale fornisce un meccanismo di chiamata a procedura remota la cui semantica è essenzialmente equivalente a quella della chiamata a una procedura locale. Il client possiede un'interfaccia_[g] dei metodi implementati nel server e quando necessario può invocarli da remoto. Oltre all'implementazione dei metodi dichiarati nell'interfaccia fornita al client, nel server sono presenti le servlet_[g], tali classi, nel nostro caso, servono per generare pagine Web in forma dinamica a seconda dei parametri della richiesta spedita tramite browser.

2.2 Spring MVC

Il prodotto software RescueMe utilizzerà il framework Spring MVC_[g], che permette di gestire le comunicazioni tra i client e l'applicazione centrale. Esso è basato sul design pattern MVC_[g] con l'aggiunta del componente *Front controller*, il quale è realizzato attraverso una servlet_[g] centrale che ha il compito di inviare le richieste al Controller.

Ogni richiesta viene così elaborata (vedi figura 2 a pagina 4):

1. Il Front controller la riceve e la inoltra al Controller;
2. Il Controller la elabora, crea dei componenti di Model e la invia al Front controller;
3. Il Front controller la inoltra alla View che effettua il rendering della vista;
4. Infine il controllo ritorna al Front controller che invia la risposta al client.

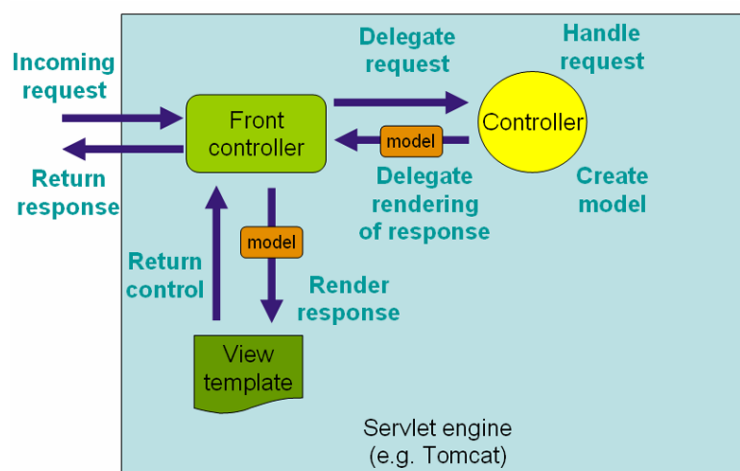


Figura 2: Processo di elaborazione di una richiesta con Spring MVC

2.3 Spring Security

L'azienda Zeta Solutions utilizzerà il framework Spring Security_[g] per la gestione della sicurezza. Esso fornisce soluzioni di sicurezza per le applicazioni basate su Spring sfruttando la tecnica chiamata dependency injection_[g], inoltre permette di gestire l'autenticazione e l'autorizzazione controllando se l'utente ha i permessi per accedere alle risorse Web dell'applicazione e, in maniera più fine, permette anche di gestire l'autorizzazione all'invocazione dei metodi.

2.4 Persistenza dei dati

App Engine offre un servizio di archivio dati caratterizzato da un motore di query e transazioni. Il database, denominato DataStore (vedi figura 3 a pagina 5), possiede le seguenti caratteristiche:

- È un database non relazionale;
- I dati inseriti vengono gestiti come degli oggetti che hanno un tipo e un insieme di proprietà. Il tipo dell'oggetto viene definito dall'applicazione stessa;
- Le query possono recuperare gli oggetti secondo dei criteri e ordinarne il risultato in base alle proprietà;

- I valori delle proprietà possono essere di un qualsiasi tipo supportato dal DataStore;
- La connessione dell'applicazione al database utilizza le interfacce Java_[g] JDO_[g] e JPA_[g];
- È fortemente consistente;
- Utilizza il sistema optimistic concurrency control_[g] per la gestione della concorrenza;
- L'applicazione può eseguire più operazioni in una singola transazione: queste avranno successo o falliranno, garantendo l'integrità dei dati.

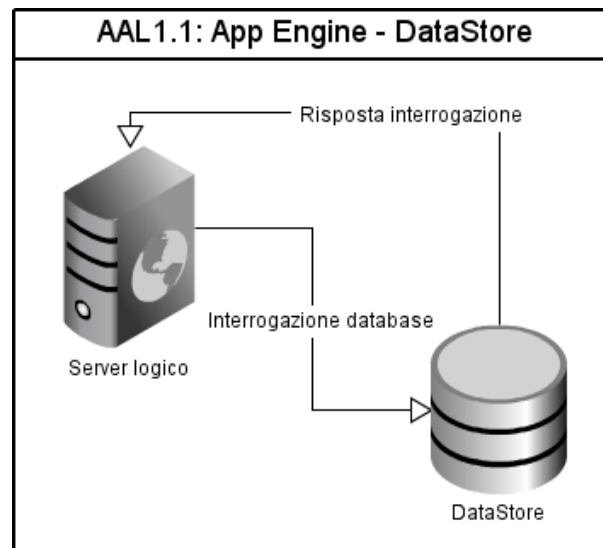


Figura 3: DataStore di Google App Engine

3 Definizione del prodotto

3.1 Metodo e formalismo di specifica

Il formalismo che viene utilizzato nel seguente documento per la specifica è quello fornito dal linguaggio UML_[g], in particolare i diagrammi delle classi. L'approccio utilizzato nella specifica è di tipo top-down_[g] ovvero, partendo da una descrizione generale dell'architettura del sistema, si forniscono delle specifiche dettagliate dei componenti. Il dettaglio della specifica è mirato ad agevolare la fase di codifica del prodotto. Nel seguente documento sono inserite alcune immagini illustrative che non aderiscono ad alcuno standard di presentazione e che hanno il solo scopo di aiutare a comprendere i concetti descritti.

3.2 Presentazione dell'architettura generale del sistema e identificazione dei componenti architetturali di alto livello

3.2.1 Architettura alto livello lato server

L'applicazione sviluppata si basa su un sistema client-server, nel quale un client richiede un servizio utente al sistema RescueMe situato nel server (Vedi figura 4 a pagina 6).

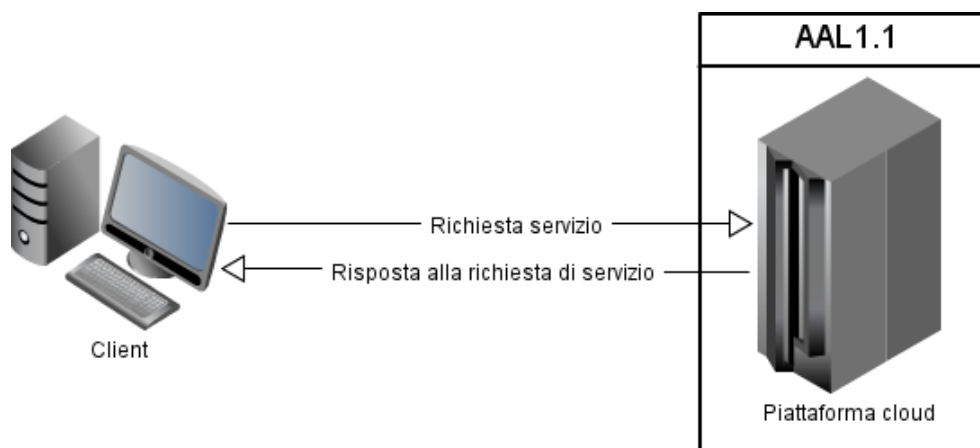


Figura 4: Schema generale applicazione Client - Server

3.2.2 Architettura alto livello lato client

L'utente interagisce con l'applicazione RescueMe attraverso delle GUI_[g] personalizzate a seconda del tipo di utente (vedi figura 5 a pagina 6). L'insieme degli utenti si suddivide principalmente in chi utilizza e chi gestisce l'applicazione.

Per chi utilizza l'applicazione vengono fornite tre GUI a seconda se si tratta di:

- Segnalanti;
- Amministratori;
- Operatori.

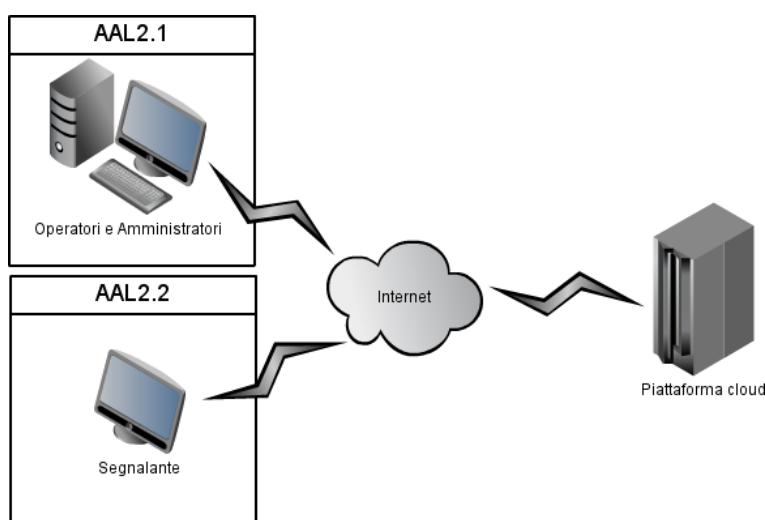


Figura 5: Schema generale interazione client-server dell'applicazione

Inoltre, sempre per chi utilizza l'applicazione si rendono disponibili diverse GUI a seconda del dispositivo con cui l'utente interagisce (vedi figura 6 a pagina 7). Le GUI per dispositivi mobili con sistema operativo diverso da Android e per computer dotati di browser vengono sviluppate con l'ausilio di GWT. Tale framework permette di sviluppare le interfacce delle applicazioni utilizzando il linguaggio Java, inoltre, in fase di compilazione crea differenti prodotti di compilazione funzionanti e ottimizzati per diversi tipi di browser tramite la tecnica deferred binding_[g]. La GUI per dispositivi mobile con sistema operativo Android è sviluppata con il SDK fornita da Google.

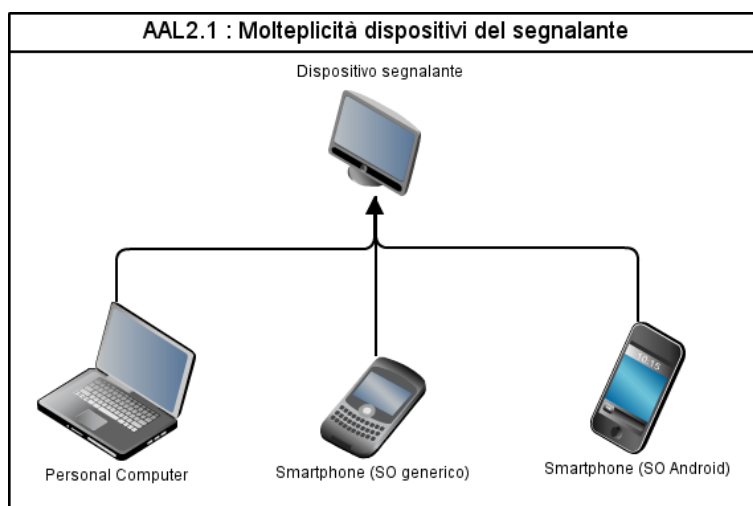


Figura 6: Specifica dei dispositivi con cui può interagire il segnalante

3.3 Diagramma dei package

I *diagrammi dei package*_[g] descrivono i rapporti che intercorrono tra i diversi *package*_[g] che compongono un sistema.

Non esplicitando una relazione di dipendenza, un package non può accedere agli elementi contenuti in un altro package. Le relazioni di dipendenza definiscono l'accessibilità al contenuto di altri package e non sono transitive, cioè se il package A può vedere B e B può vedere C, non è detto che A possa vedere C.

3.3.1 Struttura dei package

Nel prodotto software RescueMe si è individuato il package

it.zetasolutions.rescueme

come package base che contiene tutti i package che compongono il sistema. La struttura dei package rispecchia la struttura logica degli elementi presenti nel design pattern MVC (vedi sezione 7.2 a pagina 38).

All'interno del package principale si possono individuare i seguenti package (vedi figura 7 a pagina 8):

client: contiene le interfacce e le classi accessibili dal client;

view: implementa le interfacce e le classi necessarie alla realizzazione dell'interfaccia grafica.

shared: contiene interfacce e classi di comunicazione tra client_[g] e server_[g];

controller: contiene gli elementi necessari per interfacciarsi con il controller_[g];

model: contiene gli elementi necessari per la gestione di dati lato client.

server: contiene tutte le interfacce e classi accessibili esclusivamente dal server;

controller: implementa le interfacce ed estende le classi necessarie alla realizzazione del controller;

model: istanzia le classi necessarie alla realizzazione del model_[g].

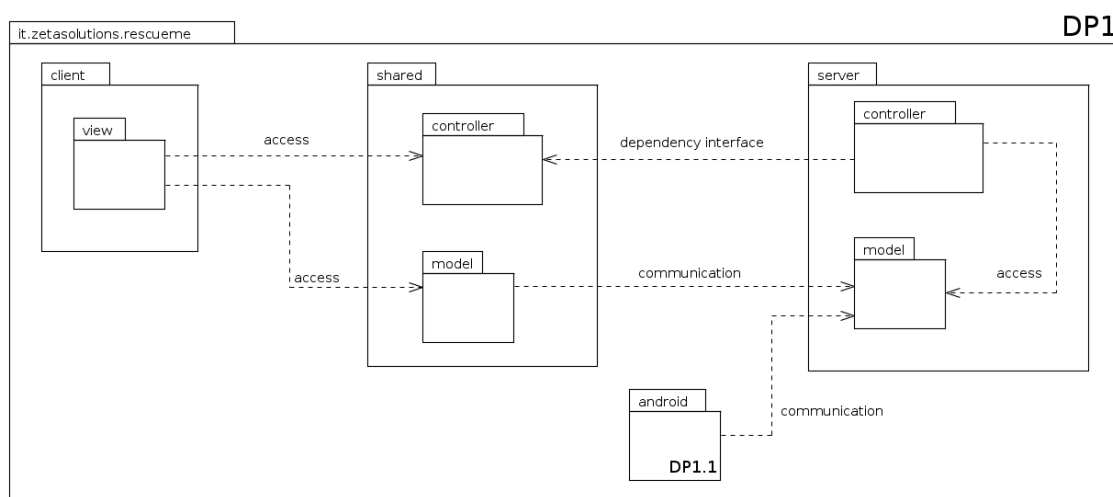


Figura 7: DP1 - Diagramma dei package e loro relazioni

Dependency interface: indica il rapporto tra due package che realizzano il design pattern di interfaccia separata, ovvero quando un package contiene le definizioni delle interfacce e delle classi (package di destinazione della freccia) mentre un altro contiene le implementazioni di queste (package di origine della freccia);

Access: gli elementi contenuti nel package raggiunti dalla freccia possono essere referenziati dagli elementi contenuti nel package dove inizia la freccia.

Communication: i due package in relazione scambiano informazioni tra loro attraverso un protocollo di comunicazione per il trasferimento dei dati gestiti dai due rispettivi package. Il protocollo permette ai due package di gestire i dati internamente in maniera indipendente, ma di comunicare e trasferire dati tra loro attraverso delle regole e convenzioni comuni.

3.3.1.1 Diagramma dei package applicazione Android

L'applicazione per smartphone Android è contenuta nel package

it.zetasolutions.rescueme.android

Essa è suddivisa nei seguenti sotto-package (vedi figura 8 a pagina 9):

controller: contiene tutte le classi necessarie per ricevere le richieste dell'utente trasmesse attraverso la view, ed esegue tutte le operazioni necessarie al fine di soddisfare tali richieste andando a interagire con i componenti di view e model;

view: contiene le classi che realizzano la GUI dell'applicazione;

model: contiene tutti i componenti necessari alla persistenza e gestione dei dati.

L'applicazione realizza il pattern architetturale MVC (vedi sezione 29 a pagina 39).

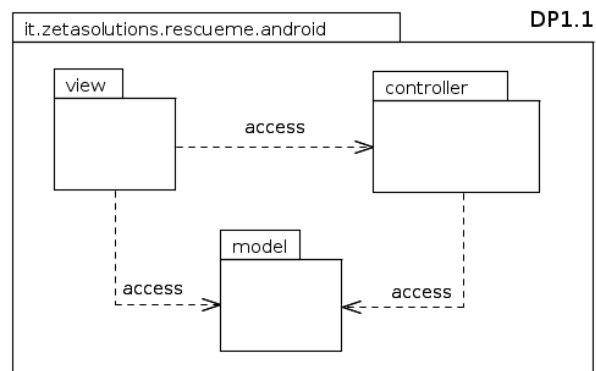


Figura 8: DP1.1 - Diagramma dei package applicazione Android

4 Diagrammi di attività

In questo capitolo saranno presentati i diagrammi di attività per le diverse tipologie di utente. Essi analizzano solamente il flusso principale delle attività occupandosi delle alternative solo nei casi principali.

4.1 Segnalante

4.1.1 Segnalante Web

Il diagramma in figura 9 a pagina 11 rappresenta il modo in cui un segnalante può interagire con il sistema. Anche un segnalante non registrato ha la possibilità di interagire con il sistema inviando una segnalazione anonima, tuttavia non avrà la possibilità di visualizzare lo stato di lavorazione delle richieste stesse.

In questo diagramma viene illustrato come un segnalante che accede al sistema via Web possa compiere dalla finestra principale tre azioni: inviare una segnalazione anonima, registrarsi all'applicazione e accedere alla stessa. Tutte le azioni intraprendibili sono disgiunte.

4.1.1.1 Registrazione

Per il dettaglio delle attività della fase di registrazione si veda il diagramma in figura 23 a pagina 20.

4.1.1.2 Segnalazione anonima

La segnalazione anonima può essere eseguita da un segnalante non registrato oppure da un segnalante non autenticato nel sistema e prevede l'inserimento obbligatorio di un indirizzo e-mail per poter comunicare al segnalante le variazioni riguardanti lo stato delle proprie segnalazioni.

4.1.1.3 Login

Il segnalante deve inserire le credenziali scelte in fase di registrazione per l'accesso alla parte privata del sistema; nel caso in cui la verifica non venga superata viene proposto il reinserimento, altrimenti il segnalante visualizza la pagina principale dell'utente autenticato. A questo punto può inviare una segnalazione, modificare i propri dati e visualizzare lo stato delle segnalazioni. Una volta inoltrata la segnalazione ne viene controllata la coerenza e la completezza dei dati inseriti fornendo la possibilità di correggerli nel caso non superino la verifica. Inviata correttamente la segnalazione se l'utente è anonimo viene rimandato alla pagina principale dell'applicazione, mentre se l'utente è autenticato ritorna alla pagina con l'elenco delle sue richieste aperte. Per il dettaglio dell'attività di modifica dei dati personali si veda il diagramma in figura 22 a pagina 20. Per il dettaglio dell'attività di invio della segnalazione si veda diagramma 21 a pagina 19.

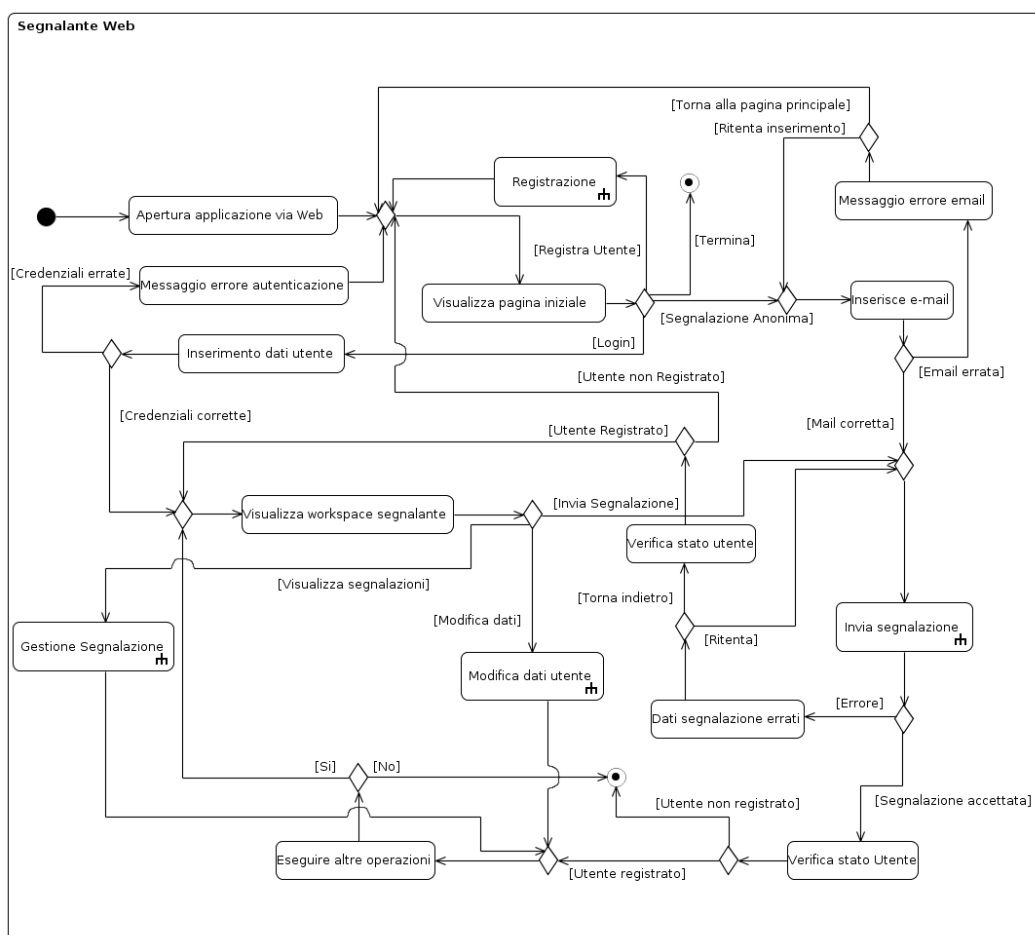


Figura 9: DA1 - Diagramma di attività del segnalante per l'interfaccia Web

4.1.2 Segnalante Android

Il diagramma in figura 10 a pagina 12 illustra come un segnalante possa accedere e interagire con il sistema tramite l'applicazione per smartphone Android. Al primo avvio dell'applicazione all'utente viene chiesto di inserire le credenziali per l'accesso alla parte privata del sistema: nel caso di errore ne viene proposto il reinserimento e in caso l'utente non sia ancora registrato nel sistema viene proposta la registrazione. Quest'ultima procedura viene eseguita una sola volta. Da questo momento il segnalante visualizza l'interfaccia tramite la quale interagisce con il sistema e, in particolare, può inviare segnalazioni e verificare lo stato di quelle aperte.

4.1.2.1 Invia segnalazione

Per il dettaglio delle attività dell'invio della segnalazione si veda il diagramma in figura 21 a pagina 19.

4.1.2.2 Visualizza stato segnalazione

Oltre alla visualizzazione dello stato della segnalazione (per vedere il dettaglio di questa attività si veda il diagramma in figura 20 a pagina 18), il segnalante può decidere se inserire un allegato o un commento oppure procedere con altre operazioni.

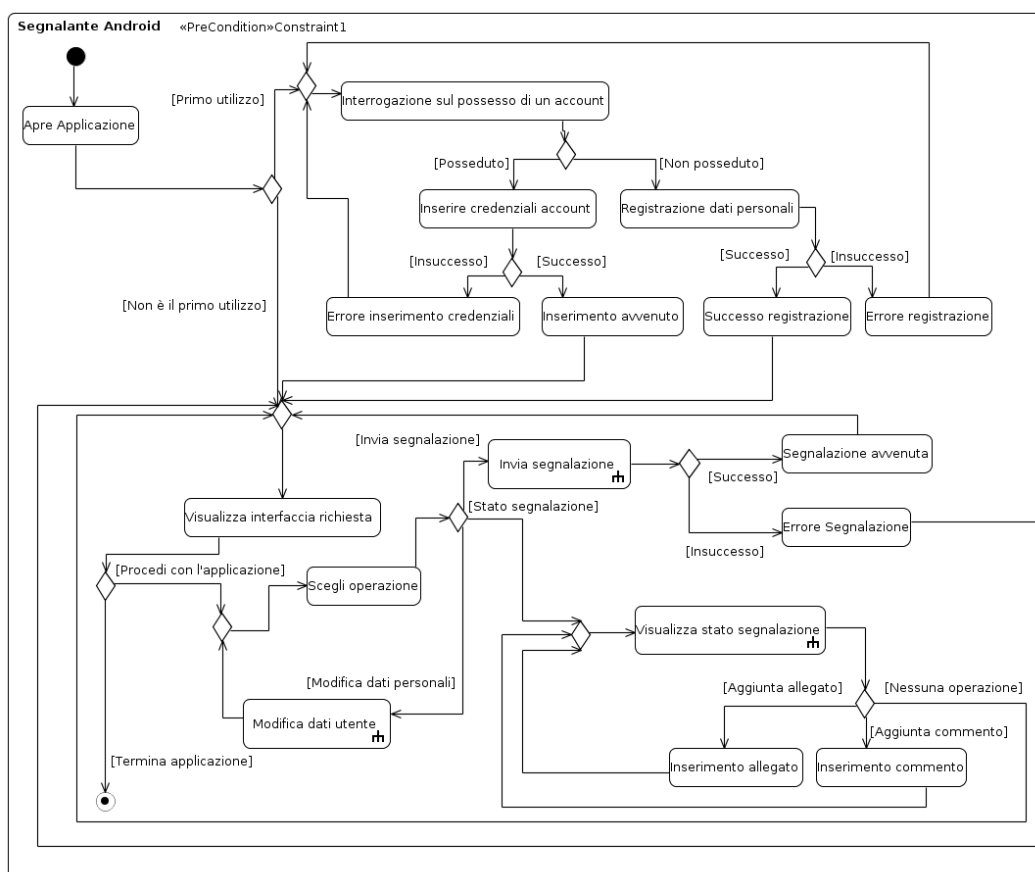


Figura 10: DA2 - Diagramma di attività del segnalante per l'applicazione Android

4.2 Amministratore

4.2.1 Amministratore d'autorità

Il diagramma in figura 11 a pagina 13 illustra le operazioni che possono essere intraprese da un amministratore di autorità.

Prima di poter compiere qualsiasi operazione l'amministratore deve essere autenticato nel sistema, e solo dopo aver compiuto questo passo può gestire una catastrofe, visualizzarne le statistiche, gestire gli operatori e le segnalazioni.

4.2.1.1 Autenticazione

Per il dettaglio delle attività dell'autenticazione si veda il diagramma in figura 14 a pagina 15.

4.2.1.2 Gestione catastrofe

Per il dettaglio delle attività della gestione catastrofe si veda il diagramma in figura 17 a pagina 17.

Questo ramo può essere iterato più volte per gestire differenti catastrofi prima di tornare alla finestra principale.

4.2.1.3 Gestione operatore

Per il dettaglio delle attività di gestione dell'operatore si veda diagramma in figura 18 a pagina 18.

Questo ramo può essere iterato più volte per gestire differenti operatori prima di tornare alla finestra principale.

4.2.1.4 Gestione richieste

Per il dettaglio delle attività della gestione richieste si veda il diagramma in figura 19 a pagina 18.

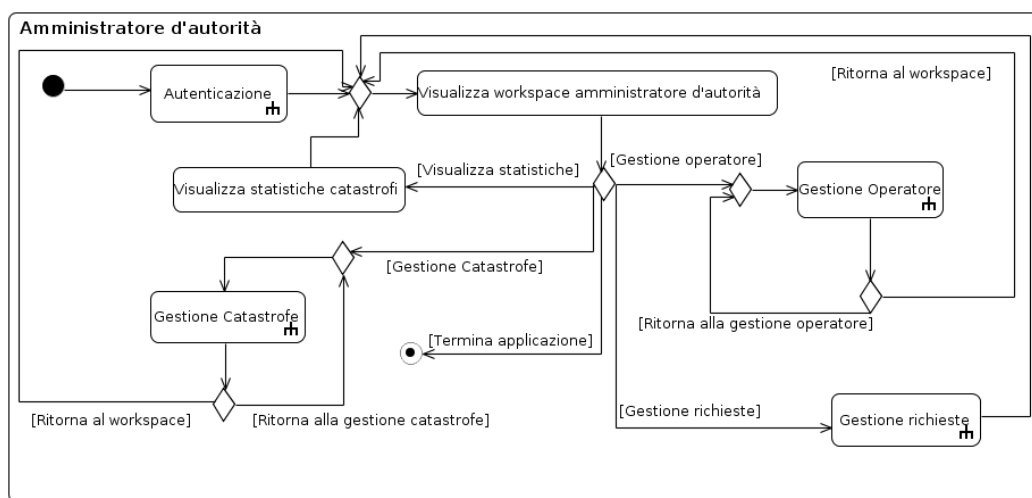


Figura 11: DA3 - Diagramma di attività dell'amministratore di autorità

4.2.2 Amministratore di sistema

Il diagramma in figura 12 a pagina 14 illustra le operazioni che possono essere intraprese da un amministratore di sistema.

Prima di poter compiere qualsiasi operazione l'amministratore deve essere autenticato nel sistema e solo dopo aver compiuto questo passo può creare una nuova autorità, creare un nuovo amministratore di autorità, visualizzare le statistiche delle autorità e gestire le autorità.

4.2.2.1 Autenticazione

Per il dettaglio delle attività dell'autenticazione si veda il diagramma in figura 14 a pagina 15.

4.2.2.2 Registra nuovo utente di tipo autorità

La registrazione di un nuovo utente di tipo autorità implica l'inserimento dei dati ad esso relativi e l'associazione di almeno un amministratore di autorità, pertanto, se nel database non è presente l'amministratore che si vuole associare all'autorità, viene richiesta la creazione di un nuovo amministratore.

4.2.2.3 Crea amministratore

Per il dettaglio delle attività della registrazione di un nuovo amministratore si veda il diagramma in figura 15 a pagina 16.

4.2.2.4 Gestione autorità

Per il dettaglio delle attività della gestione delle autorità si veda il diagramma in figura 16 a pagina 17.

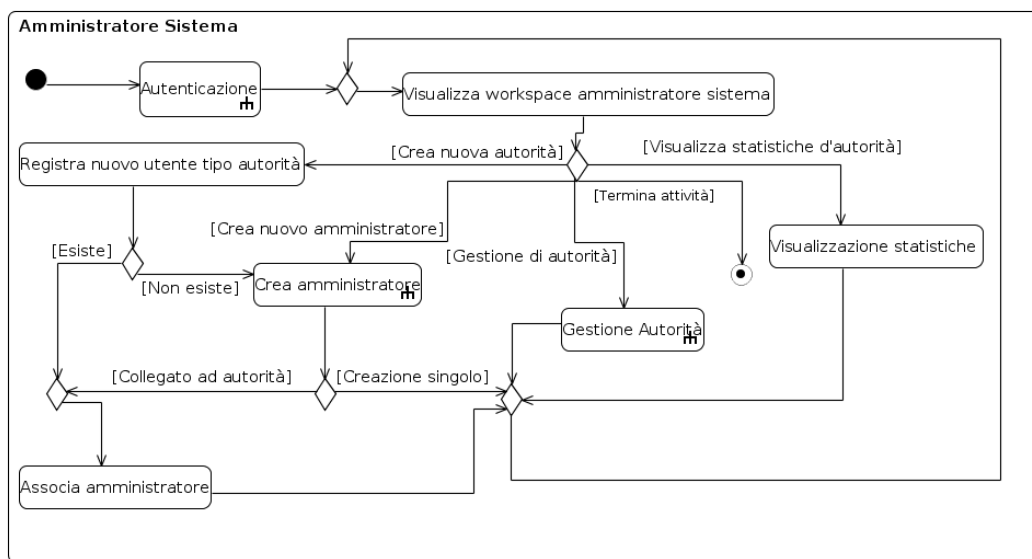


Figura 12: DA4 - Diagramma di attività dell'amministratore di sistema

4.3 Operatore

Il diagramma in figura 13 a pagina 15 illustra le operazioni che possono essere intraprese da un operatore sia esso un operatore base o un operatore avanzato. La differenza tra le due tipologie sarà resa esplicitata nel dettaglio della gestione richieste.

4.3.1 Autenticazione

Per il dettaglio delle attività dell'autenticazione si veda il diagramma in figura 14 a pagina 15.

4.3.2 Gestione richieste

Per il dettaglio delle attività della gestione delle richieste si veda il diagramma in figura 19 a pagina 18.

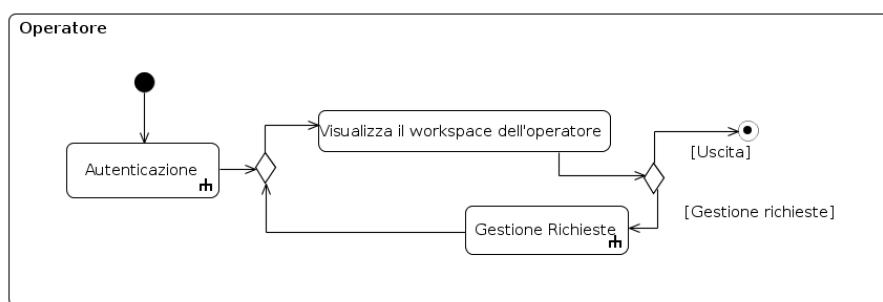


Figura 13: DA5 - Diagramma di attività dell'operatore

4.4 Sottoattività

4.4.1 Autenticazione

Il diagramma in figura 14 a pagina 15 illustra le operazioni che si intraprendono per autenticarsi al sistema.

Dopo aver aperto l'applicazione l'utente visualizza un'interfaccia tramite la quale può scegliere di autenticarsi nel sistema. Dopo aver inserito le credenziali, queste verranno verificate e se sono state inserite correttamente l'utente potrà proseguire con l'utilizzo dell'applicazione, altrimenti verrà indirizzato nuovamente alla pagina di autenticazione.

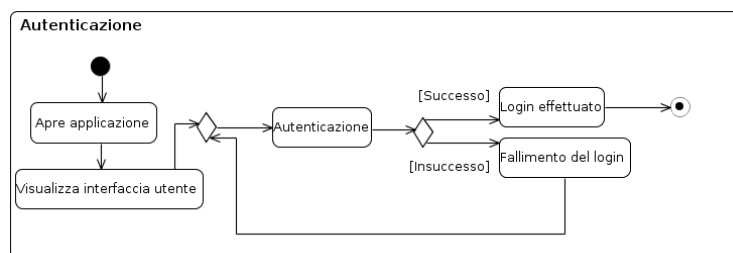


Figura 14: DA6 - Diagramma di attività per l'autenticazione

4.4.2 Crea nuovo amministratore

Il diagramma in figura 15 a pagina 16 illustra le operazioni che l'amministratore di sistema intraprende per creare un nuovo amministratore.

L'amministratore di sistema, dopo essersi autenticato, registra un nuovo utente (per il dettaglio delle attività di registrazione si veda il diagramma in figura 23 a pagina 20). Nel caso in cui i dati non siano coerenti con la figura di amministratore ne viene proposto il reinserimento, in caso contrario potrà procedere assegnando i corretti permessi di amministratore all'utente appena creato e in seguito confermare l'operazione.

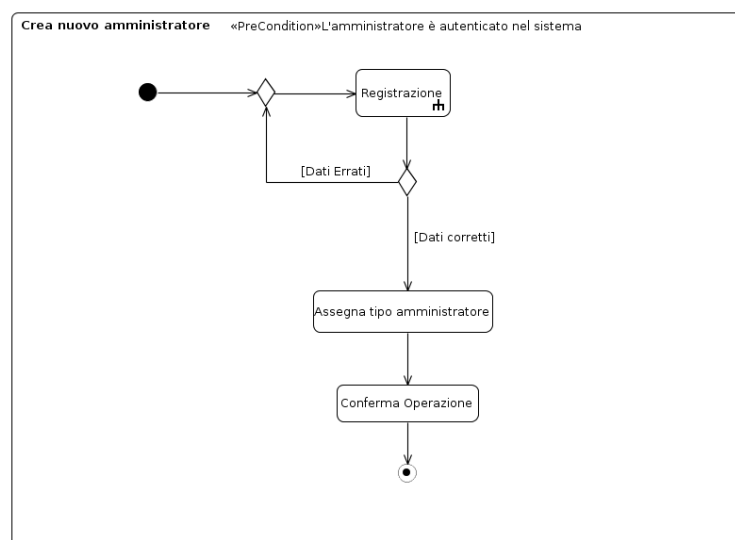


Figura 15: DA7 - Diagramma di attività della creazione di un nuovo amministratore

4.4.3 Gestione Autorità

Il diagramma in figura 16 a pagina 17 illustra le operazioni che l'amministratore di sistema può compiere per la gestione delle autorità.

Come prima cosa viene visualizzato l'elenco delle autorità registrate nel sistema e da qui l'amministratore di sistema può operare due scelte: modificare i dati dell'autorità oppure associare un nuovo operatore all'autorità.

4.4.3.1 Associa operatore

Questa azione permette di assegnare un operatore all'autorità scelta in precedenza. L'amministratore di sistema può scegliere di associare un amministratore tra la lista di amministratori visualizzata oppure creare un nuovo amministratore e associarlo all'autorità.

Per il dettaglio delle attività della creazione di un nuovo amministratore si veda il diagramma in figura 15 a pagina 16.

4.4.3.2 Modifica dati

Questa azione permette all'amministratore di modificare i dati di una autorità. I dati vengono visualizzati e, una volta modificati, l'amministratore deve confermare la modifica al sistema che si occuperà di controllarne la coerenza e di notificare all'amministratore nel caso in cui ci siano dati non correttamente inseriti.

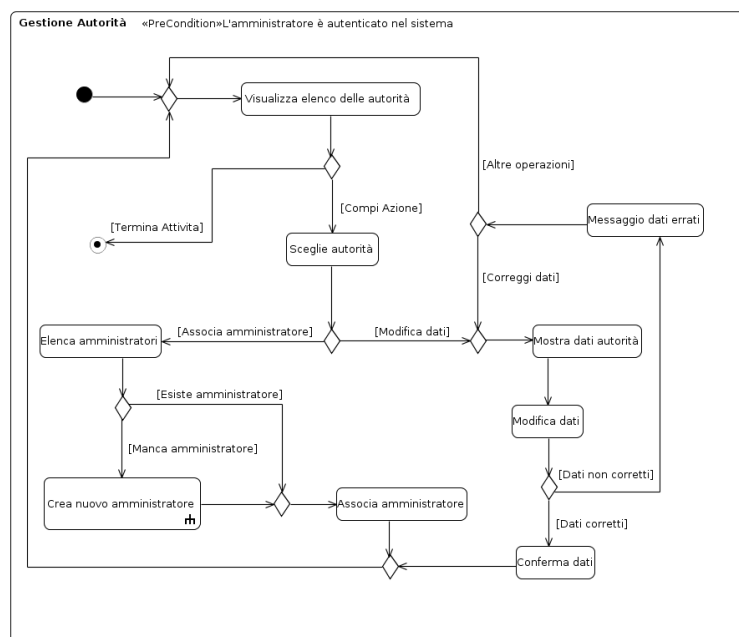


Figura 16: DA8 - Diagramma di attività della gestione di un'autorità

4.4.4 Gestione Catastrofe

Il diagramma in figura 17 a pagina 17 illustra le operazioni che l'amministratore di autorità intraprende per gestire le informazioni di una catastrofe.

Dopo essersi autenticato nel sistema l'amministratore di autorità visualizza l'interfaccia per gestire le catastrofi: può inserirne una nuova o modificare le informazioni di una catastrofe già esistente.

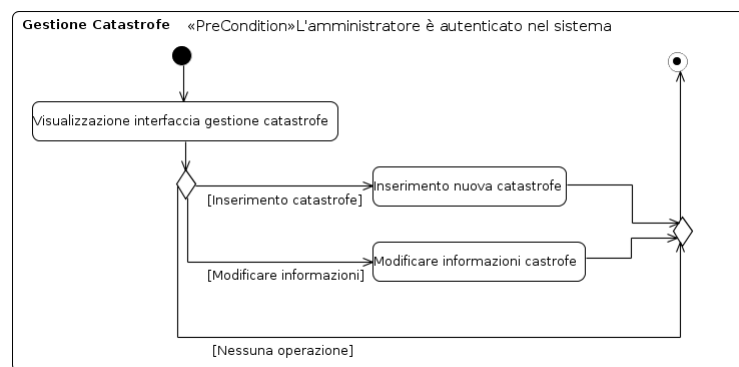


Figura 17: DA9 - Diagramma di attività della gestione catastrofe

4.4.5 Gestione Operatore

Il diagramma in figura 18 a pagina 18 illustra le operazioni che l'amministratore di autorità intraprende per gestire un operatore.

Dopo aver visualizzato l'interfaccia per la gestione degli operatori l'amministratore di autorità può creare un nuovo operatore, attivarne uno già esistente oppure abilitarlo alla gestione delle nuove richieste.

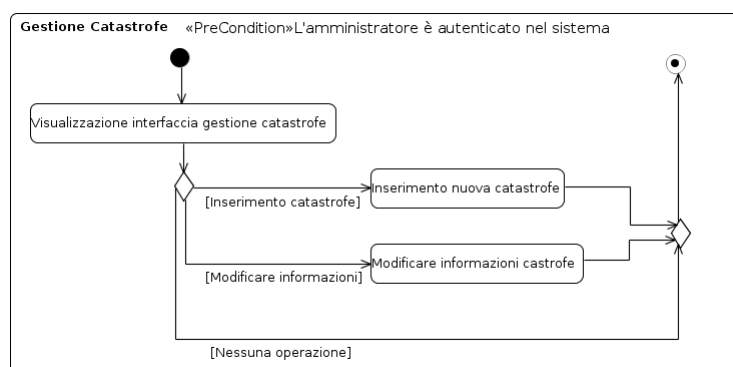


Figura 18: DA10 - Diagramma di attività della gestione operatore

4.4.6 Gestione Richieste

Il diagramma in figura 19 a pagina 18 illustra come vengono gestite le richieste. Dopo essersi autenticato nel sistema, sia l'operatore base che quello avanzato, può assegnare richieste in lavorazione a singoli operatori o a gruppi di operatori, o modificare le informazioni di una richiesta. Solo l'operatore avanzato può assegnare nuove richieste a un operatore o a un gruppo di operatori.

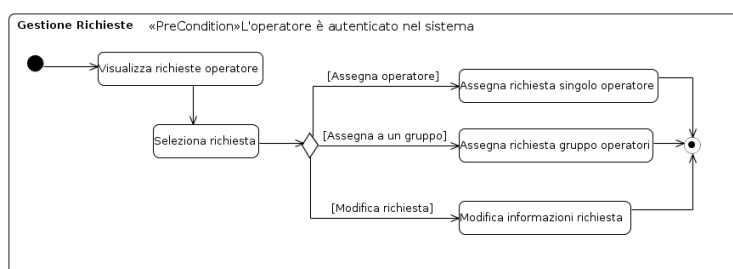


Figura 19: DA11 - Diagramma di attività della gestione richieste

4.4.7 Gestione Segnalazione

Il diagramma in figura 20 a pagina 18 illustra come il segnalante può integrare una segnalazione con maggiori informazioni. Egli può aggiungere sia un allegato che un commento alla segnalazione.

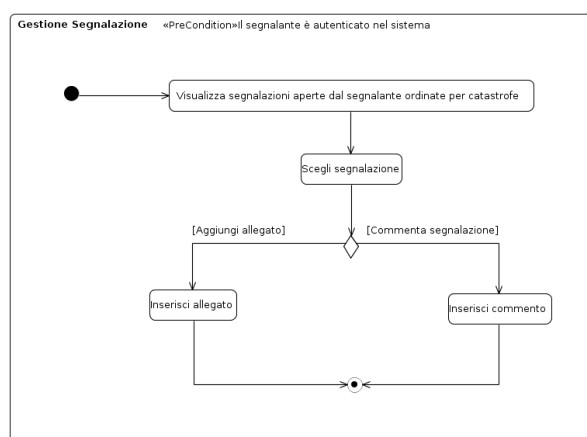


Figura 20: DA12 - Diagramma di attività della segnalazione

4.4.8 Invia Segnalazione

Il diagramma in figura 21 a pagina 19 illustra come si compone una segnalazione da inviare al sistema. Il segnalante, sia esso autenticato o anonimo (la distinzione tra le due tipologie non è importante per la composizione della segnalazione), deve completare tutti i campi richiesti, segnalare la posizione con l'aiuto della mappa messa a disposizione e inserire eventualmente un allegato.

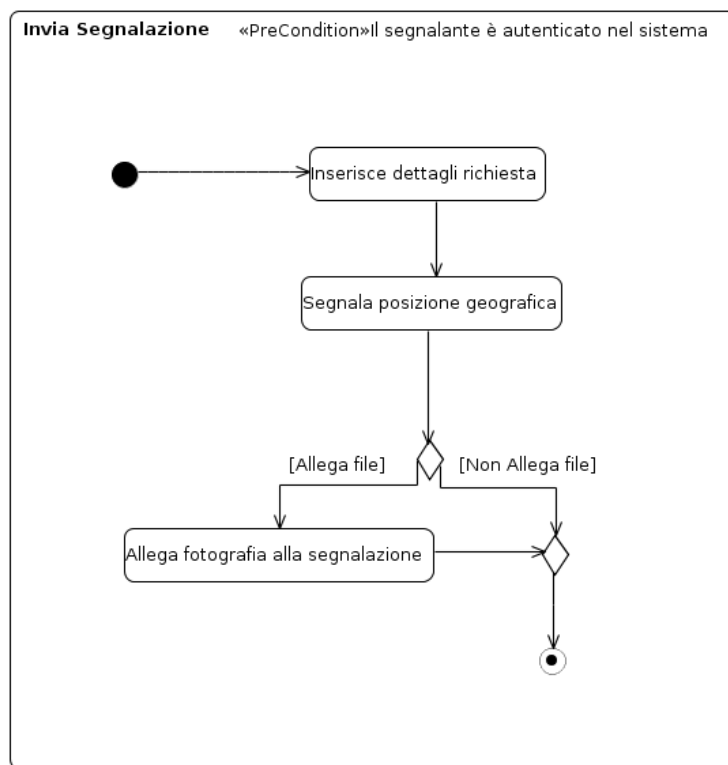


Figura 21: DA13 - Diagramma di attività dell'invio della segnalazione

4.4.9 Modifica dati utente

Il diagramma in figura 22 a pagina 20 illustra come un segnalante, una volta effettuato il login nel sistema, possa decidere di modificare i propri dati personali. Come per la registrazione dell'utente, una volta effettuata la modifica e inoltrata al sistema, i dati vengono verificati e all'utente viene visualizzato il risultato di tale verifica.

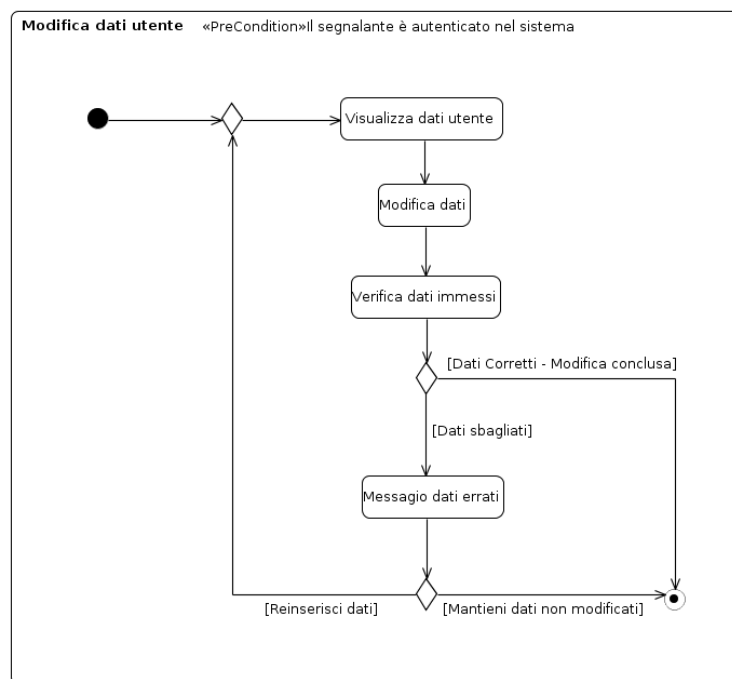


Figura 22: DA14 - Diagramma di attività della modifica dei dati di un utente

4.4.10 Registrazione

Il diagramma in figura 23 a pagina 20 illustra come un utente possa completare la registrazione al sistema. Esso deve completare i campi necessari alla registrazione e una volta inoltrata la richiesta, il sistema verifica i dati immessi e notifica all'utente l'esito dell'operazione.

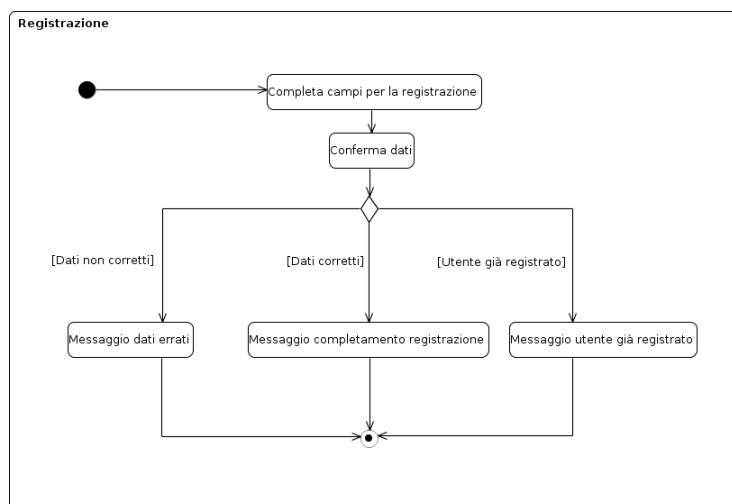


Figura 23: DA15 - Diagramma di attività della registrazione di un utente

5 Descrizione dei singoli componenti

5.1 Model

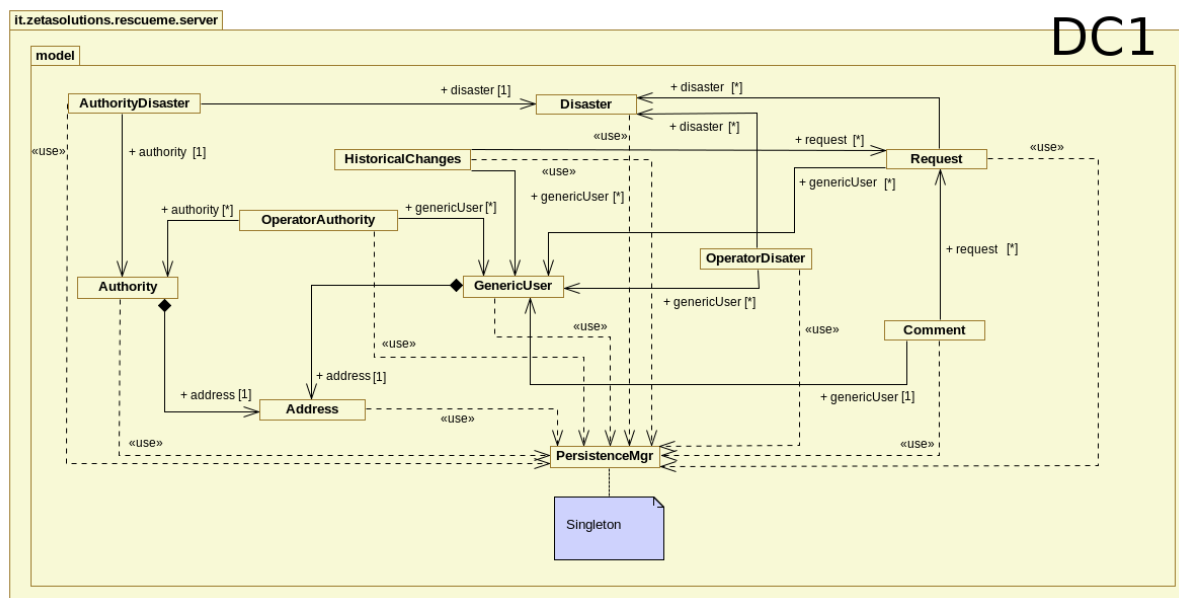


Figura 24: DC1 - Diagramma delle classi che illustra il Model

5.1.1 Descrizione del diagramma

Il componente model si occupa della rappresentazione, della persistenza dei dati dell'applicazione e delle regole di business. Esso permette di incapsulare lo stato dell'applicazione e di rispondere alle query della componente View inoltrate tramite il Controller, notificando quando lo stato dei dati cambia. In particolare, la classe `PersistenceMgr` rispetta il design pattern Singleton fornendo un punto di accesso globale a un'unica istanza di `PersistenceManager` tramite l'interfaccia `PersistenceManagerFactory` in modo da evitare operazioni concorrenti effettuate nel `DataStore`. `PersistenceManager` è l'interfaccia principale per componenti di applicazioni basate su JDO. Essa è la factory che permette di instanziare Query e Transactions, inoltre contiene i metodi per gestire il ciclo di vita degli oggetti persistenti.

5.1.2 Presentazione delle classi

5.1.2.1 Address

Nome: Address

Tipo: class

Package: `it.zetasolutions.rescueme.server`

Descrizione: rappresenta tutte le informazioni attinenti alla residenza di un utente

Funzionalità: la classe prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

5.1.2.2 GenericUser

Nome: GenericUser

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: contiene i campi dati propri di un utente generico che si contraddistingue da un utente di tipo autorità. Essa rappresenta i segnalanti registrati, gli operatori, gli operatori avanzati, gli amministratori di autorità e gli amministratori di sistema

Componenti: contiene un oggetto di tipo Address (vedi sezione [5.1.2.1](#) a pagina [21](#))

Funzionalità: la classe prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

5.1.2.3 Authority

Nome: Authority

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: contiene i campi dati propri di un'autorità che si contraddistingue da un utente di tipo generico

Componenti: la classe contiene un oggetto di tipo Address (vedi sezione [5.1.2.1](#) a pagina [21](#))

Funzionalità: la classe prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

5.1.2.4 OperatorDisaster

Nome: OperatorDisaster

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: rappresenta l'associazione tra gli operatori e le catastrofi che possono gestire

Funzionalità: prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

5.1.2.5 Disaster

Nome: Disaster

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: rappresenta la catastrofe

Funzionalità: prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

5.1.2.6 Request

Nome: Request

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: rappresenta una richiesta di soccorso effettuata da un segnalante a seguito di una catastrofe

Funzionalità: prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

5.1.2.7 Comment

Nome: Comment

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: rappresenta un commento inserito in una richiesta di soccorso

Funzionalità: prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

5.1.2.8 PersistenceMgr

Nome: PersistenceMgr

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: serve per creare, salvare, modificare, recuperare ed eliminare oggetti del DataStore

Design Pattern: Singleton (vedi sezione [7.1](#) a pagina [38](#))

5.1.2.9 HistoricalChanges

Nome: HistoricalChanges

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: serve per mantenere lo storico di cambiamenti di stato delle segnalazioni

5.1.2.10 OperatorAuthority

Nome: OperatorAuthority

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: ha il compito di definire quali operatori appartengono a una determinata autorità

5.1.2.11 AuthorityDisaster

Nome: AuthorityDisaster

Tipo: class

Package: it.zetasolutions.rescueme.server.model

Descrizione: ha il compito di definire quali autorità operano in una catastrofe

5.2 View

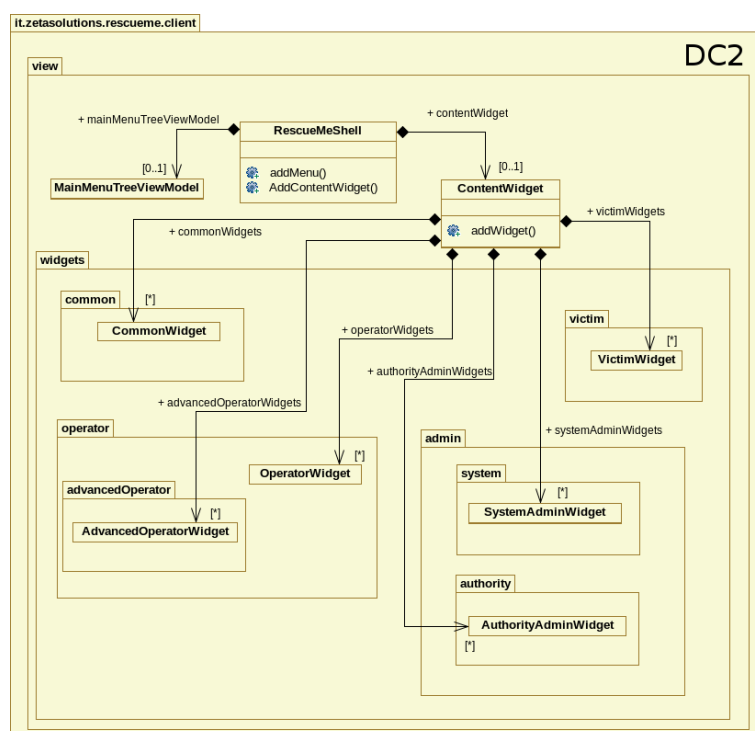


Figura 25: DC2 - Diagramma delle classi che illustra la View

5.2.1 Descrizione del diagramma

La View ha il compito di visualizzare all'utente finale, per mezzo di interfacce web, i dati contenuti nel model e i risultati delle elaborazioni effettuate dal Controller. Come è possibile vedere dalla figura 25 a pagina 24, l'architettura di questa componente all'interno del design pattern MVC risulta abbastanza semplificata in quanto essa verrà espansa completamente nel documento di Definizione di Prodotto; ne verranno descritte tutte le classi accessorie, necessarie al corretto utilizzo della tecnologia GWT all'interno della nostra applicazione.

L'interfaccia utente dell'applicazione sarà composta da un header, contenente il titolo e il logo dell'applicazione, un footer contenente i dati dell'azienda produttrice del software e un menu che, a seconda del livello di privilegi detenuto dall'utente, visualizza uno o più sottomenu.

L'area di lavoro sarà visualizzata invece nella parte destra dello schermo.

5.2.2 Presentazione delle classi

5.2.2.1 RescueMeShell

Nome: RescueMeShell

Tipo: class

Package: it.zetasolutions.rescueme.client.view

Descrizione: fornisce la struttura complessiva dell'interfaccia utente dell'applicazione

Funzionalità:

addMenu(): aggiunge il menu all'interfaccia utente;

addContentWidget(): aggiunge il riquadro per la visualizzazione del widget selezionato.

5.2.2.2 MainMenuTreeViewModel

Nome: MainMenuTreeViewModel

Tipo: class

Package: it.zetasolutions.rescueme.client.view

Descrizione: fornisce il menu utente in base al livello di autenticazione dell'utente stesso

5.2.2.3 ContentWidget

Nome: ContentWidget

Tipo: class

Package: it.zetasolutions.rescueme.client.view

Descrizione: fornisce l'area di lavoro su cui verrà visualizzato il widget attivo

Funzionalità:

addWidget(): aggiunge il widget selezionato all'area di lavoro.

5.2.2.4 Widget

Nome: Widget

Tipo: class

Package: it.zetasolutions.rescueme.client.view.widgets

Descrizione: fornisce diversi widget che compongono l'intera applicazione

5.3 Controller

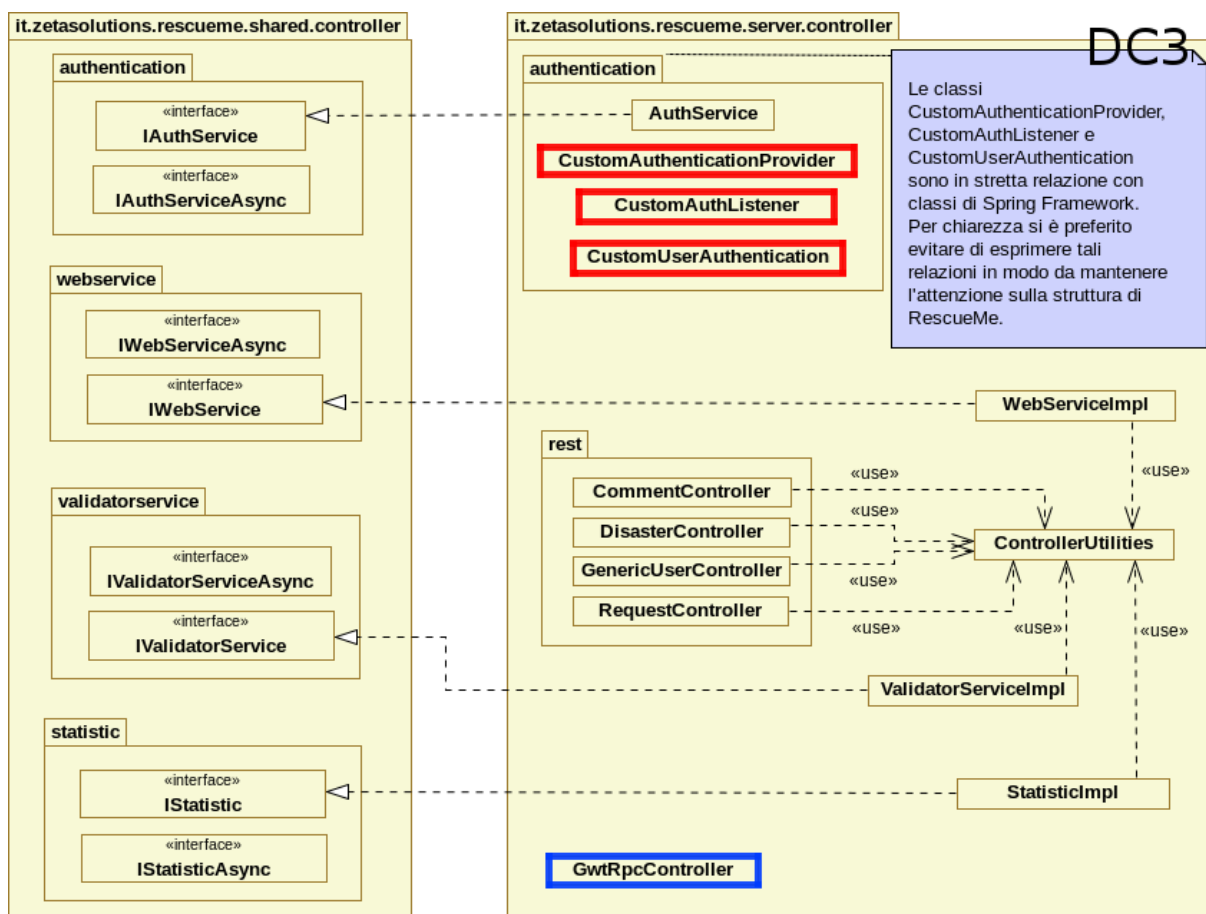


Figura 26: DC3 - Diagramma delle classi che illustra il Controller

5.3.1 Descrizione del diagramma

La componente Controller di MVC ha il compito di gestire le richieste provenienti dalla View, di elaborarle e di richiedere alla parte Model di aggiungere, modificare o restituire alla View dei dati in essa contenuti. Questa componente contiene inoltre classi che gestiscono l'autenticazione dell'utente nel sistema RescueMe.

Le classi evidenziate dal riquadro rosso sono classi in stretta relazione con lo Spring Security Framework; per non complicare il diagramma si è scelto di non esplicitare tali relazioni.

La classe evidenziata dal riquadro blu è in stretta relazione con lo Spring Framework; anche in questo caso si è scelto di non esplicitare tale relazione per non complicare il diagramma.

Le interfacce dei sottopackage del package shared che non hanno relazioni con altri componenti sono espresse in tale modo perché sono strettamente correlate al funzionamento di GWT.

5.3.2 Presentazione delle classi

5.3.2.1 ControllerUtilities

Nome: ControllerUtilities

Tipo: class

Package: it.zetasolutions.rescueme.server.controller

Descrizione: contiene metodi comuni per l'elaborazione dei dati o di altri servizi necessari alle altre classi del controller

5.3.2.2 IAuthService

Nome: IAuthService

Tipo: interface

Package: it.zetasolutions.rescueme.shared.controller.authentication

Descrizione: fornisce l'interfaccia pubblica per instanziare AuthServiceImpl e richiamare il metodo in essa contenuto

5.3.2.3 IAuthServiceAsync

Nome: IAuthServiceAsync

Tipo: interface

Package: it.zetasolutions.rescueme.shared.controller.authentication

Descrizione: permette a GWT di gestire le richieste inviate al server tramite AuthServiceImpl come asincrone

5.3.2.4 AuthServiceImpl

Nome: AuthServiceImpl

Tipo: class

Package: it.zetasolutions.rescueme.server.controller.authentication

Descrizione: implementa il metodo dichiarato in IAuthService e si occupa di fornire informazioni sull'utente autenticato

Funzionalità:

retrieveUsername(): restituisce un Vector di stringhe in cui il primo elemento contiene l'e-mail dell'utente che ha eseguito il metodo, e il secondo contiene il livello con cui l'utente è registrato nel sistema.

5.3.2.5 IWebService

Nome: IWebService

Tipo: interface

Package: it.zetasolutions.rescueme.shared.controller.webservice

Descrizione: fornisce l'interfaccia pubblica per instanziare WebServiceImpl e richiamare i metodi in essa contenuti

5.3.2.6 IWebServiceAsync

Nome: IWebServiceAsync

Tipo: interface

Package: it.zetasolutions.rescueme.shared.controller.webservice

Descrizione: permette a GWT di gestire le richieste inviate al server tramite Auth WebServiceImpl come asincrone

5.3.2.7 WebServiceImpl

Nome: WebServiceImpl

Tipo: class

Package: it.zetasolutions.rescueme.server.controller

Descrizione: implementa i metodi dichiarati in IWebService, e si occupa di fornire dei metodi tramite i quali reperire informazioni dal DataStore richiamando il PersistenceMgr attraverso l'interfaccia IPersistenceMgr presente nel package shared.model (vedi sezione [5.1.2.8](#) a pagina [23](#))

5.3.2.8 IValidatorService

Nome: IValidatorService

Tipo: interface

Package: it.zetasolutions.rescueme.shared.controller.validator

Descrizione: fornisce l'interfaccia pubblica per instanziare ValidatorServiceImpl e richiamare i metodi in essa contenuti

5.3.2.9 IValidatorServiceAsync

Nome: IValidatorServiceAsync

Tipo: interface

Package: it.zetasolutions.rescueme.shared.controller.validator

Descrizione: permette a GWT di gestire le richieste inviate al server tramite ValidatorServiceImpl come asincrone

5.3.2.10 ValidatorServiceImpl

Nome: ValidatorServiceImpl

Tipo: class

Package: it.zetasolutions.rescueme.server.controller

Descrizione: rappresenta un'entità validatrice che ha il compito di controllare tutti i dati che un utente potrebbe inserire nelle form del sistema. Qualora i dati superano il controllo vengono inseriti nel sistema richiamando il PersistenceMgr attraverso l'interfaccia IPersistenceMgr presente nel package shared.model (vedi sezione [5.1.2.8](#) a pagina [23](#))

5.3.2.11 IStatistics

Nome: IStatistics

Tipo: interface

Package: it.zetasolutions.rescueme.shared.controller.statistic

Descrizione: fornisce l'interfaccia pubblica per instanziare StatisticImpl e richiamare i metodi in essa contenuti

5.3.2.12 IStatisticsAsync

Nome: IStatisticsAsync

Tipo: interface

Package: it.zetasolutions.rescueme.shared.controller.statistic

Descrizione: permette a GWT di gestire le richieste inviate al server tramite StatisticImpl come asincrone

5.3.2.13 Statistic

Nome: Statistic

Tipo: class

Package: it.zetasolutions.rescueme.server.controller

Descrizione: elabora e genera delle statistiche riguardanti il sistema

5.3.2.14 CommentController

Nome: CommentController

Tipo: class

Package: it.zetasolutions.rescueme.server.controller.rest

Descrizione: raccoglie tutte le implementazioni dei metodi RESTful disponibili per l'interazione con la classe Comment senza conoscerne la reale struttura

Funzionalità: vengono forniti i metodi per eseguire l'inserimento e il recupero dati

5.3.2.15 DisasterController

Nome: DisasterController

Tipo: class

Package: it.zetasolutions.rescueme.server.controller.rest

Descrizione: raccoglie tutte le implementazioni dei metodi RESTful disponibili per l'interazione con la classe Disaster senza conoscerne la reale struttura

Funzionalità: vengono forniti i metodi per eseguire l'inserimento e il recupero dati

5.3.2.16 GenericUserController

Nome: GenericUserController

Tipo: class

Package: it.zetasolutions.rescueme.server.controller.rest

Descrizione: raccoglie tutte le implementazioni dei metodi RESTful disponibili per l'interazione con la classe GenericUser senza conoscerne la reale struttura

Funzionalità: vengono forniti i metodi per eseguire l'inserimento e il recupero dati

5.3.2.17 RequestController

Nome: RequestController

Tipo: class

Package: it.zetasolutions.rescueme.server.controller.rest

Descrizione: raccoglie tutte le implementazioni dei metodi RESTful disponibili per l'interazione con la classe Request senza conoscerne la reale struttura

Funzionalità: vengono forniti i metodi per eseguire l'inserimento e il recupero dati

5.3.2.18 CustomAuthenticationProvider

Nome: CustomAuthenticationProvider

Tipo: class

Package: it.zetasolutions.rescueme.server.controller.authentication

Descrizione: gestisce le richieste di autenticazione delegate dalla classe AuthenticationManager di Spring Security

5.3.2.19 CustomAuthListener

Nome: CustomAuthListener

Tipo: class

Package: it.zetasolutions.rescueme.server.controller.authentication

Descrizione: ha il compito di eseguire il metodo in essa contenuto quando viene rilevato l'evento autenticazione nel sistema

5.3.2.20 CustomUserAuthentication

Nome: CustomUserAuthentication

Tipo: class

Package: it.zetasolutions.rescueme.server.controller.authentication

Descrizione: permette di personalizzare i metodi predefiniti di Spring security che gestiscono le fasi di autenticazione dell'utente

5.3.2.21 GwtRpcController

Nome: GwtRpcController

Tipo: class

Package: it.zetasolutions.rescueme.server.controller

Descrizione: si occupa di gestire le richieste di esecuzione di un determinato metodo smistandole nel controller adeguato

6 Applicazione Android

L'applicazione Android viene sviluppata utilizzando SDK 2.3.1 corredato dalle API 8 fornite da Google. Esso permette di creare l'applicazione mediante l'utilizzo di file XML_[g] per quanto riguarda il layout grafico, e di classi Java per quanto riguarda l'implementazione delle funzionalità per l'interfacciamento con il sistema RescueMe.

L'applicazione deve poter funzionare su qualsiasi dispositivo con sistema operativo Android indipendentemente dalla natura e dalle caratteristiche dello stesso, in particolare nella progettazione sono stati considerati i seguenti aspetti:

- Limitate risorse hardware dei dispositivi mobile;
- Diversificazione degli smartphone, in particolare:
 - Le JVM_[g] dei dispositivi possono avere rilevanti differenze prestazionali;
 - I processori dei dispositivi possono avere performance molto diverse.

Al fine di realizzare un'applicazione Android il più possibile efficiente si sono tenute conto delle direttive fornite da Google (vedere riferimenti informativi 1.4.2 a pagina 2).

Ad alto livello, l'architettura implementa il design pattern MVC (vedere paragrafo 7.2 a pagina 38) composta dai seguenti tre macro-componenti:

- model;
- controller;
- view.

Ogni componente si può individuare nel diagramma della classi in figura 27 a pagina 32 dove sono presenti tre package con i nomi dei rispettivi componenti del design pattern adottato.

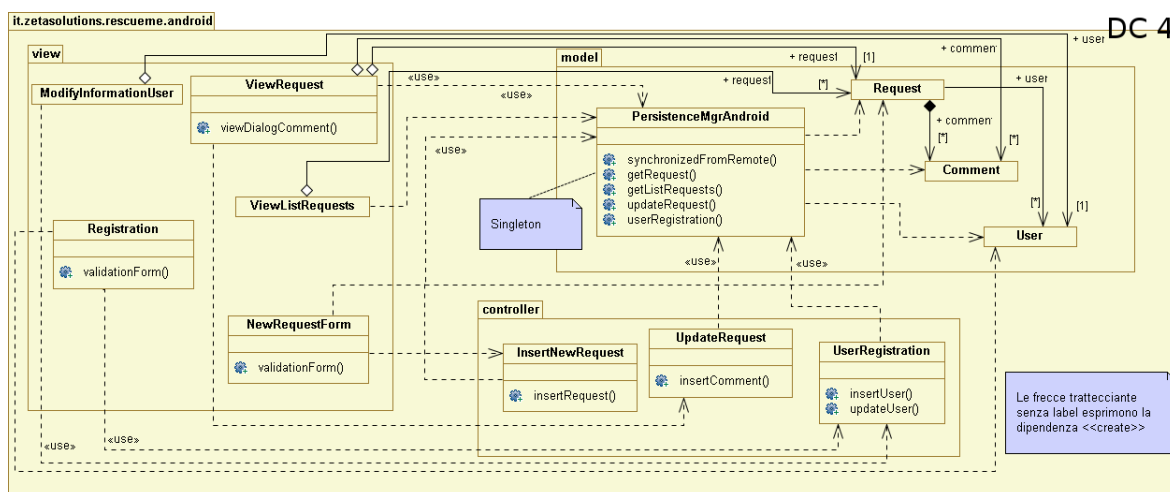


Figura 27: DC4 - Diagramma delle classi per applicazione Android

6.1 Descrizione dei singoli componenti

6.1.1 Model

Il model ha il compito di garantire la persistenza dei dati e di fornirli alle componenti view e controller quando vengono richiesti.

Si occupa del recupero dei dati salvati nelle seguenti risorse:

SQLite database: database locale e interno all'applicazione Android;

DataStore: database remoto ed esterno all'applicazione.

6.1.1.1 Presentazione delle classi

6.1.1.1.1 Request

Nome: Request

Tipo: class

Package: it.zetasolutions.rescueme.android.model

Descrizione: rappresenta una richiesta di soccorso effettuata da un segnalante a seguito di una catastrofe. Contiene le sole informazioni della richiesta inviata dal segnalante

Funzionalità: prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

Componenti: contiene oggetti di tipo Comment (vedere paragrafo [6.1.1.1.2](#) a pagina [33](#))

6.1.1.1.2 Comment

Nome: Comment

Tipo: class

Package: it.zetasolutions.rescueme.android.model

Descrizione: rappresenta un commento inserito in una richiesta di soccorso

Funzionalità: prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

6.1.1.1.3 User

Nome: User

Tipo: class

Package: it.zetasolutions.rescueme.android.model

Descrizione: rappresenta un utente e i suoi dati nell'applicazione Android

Funzionalità: prevede i metodi per poter leggere e modificare i campi dati in essa contenuti

6.1.1.1.4 PersistenceMgrAndroid

Nome: PersistenceMgrAndroid

Tipo: class

Package: it.zetasolutions.rescueme.android.model

Descrizione: ha il compito di gestire e di garantire la persistenza dei dati. Fornisce tutti i metodi d'accesso e di gestione dei dati necessari alle altre componenti dell'applicazione in modo che la persistenza sia trasparente al resto dell'applicazione. Questa classe implementa il design pattern Singleton (vedere paragrafo [7.1](#) a pagina [38](#))

Funzionalità:

synchronizedFromRemote(): sincronizza i dati locali con quelli remoti;

getRequest(): restituisce le informazioni associate a una richiesta di soccorso;

getListRequests(): fornisce tutte le richieste inserite da un utente specifico;

updateRequest(): aggiorna una richiesta di soccorso, ad esempio aggiungendo un commento;

userRegistration(): registra un nuovo utente nel sistema.

6.1.2 View

L'utente interagisce con l'applicazione attraverso le componenti presenti nel package *it.zetasolutions.rescueme.android.view* (vedi diagramma delle classi [27](#) a pagina [32](#)) che forniscono le opportune interfacce grafiche d'interazione.

L'applicazione è composta da cinque classi che estendono la classe *android.app.Activity* o una sua derivata. La classe *Activity* rappresenta un'interfaccia utente che permette all'utilizzatore dell'applicazione di interagire con essa.

6.1.2.1 Presentazione delle classi

6.1.2.1.1 Registration

Nome: Registration

Tipo: class

Package: it.zetasolutions.rescueme.android.view

Estende: android.app.Activity

Descrizione: fornisce la form per l'inserimento dei dati necessari alla registrazione di un utente nel sistema RescueMe

Funzionalità:

validationForm(): verifica che i dati inseriti dall'utente rispettino la forma prevista (es. l'indirizzo e-mail formato aaa@bbbb.ccc) e richiama il componente UserRegistration (vedi paragrafo [6.1.3.1.3](#) a pagina [37](#)) per la registrazione nel sistema.

Componenti: crea oggetti di tipo UserRegistration

6.1.2.1.2 NewRequestForm

Nome: NewRequestForm

Tipo: class

Package: it.zetasolutions.rescueme.android.view

Estende: android.app.Activity

Descrizione: fornisce la form per l'inserimento di una nuova richiesta di soccorso nel sistema RescueMe

Funzionalità:

validationForm(): verifica che i dati inseriti dall'utente rispettino la forma prevista e che tutti quelli necessari siano effettivamente inseriti, dopodiché richiama il componente InsertNewRequest (vedi paragrafo 6.1.3.1.1 a pagina 36) per l'inserimento della richiesta nel sistema.

Componenti: crea oggetti di tipo UserRegistration (vedi paragrafo 6.1.3.1.3 a pagina 37)

6.1.2.1.3 ViewRequest

Nome: ViewRequest

Tipo: class

Package: it.zetasolutions.rescueme.android.view

Estende: android.app.Activity

Descrizione: visualizza tutti i dettagli di una specifica richiesta di soccorso

Funzionalità:

viewDialogComment(): fornisce la form per l'inserimento di un commento collegato alla richiesta di soccorso selezionata.

6.1.2.1.4 ViewListRequests

Nome: ViewListRequests

Tipo: class

Package: it.zetasolutions.rescueme.android.view

Estende: android.app.Activity

Descrizione: visualizza la lista di tutte le richieste che l'utente autenticato nell'applicazione ha inserito, sia dall'applicazione stessa che da altre interfacce grafiche. Selezionando una particolare richiesta, l'utente vede tutte le informazioni legate a esse (vedi paragrafo 6.1.2.1.3 a pagina 35)

Nota: la classe *NON* crea oggetti di tipo ViewRequest, ma avvia una nuova Activity di tipo ViewRequest

6.1.2.1.5 ModifyInformationUser

Nome: ModifyInformationUser

Tipo: class

Package: it.zetasolutions.rescueme.android.view

Estende: android.app.Activity

Descrizione: fornisce un form che permette all'utente di modificare i dati inseriti in fase di registrazione

6.1.3 Controller

I componenti presenti nel controller, package *it.zetasolutions.rescueme.android.controller* (vedi diagramma delle classi [8](#) a pagina [9](#)), hanno il compito di ricevere le richieste dell'utente provenienti dalla view e di attuarle modificando lo stato della view stessa e del model.

6.1.3.1 Presentazione delle classi

6.1.3.1.1 InsertNewRequest

Nome: InsertNewRequest

Tipo: class

Package: it.zetasolutions.rescueme.android.controller

Descrizione: riceve i dati per l'inserimento di una nuova richiesta di soccorso, esegue gli opportuni controlli e caricamenti (ad esempio l'eventuale foto allegata) e inserisce la richiesta nel sistema richiamando il persistence manager reso disponibile dal model

Funzionalità:

insertRequest(): inserisce la richiesta di soccorso nel sistema.

Componenti: crea un componente di tipo PersistenceMgrAndroid (vedere paragrafo [6.1.1.1.4](#) a pagina [34](#)) per richiamare il servizio del model

6.1.3.1.2 UpdateRequest

Nome: UpdateRequest

Tipo: class

Package: it.zetasolutions.rescueme.android.controller

Descrizione: aggiorna i dati di una richiesta di soccorso

Funzionalità:

insertComment(): effettua le operazioni necessarie per l'inserimento di un commento nel sistema RescueMe richiamando il persistence manager offerto dal model

Componenti: crea un componente di tipo PersistenceMgrAndroid (vedere paragrafo [6.1.1.1.4](#) a pagina [34](#)) per richiamare il servizio del model

6.1.3.1.3 UserRegistration

Nome: UserRegistration

Tipo: class

Package: it.zetasolutions.rescueme.android.controller

Descrizione: verifica che l'indirizzo e-mail inserito non sia già presente nel sistema: se non è presente inserisce la registrazione nel sistema RescueMe richiamando il persistence manager offerto dal model

Funzionalità:

insertUser(): effettua la registrazione di un nuovo utente nel sistema;

updateUser(): effettua l'aggiornamento dei dati appartenenti a un utente.

Componenti: crea un componente di tipo PersistenceMgrAndroid (vedere paragrafo [6.1.1.1.4](#) a pagina [34](#)) per richiamare il servizio del model

6.2 Comunicazione con il sistema RescueMe

L'applicazione Android comunicherà con il sistema RescueMe attraverso delle API RESTful in Spring MVC, che permettono una facile integrazione di un'applicazione esterna con un sistema centrale tramite la re-direzione di determinate richieste Web in metodi e/o classi progettate specificatamente per questo scopo.

La mappatura degli URI tramite la notazione *@RequestMapping* di Spring permette di ricavarne i parametri e di inviarli ai metodi interessati. Questa integrazione fornirà inoltre le funzionalità necessarie per la configurazione di altri software con il sistema.

Inviando delle richieste *HTTP* formattate in un modo preciso, l'applicazione potrà richiedere al sistema l'inserimento, l'aggiornamento, la lettura e la cancellazione di determinati dati.

6.2.1 Protezione URI API RESTful

Gli URI delle API RESTful verranno protetti da accessi non autorizzati attraverso il framework *Spring Security 3.1.0 RC* o superiore che permette di direzionare le richieste su determinati URI su canale protetto *HTTPS* e di richiederne l'autenticazione *http-basic*.

6.2.2 Persistenza dati

La persistenza dei dati dell'applicazione viene effettuata usando le seguenti risorse:

Database SQLite: i dati vengono memorizzati localmente nel dispositivo dell'utente;

DataStore: i dati vengono memorizzati in remoto.

In questo modo è possibile velocizzare il recupero delle informazioni richieste dall'utente andandole a recuperare localmente. I dati locali verranno costantemente sincronizzati con quelli presenti nel DataStore remoto, in questo modo solo i dati nuovi e/o aggiornati verranno scaricati dall'applicazione: questo permette una più rapida elaborazione delle richieste e un'occupazione della banda dell'utente solo per il trasferimento di nuovi dati e/o aggiornamenti.

7 Design pattern

Nell'architettura del sistema RescueMe sono stati applicati dei design pattern che forniscono delle soluzioni note, corrette e ripetute per alcuni problemi.

7.1 Singleton

Il design pattern Singleton garantisce che una determinata classe possa essere istanziata una sola volta e ne fornisce un punto di accesso globale. Singleton fa parte dei *design pattern creazionali*_[g] e richiede che la classe che lo adotta non disponga di un costruttore pubblico, ma solo privato o protetto (vedi figura 28 a pagina 38).

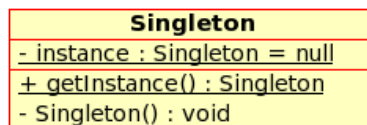


Figura 28: Design Pattern Singleton in UML_[g]

Singleton va utilizzato in tutti gli ambiti in cui è necessario che l'accesso a una certa risorsa sia unico per evitare interferenze e per permettere una gestione ottimale della risorsa stessa, ad esempio nella gestione di un database o di una coda di stampa.

Questo design pattern evita l'uso di variabili globali per far condividere risorse a più parti del sistema.

7.2 MVC

Il pattern architetturale_[g] MVC (vedi figura 29 a pagina 39) è utilizzato in contesti in cui vi è un accesso ai dati tramite tecnologie e modalità non omogenei. Esso prevede di strutturare un'applicazione nei seguenti tre componenti:

Model: si individua la rappresentazione dei dati dell'applicazione e le regole di business con cui tali dati vengono acceduti e modificati;

View: si occupa di presentare graficamente la vista del modello;

Controller: interpreta le richieste della view in azioni che vanno a interagire con il Model aggiornando conseguentemente la View stessa.

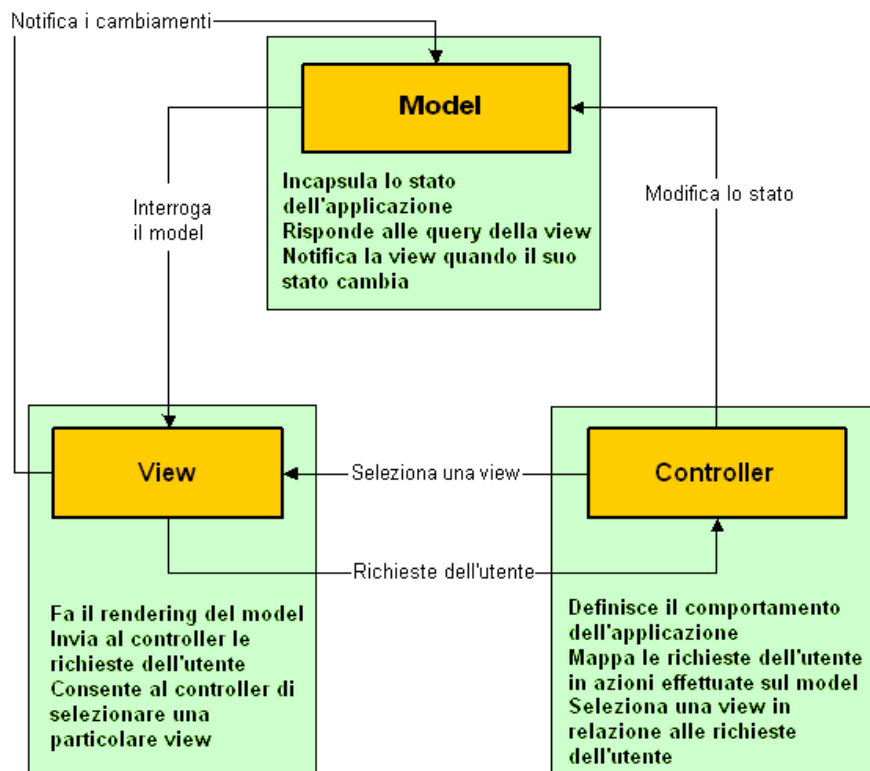


Figura 29: Pattern architetturale MVC

L'intento di questo pattern è di disaccoppiare il più possibile tra loro le componenti dell'applicazione. Questo approccio porta i seguenti vantaggi:

- Indipendenza tra il business data (model), la logica di presentazione (view) e quella di controllo (controller);
- Separazione dei ruoli e delle relative interfacce;
- Viste diverse per il medesimo model;
- Semplice supporto per nuove tipologie di client.

8 Interfacce utenti

Il prodotto software finale disporrà di alcune GUI_[g] che permetteranno all'utente di interagire con il sistema RescueMe.

Le GUI descritte in questa sezione sono esclusivamente a scopo illustrativo.

8.1 GUI1: Pagina principale

La figura 30 a pagina 40 illustra la pagina iniziale di RescueMe. Da questa pagina l'utente potrà accedere a tre diverse funzionalità:

Login: l'utente accede al proprio pannello autenticandosi nel sistema;

Invio richiesta: l'utente, che sia autenticato o no, potrà inserire nel sistema una richiesta di soccorso;

Registrazione: l'utente si registra nel sistema.

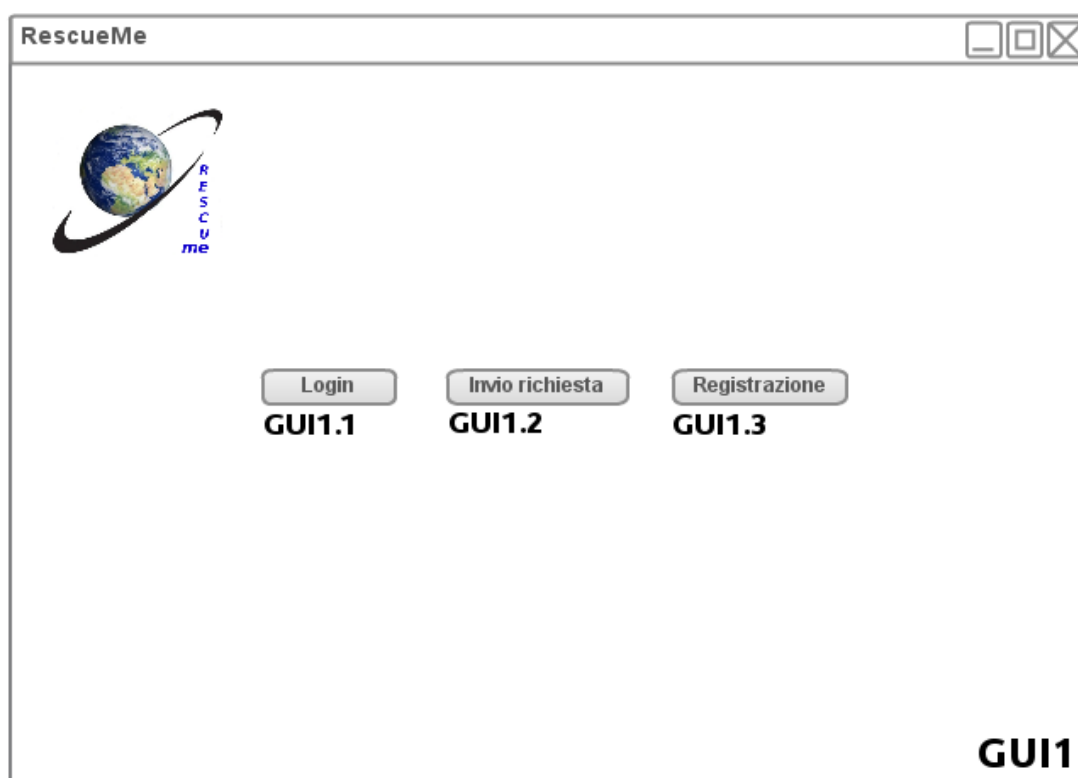


Figura 30: GUI1 - Home Page di RescueMe

8.1.1 GUI1.1: Autenticazione nel sistema

Attraverso l'interfaccia di figura 31 a pagina 41 l'utente potrà inserire le proprie credenziali e accedere al sistema. La form di autenticazione sarà uguale per tutti i tipi di utenti.

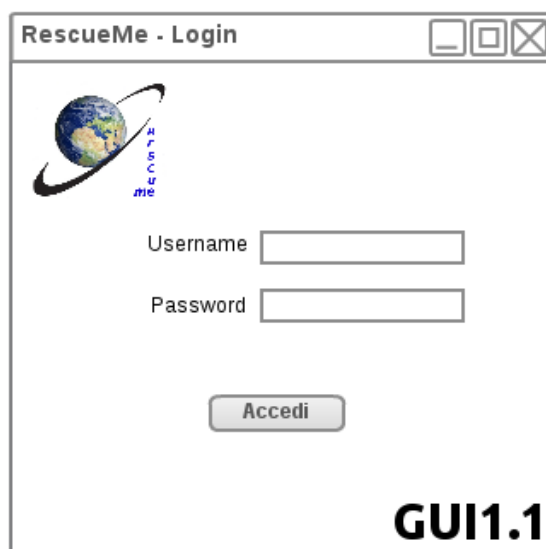


Figura 31: GUI1.1 - Pagina Autenticazione

8.1.2 GUI1.2: Invio richiesta

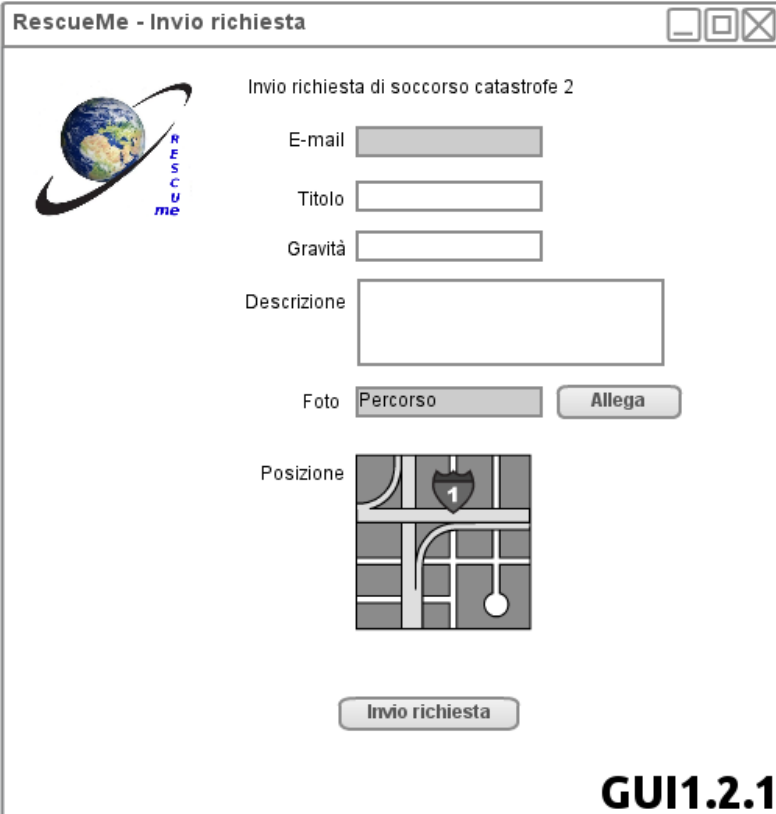
Per inviare una richiesta, vedi figura 32 a pagina 41, l'utente deve prima di tutto selezionare una catastrofe attiva nel sistema e successivamente scegliere la modalità di invio nel sistema: utente anonimo, utente registrato.



Figura 32: GUI1.2 - Pagina per l'invio della richiesta, primo passo

8.1.2.1 GUI1.2.1: Form di inserimento richiesta di soccorso

Nella figura 33 a pagina 42 è possibile vedere come verrà strutturata la form per l'inserimento dei dati relativi a una richiesta. Tale form verrà personalizzata con i campi più opportuni secondo il tipo di richiesta che l'utente vorrà inserire: richiesta di soccorso anonima oppure non anonima.

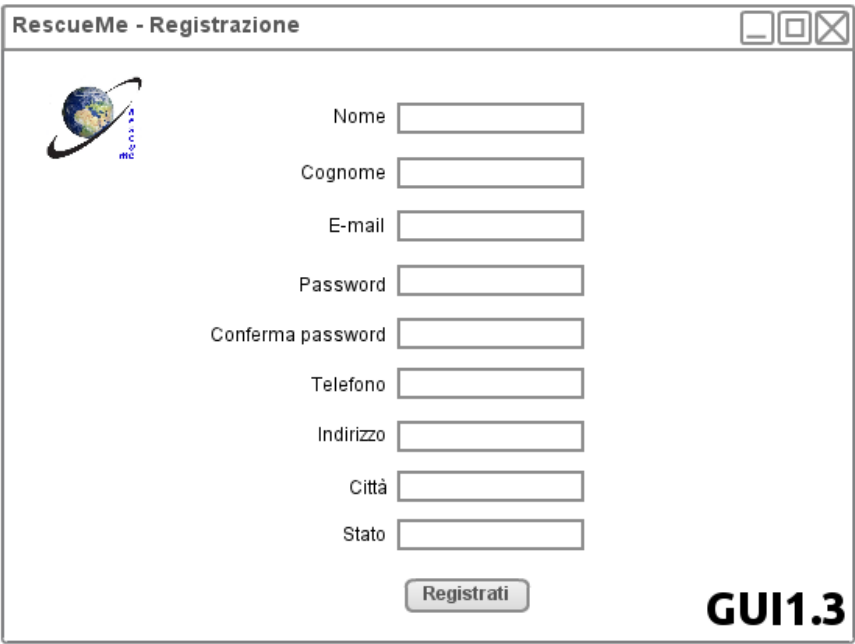


The screenshot shows a window titled "RescueMe - Invio richiesta". On the left is the RescueMe logo, which features a globe with a satellite orbit and the text "RESCUEme". The main content area is titled "Invio richiesta di soccorso catastrofe 2". It contains several input fields: "E-mail", "Titolo", "Gravità", and "Descrizione". Below these is a "Foto" section with a "Percorso" button and an "Allega" button. There is also a "Posizione" section with a map icon showing a road intersection and a shield with the number "1". At the bottom center is a button labeled "Invio richiesta". The text "GUI1.2.1" is displayed in the bottom right corner of the window.

Figura 33: GUI1.2.1 - Pagina per l'invio della richiesta, secondo passo

8.1.3 GUI1.3: Form di registrazione

La struttura della form per la registrazione di un utente nel sistema si può vedere nella figura 34 a pagina 42.



The screenshot shows a window titled "RescueMe - Registrazione". On the left is the RescueMe logo. The main content area contains a series of input fields for user registration: "Nome", "Cognome", "E-mail", "Password", "Conferma password", "Telefono", "Indirizzo", "Città", and "Stato". At the bottom center is a button labeled "Registrati". The text "GUI1.3" is displayed in the bottom right corner of the window.

Figura 34: GUI1.3 - Pagina per la registrazione

8.2 GUI2: Pannello segnalante

Il segnalante avrà a disposizione un pannello utente (vedi figura 35 a pagina 43) dove potrà accedere a tutte le funzionalità a lui dedicate tramite un menù posizionato alla sua sinistra. Nell'area di lavoro, l'utente vedrà i vari contenuti a seconda della voce del menù che seleziona.



Figura 35: GUI2 - Interfaccia web segnalante

8.3 GUI3: Pannello amministratore di sistema

L'interfaccia a cui accederanno gli amministratori di sistema è illustrata nella figura 36 a pagina 44. Come per il pannello segnalante (figura 35 a pagina 43), l'amministratore avrà un menù a sinistra e un'area di lavoro nel restante spazio dell'interfaccia.



Figura 36: GUI3 - Interfaccia web amministratore di sistema

8.4 GUI4: Pannello operatore, operatore avanzato e amministratore d'autorità

L'interfaccia per gli operatori, gli operatori avanzati e gli amministratori d'autorità è unica. Sarà il sistema di volta in volta, a seconda dei privilegi dell'utente, a visualizzare o nascondere le funzionalità a cui l'utente può o non può accedere (vedi figura 37 a pagina 44).

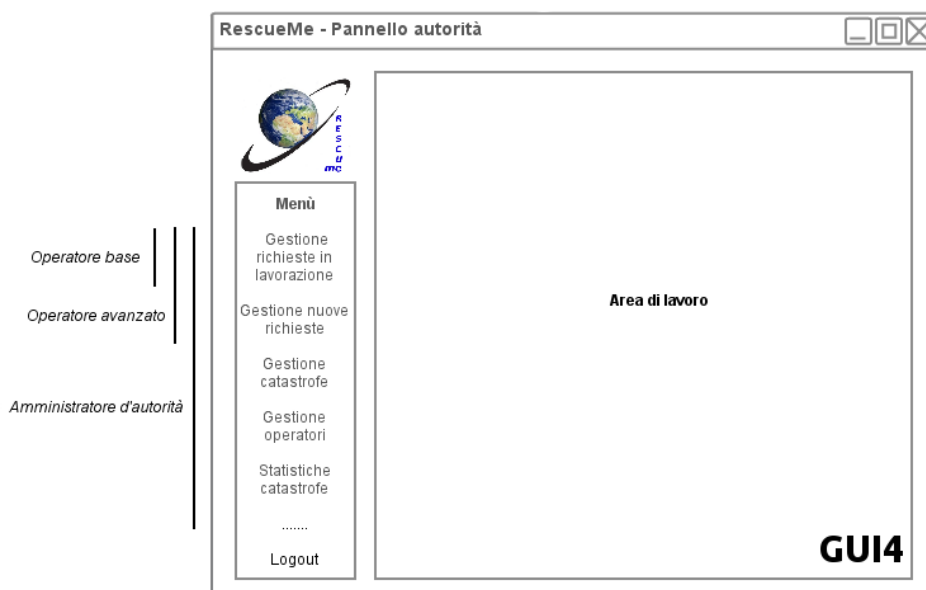


Figura 37: GUI4 - Interfaccia web operatore, operatore avanzato e amministratore d'autorità

8.5 GUI5: Applicazione Android

L'interfaccia dell'applicazione Android sarà struttura come la figura 38 a pagina 45. L'utente avrà nella parte inferiore dello schermo, dopo la pressione di un determinato pulsante nel terminale, il menù con le funzionalità accessibili dall'applicazione. L'area di lavoro sarà l'intero schermo del terminale quando non presente il menu stesso.



Figura 38: GUI5 - Interfaccia per l'applicazione smartphone

9 Stime di fattibilità

Il prodotto software RescueMe viene realizzato affidandosi alla tecnologia GAE come piattaforma cloud sulla quale far risiedere l'applicazione, mentre la persistenza dei dati viene garantita dal database Google DataStore, presente nella piattaforma succitata. RescueMe viene sviluppato utilizzando il linguaggio Java sfruttando l'IDE Eclipse. Usufruendo del Google Plugin per Eclipse verrà garantita la praticità di poter compilare il codice Java scritto e ottenere pagine dinamiche JSP_[g] che potranno essere importate direttamente in GAE. Google plugin per Eclipse integra, infatti, Google Web Toolkit il quale è un SDK che si occupa di eseguire il compito sopra descritto (vedi paragrafo 3.2.2 a pagina 6).

Per la creazione delle interfacce grafiche con GWT, se necessario, sarà utilizzato anche il plugin per Eclipse GWT Designer fornito da Google.

Per la strutturazione dell'applicazione l'azienda si appoggerà al framework Spring e l'integrazione di Spring Security per la gestione della sicurezza (autenticazione e autorizzazione degli utenti).

Nello sviluppo dell'applicazione Android viene utilizzato il SDK ufficiale fornito da Google.

La comunicazione tra l'applicazione Android e il sistema RescueMe avviene attraverso i servizi RESTful sviluppati utilizzando l'architettura REST, messa a disposizione dallo Spring Framework.

10 Tracciamento della relazione componenti - requisiti

Package	Componente
it.zetasolutions.rescueme.client.view	V1 MainMenuTreeViewModel V2 RescueMeShell V3 ContentWidget
it.zetasolutions.rescueme.client.view.widgets. admin.system	VWAS1 SystemAdminWidget
it.zetasolutions.rescueme.client.view.widgets. admin.authority	VWAA1 AuthorityAdminWidget
it.zetasolutions.rescueme.client.view.widgets. common	VWC1 CommonWidget
it.zetasolutions.rescueme.client.view.widgets. operator	VWO1 OperatorWidget
it.zetasolutions.rescueme.client.view.widgets. operator.advancedoperator	VWOA1 AdvancedOperatorWidget
it.zetasolutions.rescueme.client.view.widgets. victim	VWV1 VictimWidget
it.zetasolutions.rescueme.server.model	M1 AuthorityDisaster M2 Authority M3 OperatorAuthority M4 Disaster M5 OperatorDisaster M6 GenericUser M7 Address M8 Request M9 Comment M10 HistoricalChanges M11 PersistenceMgr
it.zetasolutions.rescueme.server.controller	C1 WebServiceImpl C2 ValidatorServiceImpl C3 GwtRpcController C4 ControllerUtilities C5 StatisticImpl
it.zetasolutions.rescueme.server.controller. authentication	SE-CA1 AuthService SE-CA2 CustomAuthenticationProvider SE-CA3 CustomAuthListener SE-CA4 CustomUserAuthentication
it.zetasolutions.rescueme.server.controller.rest	CR1 CommentController CR2 DisasterController CR3 GenericUserController CR4 RequestController
it.zetasolutions.rescueme.shared.controller. authentication	SH-CA1 IAuthService SH-CA2 IAuthServiceAsync
it.zetasolutions.rescueme.shared.controller. webservice	CW1 IWebService CW2 IWebServiceAsync
it.zetasolutions.rescueme.shared.controller. validator	CV1 IValidatorService CV2 IValidatorServiceAsync

CONTINUA SULLA PAGINA SUCCESSIVA

Package	Componente
it.zetasolutions.rescueme.shared.controller. statistic	CS1 IStatisticService CS2 IStatisticServiceAsync
it.zetasolutions.rescueme.android.view	AV1 ModifyInformationUser AV2 NewRequestForm AV3 Registration AV4 ViewListRequests AV5 ViewRequest
it.zetasolutions.rescueme.android.controller	AC1 InsertNewRequest AC2 UpdateRequest AC3 UserRegistration
it.zetasolutions.rescueme.android.model	AM1 Comment AM2 PersistenceMgrAndroid AM3 Request AM4 User

Tabella 1: Tracciamento package - componenti

Codice unificato	Componenti
AUTH	SE-CA1, SE-CA2, SE-CA3, SE-CA4, SH-CA1, SH-CA2
WEBS	CW1, CW2, C1
VAL	CV1, CV2, C2
STA	CS1, CS2, C5
WID	V1, V2, V3, VWC1

Tabella 2: Raggruppamento componenti atomici

Componenti	Requisiti
WID	FOB01, FOB02, FOB03, FOB04, FOB05, FOB06, FOB07, FOB08, FOB09, FOB10, FOB11, FOB12, FOB13, FOB14, FOB15, FOB16, FOB17, FOB19, FOB20, FDE01, FDE02, FDE03, FDE04, FDE05, FOP01, FOP02, FOP03, FOP04, VOB02, VOB03, VOB04
VWAS1	FOB05, FDE03, FOP04
VWAA1	FOB04, FOB09, FOB10, FOB11, FOB12, FOB13, FOB14, FOB15, FOB16, FOB20, FDE02, FOP03
VWO1	FOB03, FOB08, FDE01, FOP02, VOB03, VOB04
VWOA1	FOB08, FOB19, FOB20, FDE05
VWV1	FOB01, FOB07, FOP01, VOB02
C3	FOB01, FOB06, FOB07, FOB08, FOB09, FOB10, FOB11, FOB12, FOB13, FOB14, FOB15, FOB16, FOB19, FOB20, FDE01, FDE02, FDE03, FDE04, FDE05, FOP01, FOP02, FOP03, FOP04, VOB03, VOB04
C4	FOB01, FOB06
M1	FOB09, FOB11, FOB14, FOB16, FOB19, FOB20, FDE02, FOP03
M2	FOB09, FOB11, FOB13, FOB14, FOB15, FOB16, FOB19, FOB20, FDE02, FDE03, FOP03, FOP04
M3	FOB11, FOB13, FOB14, FOB15, FOB19, FOB20
M4	FOB01, FOB07, FOB09, FOB10, FOB11, FOB12, FOB14, FOB16, FOB20, FDE02, FDE03, FOP02, FOP03, FOP04, VOB03, VOB04

CONTINUA SULLA PAGINA SUCCESSIVA

Componenti	Requisiti
M5	FOB08, FOB19, FOP02
M6	FOB01, FOB06, FOB07, FOB08, FOB09, FOB10, FOB11, FOB12, FOB13, FOB14, FOB15, FOB16, FOB19, FOB20, FDE01, FDE02, FDE03, FDE05, FOP02, FOP03, FOP04
M7	FOB01, FOB06, FOB13
M8	FOB01, FOB07, FOB08, FOB11, FOB20, FDE01, FDE02, FDE03, FDE04, FDE05, FOP01, FOP02, FOP03, FOP04, VOB03, VOB04
M9	FOB07, FOB08, FDE01, FOP01
M10	FOB10, FOB20, FDE01, QOB04
M11	FOB01, FOB06, FOB07, FOB08, FOB09, FOB10, FOB11, FOB12, FOB13, FOB14, FOB15, FOB16, FOB19, FOB20, FDE01, FDE02, FDE04, FDE05, FOP01, FOP02, FDE03, FOP03, FOP04, VOB03, VOB04
AUTH	FOB01, FOB07, FOB08, FOB09, FOB10, FOB11, FOB12, FOB13, FOB14, FOB15, FOB16, FOB19, FOB20, FDE01, FDE02, FDE03, FDE05, FOP03, FOP04
CR1	FOB07
CR2	FOB01
CR3	FOB06
CR4	FOB01, FOB07
WEBS	FOB01, FOB06, FOB07, FOB08, FOB09, FOB10, FOB11, FOB12, FOB13, FOB14, FOB15, FOB16, FOB19, FOB20, FDE01, FDE02, FDE03, FDE04, FDE05, FOP01, FOP02, FOP03, FOP04, VOB03, VOB04
VAL	FOB01, FOB06, FOB10, FOB12, FOB13, FOB14, FOB15, FOB16, FOB20, FDE01, FOP01
STA	FDE02, FDE03, FOP03, FOP04
AV1	VOB02
AV2	FOB01, FDE04, FOP01, VOB02
AV3	FOB06, FOB18, VOB02
AV4	FOB07, VOB02
AV5	FOB07, FOP01, VOB02
AC1	FOB01, FDE04, FOP01
AC2	FOP01
AC3	FOB06, FOB18
AM1	FOB07, FOP01
AM2	FOB01, FOB06, FOB07, FOB18, FDE04, FOP01
AM3	FOB01, FOB07, FDE04, FOP01
AM4	FOB01, FOB06, FOB18

Tabella 3: Tracciamento Componenti - Requisiti

Requisiti	Componenti
FOB01	WID, VWV1, M4, M6, M7, M8, M11, AUTH, WEBS, VAL, C3, C4, CR2, CR4, AV2, AC1, AM2, AM3, AM4
FOB02	WID, VWV1
FOB03	WID, VWO1
FOB04	WID, VWVAA1
FOB05	WID, VWAS1
FOB06	WID, WEBS, C3, VAL, C4, CR3, M6, M7, M11, AV3, AC3, AM2, AM4

CONTINUA SULLA PAGINA SUCCESSIVA

Requisiti	Componenti
FOB07	WID, VWV1, AUTH, WEBS, C3, CR1, CR4, M4, M6, M8, M9, M11, AV4, AV5, AM1, AM2, AM3
FOB08	WID, AUTH, VWOA1, VWO1, WEBS, C3, M5, M6, M8, M9, M11
FOB09	WID, AUTH, VWAA1, WEBS, C3, M1, M2, M4, M6, M11
FOB010	WID, AUTH, VWAA1, WEBS, VAL, C3, M4, M6, M10, M11
FOB011	WID, VWAA1, AUTH, WEBS, C3, M1, M2, M3, M4, M6, M8, M11
FOB012	WID, VWAA1, AUTH, WEBS, C3, VAL, M4, M6, M11
FOB013	WID, VWAA1, AUTH, WEBS, C3, VAL, M2, M3, M6, M7, M11
FOB014	WID, VWAA1, AUTH, WEBS, C3, VAL, M1, M2, M3, M4, M6, M11
FOB015	WID, VWAA1, AUTH, WEBS, VAL, C3, M2, M3, M6, M11
FOB016	WID, VWAA1, AUTH, WEBS, VAL, C3, M1, M2, M4, M6, M11
FOB017	WID, WEBS, VAL, C3, M4, M8, M11
FOB018	AV3, AC3, AM2, AM4
FOB19	WID, VWOA1, AUTH, WEBS, VAL, C3, M1, M2, M3, M5, M6, M11
FOB20	WID, VWAA1, VWOA1, C3, M1, M2, M3, M4, M6, M8, M10, M11, AUTH, WEBS, VAL
FDE01	WID, VWO1, AUTH, WEBS, VAL, C3, M6, M8, M9, M10, M11
FDE02	WID, VWAA1, AUTH, WEBS, STA, C3, M1, M2, M4, M6, M8, M11
FDE03	WID, VWAS1, AUTH, WEBS, STA, C3, M2, M4, M6, M8, M11
FDE04	WID, WEBS, C3, M8, M11, AV2, AC1, AM2, AM3
FDE05	WID, VWOA1, AUTH, WEBS, C3, M6, M8, M11
FOP01	WID, VWV1, WEBS, VAL, C3, M8, M9, M11, AV2, AV5, AC1, AC2, AM2, AM3, AM1
FOP02	WID, VW01, WEBS, C3, M4, M5, M6, M8, M11
FOP03	WID, VWAA1, AUTH, WEBS, STA, C3, M1, M2, M4, M6, M8, M11
FOP04	WID, VWAS1, AUTH, WEBS, STA, C3, M2, M4, M6, M8, M11
POB01	Si utilizzerà il servizio cloud di Google al fine di soddisfare questo requisito (vedi parte di Google App Engine 2.1 a pagina 3)
PDE01	Si presterà attenzione a questo aspetto durante la programmazione dell'applicazione Android seguendo le direttive fornite da Google (Design for performance: http://developer.android.com/guide/practices/design/performance.html)
PDE02	Si presterà attenzione a questo aspetto durante la realizzazione dell'interfaccia mobile
QOB01	Si presterà attenzione a questo aspetto durante la realizzazione delle interfacce grafiche. Il soddisfacimento del requisito verrà verificato con dei test mirati.
QOB02	Si presterà attenzione a questo aspetto durante la realizzazione delle interfacce Web e dell'applicazione per Android. L'applicazione Android salverà dei dati utente localmente al fine di non generare traffico di rete per reperirli. Il soddisfacimento del requisito verrà verificato con dei test mirati.
QOB03	Si presterà attenzione a questo aspetto durante la programmazione del software. Il soddisfacimento del requisito verrà verificato con dei test mirati.
QOB04	M10
QOB05	Si utilizzerà il servizio cloud di Google al fine di soddisfare questo requisito (vedi parte di Google App Engine 2.1 a pagina 3). Il software avrà la possibilità di inserire lingue diverse al fine di permettere il suo utilizzo a popolazioni con diverse lingue parlate.
QOB06	L'azienda consegnerà in sede di RA un manuale di installazione e i manuali utenti divisi per tipologia d'utente.
QDE01	Le interfacce Web e l'applicazione Android implementeranno un help contestuale. Sarà disponibile un manuale online per la definizione di tutti gli errori.

CONTINUA SULLA PAGINA SUCCESSIVA

Requisiti	Componenti
QOP1	Tutte le interfacce web e le GUI per smartphone Android avranno una versione bilingue italiano/inglese.
VOB01	AUTH - Il soddisfacimento del requisito verrà verificato con dei test mirati.
VOB02	WID, VWV1, AV1, AV2, AV3, AV4, AV5
VOB03	WID, VWO1, WEBS, C3, M4, M8, M11
VOB04	WID, VWO1, WEBS, C3, M4, M8, M11
VOB05	Vincolo tecnologico implementato dal software.
VOB06	Vincolo tecnologico implementato dal software.
VOB07	Vincolo tecnologico implementato dal software.
VOB08	Vincolo tecnologico implementato dal software.
VOB09	Vincolo tecnologico implementato dal software.
VOB10	Vincolo tecnologico implementato dal software.
VOB11	L'azienda presenterà in sede di RA la documentazione richiesta in tecnologia JavaDoc
VOB12	L'azienda presenterà in sede di RA l'applicazione per dispositivi Android.
VOP01	Parte del requisito viene soddisfatto dai componenti CR1, CR2, CR3 e CR4 per l'interfacciamento dell'applicazione Android gestita come un sistema esterno a RescueMe e quindi utilizzabile dalle autorità per l'interfacciamento con i propri sistemi. Non è stata prevista l'architettura per la gestione completa del prodotto via REST, in quanto l'azienda non soddisferà il requisito se non esplicitamente richiesto dal committente.

Tabella 4: Tracciamento Requisiti - Componenti