Relazione Progetto

SCISCI VALERIO
IEZZI JACOPO
ABBADINI LORENZO
FLORIO MICHELE

Gestione dei contratti d'appalto tramite Smart Contract e Blockchain Ethereum

Università Politecnica delle Marche
Dipartimento di Ingegneria

Gruppo:

Indice

1 Introduzione

Questa è la relazione atta ad esporre al lettore i processi che sono stati necessari per la realizzazione del progetto Smart Contract.

Il progetto Smart Contract (SC) consiste nella realizzazione di un'applicazione che permetta agli utenti di un contratto di appalto di poter gestire i documenti che lo compongono (Libretto delle Misure, Registro di Contabilità, Stato Avanzamento Lavori, Giornale dei Lavori) in maniera digitale.

Tale risultato è stato raggiunto grazie all'utilizzo di un'interfaccia grafica per l'interazione utente e all'utilizzo di una blockchain per la conferma della validità delle operazioni svolte e per memorizzare lo stato di avanzamento dei lavori.

Per tutta la durata della realizzazione del progetto sono stati utilizzati i concetti studiati in Ingegneria del Software e per tale motivo sono stati realizzati i vari diagrammi necessari alla corretta esecuzione del progetto.

Per questo motivo le parti successive descrivono le varie fasi dello sviluppo del progetto esaminate nel corso.

La sezione 2 è relativa alla descrizione del **project management:** definizione dell'ambito, individuazione degli obiettivi, creazione di un piano e sviluppo.

Nella sezione 3 viene illustrato il processo di raccolta e classificazione dei requisiti (storie utente) e nella sezione 4 questi vengono analizzati.

La sezione 5 mostra la progettazione delle interfacce, mockup e architettura del sistema.

Nella sezione 6 sono descritte le tecnologie adottate nella implementazione del sistema.

Nella sezione 7 sono mostrati diversi indici necessari alla valutazione del sistema, inoltre vengono mostrati anche i processi di collaudo del sistema, come test su dati non corretti, verifiche di correttezza.

Infine, nella sezione 8 è presente la conclusione in cui vengono avanzate delle proposte per un possibile sviluppo futuro di questo progetto.

2 Pianificazione

2.1 Definizione dell'ambito

Questo progetto si colloca nell'ambito della gestione dei documenti dei contratti di appalto di una stazione appaltante:

1. Libretto delle misure;
2. Giornale dei lavori;
3. Registro di contabilità;
4. Stato avanzamento dei lavori.

Per una descrizione più precisa e dettagliata si consulti il Project Scoping Form qui sotto riportato:

Project Scoping Form	
Progetto: Smart Contract	**ri del Team:** le Florio, Valerio Scisci, Jacopo Iezzi, Lorenzo Abbadini
Stato del problema/Opportunità: Non esiste ancora un software in grado di trascrivere un contratto d'appalto (in ambito edilizio) sotto forma di Smart Contract utilizzando le Blockchain. Un contratto d'appalto è abbastanza complesso (prevede qualcuno che fa un lavoro per un altro, per una certa durata di tempo) e si basa su quelli che vengono chiamati "Stati di Avanzamento"; ogni volta che lo stato di avanzamento supera la soglia prefissata dalla Stazione Appaltante viene pagato un corrispettivo alla ditta appaltatrice.	
Goal del progetto: Digitalizzazione dei contratti d'appalto utilizzando gli Smart Contract e le Blockchain.	
Obiettivi del progetto: 1) Definire le interfacce utente che permettono ai tre attori di compiere le relative azioni; (Compiere queste azioni significa andarle a scrivere sulla blockchain) 2) Stabilire come è fatto uno Smart Contract; (di solito si lavora con più Smart Contract collegati tra di loro) sare quali sono le transazioni che devono essere registrate e che fanno avanzare lo stato della computazione dello Smart Contract.	
i di successo:	

vare deve permettere la registrazione delle misure dei lavori svolti ed una corretta gestione dello Stato di Avanzamento dei Lavori garantendo inoltre l'invio delle notifiche di pagamento qualora la soglia prefissata per il pagamento sia stata superata.
ed ostacoli: ile tentativo di alterare le misurazioni registrate.

2.2 Sviluppo del piano

Nell sviluppo del processo sono stati combinati modelli di processo incrementali, evolutivi e concorrenti. In più è stata scelta una strategia mista tra quella plan-driven e quella delle metodologie agili. Le attività più generali sono state definite in anticipo, il progetto possiede una scadenza programmata e il progresso totale del progetto è proporzionale a quante di queste attività sono state completate. Comunque si è data la priorità alle modifiche e ai cambiamenti delle caratteristiche del progetto imposti dagli stakeholder durante le interviste e gli incontri tenutesi con questi ultimi.

2.3 Matrice delle responsabilità

Le persone del gruppo che lavorano per la realizzazione del progetto possono essere considerate alla pari di risorse, per questo risulta fondamentale, come per qualsiasi altra risorsa, la loro corretta gestione. Ed è per questo motivo che si è realizzata la seguente matrice delle responsabilità che descrive i vari compiti necessari al completamento del progetto e a quali membri del gruppo tali compiti sono stati assegnati.

Responsabilità	**chele**	**lerio**	**copo**	**renzo**
intervista con gli stakeholder				
da intervista con gli stakeholder	X	X	X	X
intervista con gli stakeholder	X			
zione User Stories	X	X	X	X

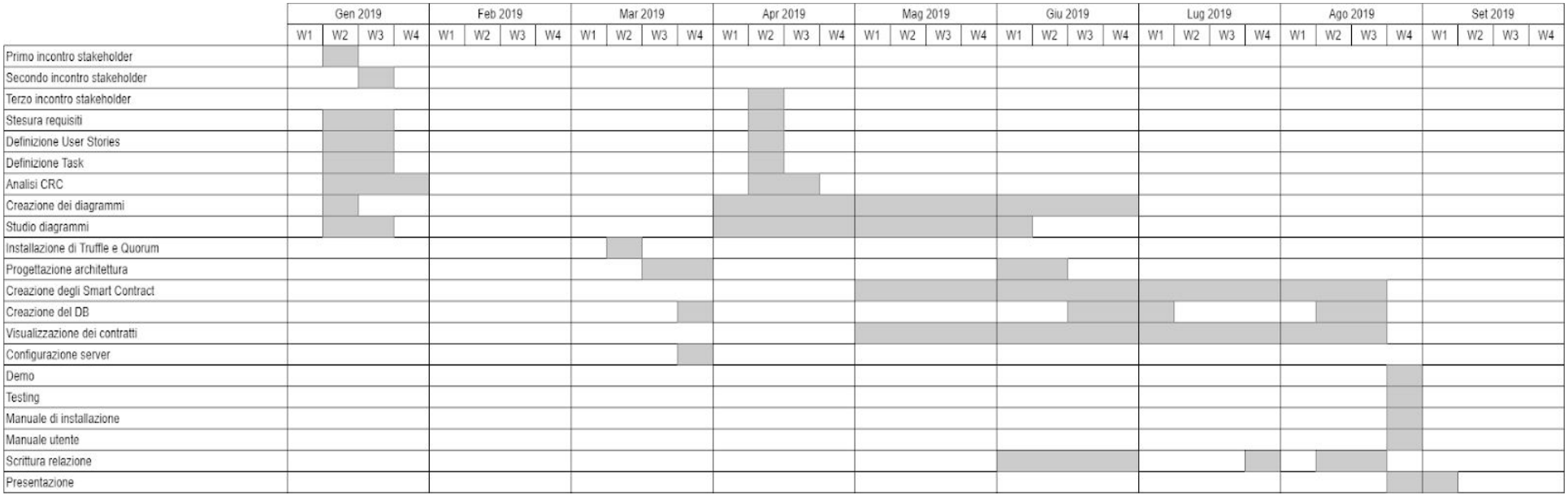
zione dei Task	X	X	X	X
i CRC	X	X	X	X
one Diagramma dei Casi D'uso	X	X	X	X
one del Diagramma delle Classi	X			X
one Mockup	X	X		X
one Modelli di Navigazione	X			X
one Modelli di Presentazione	X		X	X
one Modelli di Dialogo	X			X
one Modello dei Package	X			X
one Modello delle Collaborazioni	X			X
one Modello delle Sequenze	X			X
one Modello degli Stati	X			X
one Modello delle Attività	X			X
one Modello dei Componenti	X	X	X	X
azione di Truffle e Quorum		X	X	
tazione architettura		X	X	

one degli Smart Contract		X	X	
one del DB		X	X	
izzazione dei contratti		X	X	
gurazione server		X	X	
		X	X	
g		X	X	
le di installazione	X	X	X	X
ra relazione	X			X
tazione	X			

2.4 Considerazioni sul modello applicato

Come precedentemente accennato l'approccio adottato è un misto tra la metodologia plan-driven e quella agile. Infatti, le attività sono state pianificate in anticipo come mostrato nel diagramma di Grantt precedente. Nel corso dello sviluppo del progetto però si è mantenuto comunque un approccio agile cercando di rispettare il più possibile i principi dell'Extreme Programming. Prima di tutto creando un dialogo regolare con gli stakeholders tramite incontri faccia a faccia, in modo da esporre i problemi avuti durante la realizzazione del progetto in modo da accordarci per una soluzione, ma anche e soprattutto per poter ricevere richieste di cambiamento e di implementazione di nuove funzionalità. La priorità è stata data alla realizzazione di una prima release funzionante anche se con solo le funzionalità basilari. L'avanzamento del progetto è stato misurato proporzionalmente al software funzionante realizzato. Inoltre, si è cercato di mantenere un carico di lavoro per i componenti del gruppo in base ai loro impegni universitari e di lavoro in modo da permette un avanzamento costante di tutti i progetti assegnati a ciascun membro. Il modello di processo è incrementale e suddiviso in iterazioni che possono essere espanse in base alle richieste del committente, inoltre è evolutivo in quanto il codice già realizzato nelle release precedenti viene

costantemente aggiornato ed infine è concorrente perché lo sviluppo di alcuni task è stato eseguito in parallelo da diversi membri del gruppo.


Gen 2019
Feb 2019
Mar 2019
Apr 2019
Mag 2019
Giu 2019
Lug 2019
Ago 2019
Set 2019
W1 W2 W3 W4
Primo incontro stakeholder
Secondo incontro stakeholder
Terzo incontro stakeholder
Stesura requisiti
Definizione User Stories
Definizione Task
Analisi CRC
Creazione dei diagrammi
Studio diagrammi
Installazione di Truffle e Quorum
Progettazione architettura
Creazione degli Smart Contract
Creazione del DB
Visualizzazione dei contratti
Configurazione server
Demo
Testing
Manuale di installazione
Manuale utente
Scrittura relazione
Presentazione

3 Raccolta dei requisiti

3.1 Definizione dell'ambito

Innanzitutto, sono state identificate le fonti che possono essere divise in tre categorie: stakeholder, documentazioni esistenti e soluzioni preesistenti.

- **Stakeholder**: la raccolta dei requisiti con gli stakeholder sono avvenute tramite incontri di persona con i committenti del progetto. Gli stakeholder intervistati sono il prof. Luca Spalazzi e il prof. Berardo Naticchia. Gli incontri si sono tenuti all'interno della struttura di ingegneria in diverse sessioni. Ad ogni incontro sono stati mostrati i progressi realizzati e sono stati affrontati i dubbi sorti durante la realizzazione del progetto ed inoltre sono state modificate o aggiunte specifiche al progetto in corso. Durante il primo incontro sono stati definiti gli utenti, il loro ruolo all'interno dello SC insieme con il loro grado di interazione con il sistema, è stato definito inoltre l'obiettivo del progetto e le condizioni che permettono al sistema di far avanzare lo stato di completamento dello SC.

- **Documentazioni esistenti**: è stata illustrata una breve presentazione che descrive in maniera generale le specifiche del progetto a cura di Francesco Spegni e Luca Spalazzi. Sono stati consultate le normative e i documenti riguardanti i contratti di appalto in modo particolare si è prestata attenzione alla rappresentazione disponibile on-line dei vari documenti facenti parte del contratto di appalto e del nostro progetto.

- **Soluzioni preesistenti:** L'unico approccio preesistente alla creazione di uno smart contract per la gestione di contratto di appalto proviene dalla tesi di un nostro compagno di corso, il quale illustra le tecnologie utilizzate per la creazione delle Blockchain e degli smart contract accompagnati da esempi di applicazione nell'ambito della gestione di un contratto d'appalto.

Dopo questa prima di identificazione delle fonti e dopo aver partecipato al primo incontro con gli stakeholder si è concluso che lo scopo del progetto sia quello di realizzare un programma che permetta, tramite interfaccia utente, la creazione di contratti di appalto e il successivo inserimento delle scritture dei lavori effettuati. Inoltre, tramite automatismi il programma deve essere in grado di aggiornare automaticamente tutti i documenti che compongono un contratto d'appalto, monitorando di volta in volta lo stato di avanzamento dei singoli lavori e il totale complessivo di completamento dell'appalto. In più, ad ogni superamento della soglia fissata dalla Stazione Appaltante all'inizio dell'inserimento del contratto di appalto, la Stazione Appaltante deve ricevere una notifica di pagamento nei confronti della Ditta Appaltatrice. Il sistema poi deve essere implementato utilizzando gli SC e le Blockchain, deve, inoltre, essere in grado di interfacciarsi con un server virtuale per l'implementazione del codice.

3.2 Classificazione dei requisiti

3.2.1 Requisiti utente

Esistono tre utenti: Stazione Appaltante, Direttore dei Lavori e la Ditta Appaltatrice ognuno di essi ricopre un ruolo fondamentale nel contratto d'appalto.

La Stazione Appaltante è l'utente che crea un nuovo contratto d'appalto specificando sia chi ricopre gli altri due ruoli del contratto, sia i lavori e rispettivi costi necessari per il completamento dell'opera appaltata, sia la soglia stabilita per la notifica di pagamento della Ditta Appaltatrice. Inoltre, la Stazione Appaltante ha il compito di confermare sia le scritture del Libretto delle Misure sia quelle del Registro di Contabilità, in modo da far progredire lo stato di avanzamento dei lavori.

Il Direttore dei Lavori è una figura scelta dalla Stazione Appaltante nel momento dell'inserimento di un nuovo contratto di appalto, il suo compito è quello di inserire sia le misure dei lavori effettuati nel Libretto delle Misure (con l'eventuale inserimento della riserva), sia le voci all'interno del Giornale dei Lavori.

Infine, la Ditta Appaltatrice è la ditta alla quale la Stazione Appaltante ha deciso di rivolgersi per il completamento dell'opera appaltata in quanto vincitrice dell'appalto. All'interno del sistema la Ditta Appaltatrice risulta essere un mero osservatore senza alcuna capacità di modifica diretta dello SC.

Tutti e tre gli utenti hanno inoltre la capacità di visualizzare lo stato di avanzamento dei lavori insieme con il contenuto di tutti i documenti correlati ad uno specifico SC.

3.2.2 Requisiti di sistema

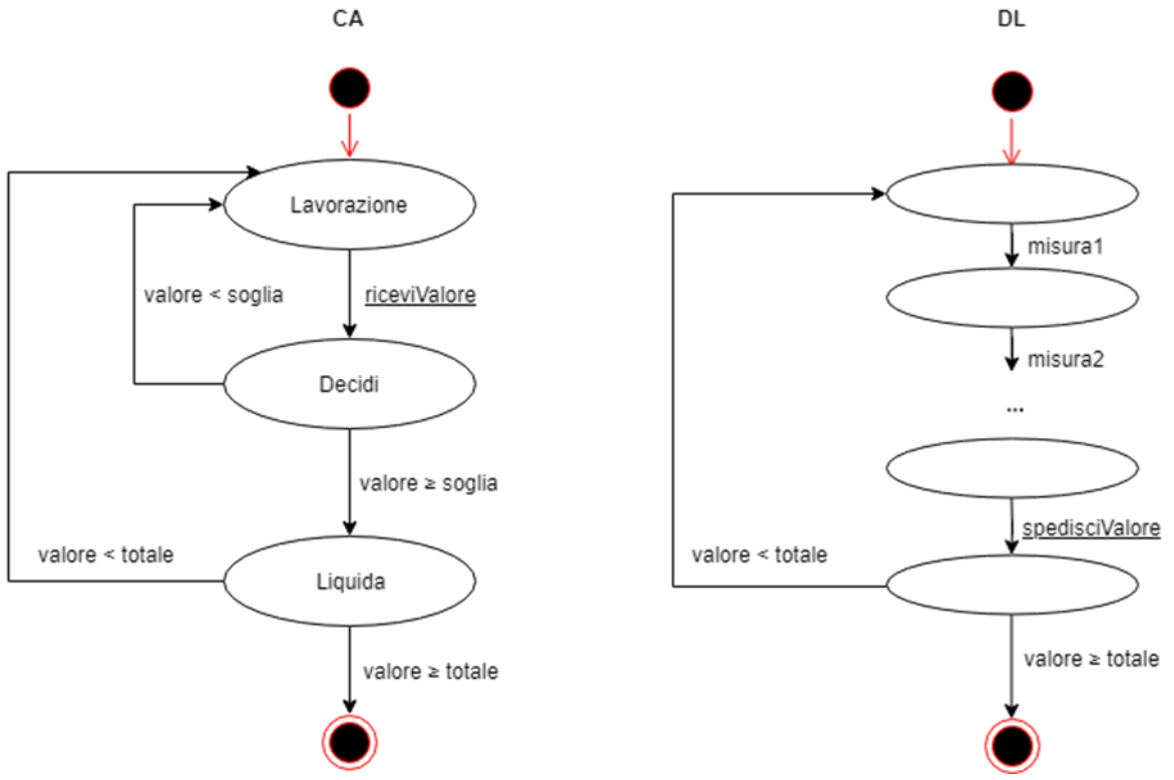
Requisiti funzionali:

1. Il sistema deve mostrare lo stato di avanzamento del contratto d'appalto;
2. Il sistema deve mostrare il Libretto delle Misure;
3. Il sistema deve mostrare il Giornale dei Lavori;

4. Il sistema deve mostrare il Registro di Contabilità;
5. Il sistema deve mostrare lo Stato di Avanzamento dei Lavori;
6. Il sistema deve permettere l'inserimento di nuove voci nel Libretto delle Misure;
7. Il sistema deve permettere l'inserimento di nuove voci nel Giornale dei Lavori;
8. Il sistema deve aggiornare automaticamente il Registro di Contabilità;
9. Il sistema deve aggiornare automaticamente lo Stato di Avanzamento dei Lavori;
10. Il sistema deve inviare la notifica di pagamento alla Stazione Appaltante quando il totale dei lavori compiuti supera la soglia prestabilita;
11. Il sistema deve permettere l'inserimento dell'eventuale riserva sulle misure.

Requisiti non funzionali:

1. Il sistema deve essere implementato usando Quorum Blockchain e gli SC;
2. Il sistema deve possedere un'interfaccia grafica per interfacciarsi con l'utente;
3. Il codice del sistema deve essere implementato su un server virtuale ed eventualmente è possibile effettuare il deploy del codice sul server dell'università.



I seguenti diagrammi a stati forniscono una visione semplificata degli Smart Contract che modellano il Contratto di Appalto (CA) ed il Direttore dei Lavori (DL).

3.2.3 Costruzione del glossario

Finora i requisiti sono stati espressi in linguaggio naturale. Nelle user stories invece il linguaggio utilizzato è naturale semi-strutturato. Per questo viene costruito un glossario che riassume i termini introdotti nella raccolta, fornendone una breve descrizione:

Termine	Descrizione	onimi e tipi specifici	Collegamenti
rettore dei Lavori	ersona incaricata di effettuare le misure sui lavori compiuti in cantiere e scrivere i	DL, Utente	ra, password, Libretto Misure, Giornale dei Lavori, autenticazione, Ditta

	risultati nel Libretto delle Misure e nel Giornale dei Lavori		Appaltatrice, Stazione Appaltante
itta Appaltatrice	enda che si occupa di eseguire i lavori stabiliti nel contratto d'appalto dietro remunerazione	DA, Utente	ore dei Lavori, Stazione Appaltante, password, notifica, lavoro, compenso, autenticazione
azione Appaltante	esa che ha stipulato il contratto d'appalto con un'azienda per la realizzazione di un'opera	SA, Utente	, Direttore dei Lavori, Ditta Appaltatrice, password, autenticazione, pagamento
Login	ocesso che permette l'autenticazione degli utenti	Autenticazione	e, username, password
Smart Contract	rtualizzazione di un contratto d'appalto su una Blockchain	SC	ockchain, contratto d'appalto
ontratto d'appalto	alto è il contratto con il quale una parte assume, con organizzazione dei mezzi necessari e con gestione a proprio rischio, il compimento di un'opera o di un servizio verso un corrispettivo in danaro	Appalto	rt Contract, Stazione Appaltante, Ditta Appaltatrice, Direttore dei Lavori
Blockchain	ttura dati condivisa e immutabile definita come un registro digitale le cui voci sono raggruppate in blocchi concatenati in ordine cronologico	hereum Blockchain	art Contract, misure
oretto delle Misure	hento che contiene tutte le misure dei lavori,	LM, Documento	ale dei Lavori, Registro di Contabilità, Stato

	misurati dal Direttore dei Lavori		Avanzamento Lavori, Direttore dei Lavori
iornale dei Lavori	)ocumento con cui monitorare passo passo l'andamento tecnico ed economico di un'opera	GL, Documento	gistro di Contabilità, Libretto delle Misure, Stato Avanzamento Lavori, Direttore dei Lavori
gistro di Contabilità	umento contabile che riassume ed accentra l'intera contabilizzazione dell'opera	RC, Documento	bretto delle Misure, Giornale dei Lavori, Stato Avanzamento Lavori
Avanzamento Lavori	umento funzionale al pagamento della Ditta Appaltatrice; riassume tutti i lavori eseguiti dall'inizio dell'appalto e il loro avanzamento rispetto al totale	SAL, Documento	bretto delle Misure, Giornale dei Lavori, Registro di Contabilità
tifica di Pagamento	ca diretta alla Stazione Appaltante come promemoria del pagamento dovuto nei confronti della Ditta Appaltatrice	Notifica	one Appaltante, Stato Avanzamento Lavori
lia per il pagamento	a fissa dell'importo dei lavori eseguiti stabilita per l'invio della Notifica di Pagamento	Soglia	ca di Pagamento, Stato Avanzamento Lavori, Stazione Appaltante
Riserva	serva su una misura effettuata e registrata dal Direttore dei Lavori, richiesta dalla Ditta Appaltatrice per discordanza sulla misura del lavoro effettuato		, Libretto delle Misure, Direttore dei Lavori, Ditta Appaltatrice

3.2.4 Storie utente

Nella fase di exploration per poter proseguire con Release Planning Game sono state compilate 5 Story Card in cui vengono descritte le funzionalità che devono essere implementate nel programma. Di seguito sono riportate le storie utente:

Release:	Story ID: 2	Release:
Story Tag: Visualizzazione documenti fondamentali		Priority: 1
Author: Lorenzo	on: 21/03/19	Accepted:

Description:

L'applicazione, attraverso l'interfaccia, dovrà mostrare alcuni documenti che permettono di visualizzare lo stato dello SmartContract e lo storico dei lavori, questo storico è rappresentato dalle voci che costituiscono i seguenti documenti:

- **Libretto delle Misure;**
- **Registro delle Contabilità;**
- **Giornale dei Lavori;**
- **Stato Avanzamento dei Lavori (SAL).**

Release:	Story ID: 3	Release:
Story Tag: Inserimento di nuove voci nei documenti		Priority: 1
Author: Michele	on: 19/11/18	Accepted:

Description:

Attraverso l'interfaccia il Direttore dei Lavori può inserire i valori delle misurazioni dei lavori effettuati (eventualmente inserendo anche la riserva per la misura registrata), così da permettere l'avanzamento dei lavori e sempre attraverso l'interfaccia il Direttore dei Lavori può inserire nuove voci nel Giornale dei Lavori.

Release:	Story ID: 4	Release:
Story Tag: Creazione dello Smart Contract		Priority: 4
Author: Valerio	on: 19/11/18	Accepted:

Description:

La Stazione Appaltante crea un nuovo Smart Contract indicando i soggetti coinvolti (la Ditta Appaltatrice ed il Direttore dei Lavori) ed i parametri relativi al contratto d'appalto (come i lavori da effettuare con i relativi costi previsti e la soglia per l'invio della notifica di pagamento), precedentemente concordati con i soggetti indicati.

Release:	Story ID: 5	Release:
Story Tag: Logica di funzionamento dell'applicazione		Priority: 2
Author: Jacopo	on: 19/11/18	Accepted:

Description:

L'applicazione dovrà prevedere meccanismi di verifica dello stato di completamento dell'opera (come ad esempio la conferma da parte della Stazione Appaltante ad ogni scrittura inserita, necessaria per l'avanzamento dei lavori) ; in particolare, al raggiungimento di soglie prestabilite verrà notificato un avanzamento nel completamento dell'opera e verrà segnalata la necessità di liquidare un pagamento nei confronti della Ditta Appaltatrice; inoltre l'applicazione dovrà permettere l'inserimento delle riserve sulle misure registrate, in modo che le misure con riserva non facciano avanzare lo stato dei lavori, ma siano comunque registrate nel Libretto delle Misure.

3.2.5 Costruzione delle Task Card

Nel Release Planning Game di volta in volta viene selezionata una story card che dovrà essere sviluppata. La fase di sviluppo di questa story card è iterativa, suddividendosi in:

1. Selezione della storia utente da sviluppare;
2. Suddivisione della storia in task card;
3. Nell'Iteration Planning Game vengono scelte di volta in volta le task card da implementare;
4. Progettazione e implementazione del task scelto;
5. Rilascio;
6. Integrazione.

Di seguito sono mostrate tutte le task card generate a partire dalle singole storie utente:

Storia 1

Release:	**Story ID: 1**	**Task ID: 1.1**
Task Tag: Interfaccia Inserimento Dati login		**Release:**
Software Engineer: Valerio		
Description: **Implementare l'interfaccia grafica (due form per l'inserimento dell'username e password) necessaria all'utente per poter effettuare il login.**		

Release:	Story ID: 1	Task ID: 1.2
Task Tag: Creazione del Mockup		Release:
Software Engineer: Lorenzo		

Description:

Creazione del mockup che illustra l'interfaccia del login

Release:	Story ID: 1	Task ID: 1.3
Task Tag: Test di sicurezza		Release:
Software Engineer: Valerio		

Description:

Test sulla sicurezza d'accesso al programma

Release:	Story ID: 1	Task ID: 1.4
Task Tag: Creazione documentazione		Release:
Software Engineer: Michele		

Description:

Creazione dei modelli necessari utili all'analisi e alla progettazione (CRC, Class Diagram, Diagramma di Stato, etc.)

Storia 2

Release:	Story ID: 2	Task ID: 2.1
Task Tag: Creazione interfaccia del Libretto delle Misure		Release:
Software Engineer: Michele		

Description:

Creazione dell'interfaccia del documento Libretto delle Misure

Release:	Story ID: 2	Task ID: 2.2
Task Tag: Creazione interfaccia del Giornale dei Lavori		Release:
Software Engineer: Michele		

Description:

Creazione dell'interfaccia del documento Giornale dei Lavori

Release:	Story ID: 2	Task ID: 2.3
Task Tag: Creazione interfaccia del Registro di Contabilità		Release:
Software Engineer: Michele		

Description:

Creazione dell'interfaccia del documento Registro di Contabilità

Release:	Story ID: 2	Task ID: 2.4
Task Tag: Creazione interfaccia dello Stato di Avanzamento dei Lavori		Release:
Software Engineer: Michele		

Description:

Creazione dell'interfaccia del documento Stato di Avanzamento dei Lavori

Release:	Story ID: 2	Task ID: 2.5
Task Tag: Creazione del Mockup		Release:
Software Engineer: Lorenzo		

Description:

Creazione del mockup che illustra l'interfaccia dei documenti che compongono il contratto d'appalto

Release:	Story ID: 2	Task ID: 2.6
Task Tag: Test di visualizzazione		Release:
Software Engineer: Valerio		

Description:

Test sulla corretta visualizzazione delle voci dei vari documenti

Storia 3

Release:	Story ID: 3	Task ID: 3.1
Task Tag: Interfaccia di inserimento delle nuove misure		Release:
Software Engineer: Jacopo		

Description:

Implementazione dell'interfaccia grafica necessaria al Direttore dei Lavori per l'inserimento delle misure dei nuovi lavori nel Libretto delle Misure .

Release:	Story ID: 3	Task ID: 3.2
Task Tag: Interfaccia di inserimento di nuove voci nel Giornale dei Lavori		Release:
Software Engineer: Jacopo		

Description:

Implementazione dell'interfaccia grafica necessaria al Direttore dei Lavori per l'inserimento delle nuove voci nel Giornale dei Lavori .

Release:	Story ID: 3	Task ID: 3.3
Task Tag: Creazione del Mockup		Release:
Software Engineer: Lorenzo		

Description:

Creazione del mockup che illustra l'interfaccia per l'inserimento di nuove voci nel Libretto delle Misure e nel Giornale dei Lavori

Release:	Story ID: 3	Task ID: 3.4
Task Tag: Test di inserimento		Release:
Software Engineer: Valerio		

Description:

Test sulla corretto inserimento delle voci nei vari documenti

Storia 4

Release:	Story ID: 4	Task ID: 4.1
Task Tag: Interfaccia per la creazione di un nuovo SC		Release:
Software Engineer: Jacopo		

Description:

Implementazione dell'interfaccia grafica necessaria alla Stazione Appaltante per l'inserimento di tutte le informazioni necessarie alla creazione di un nuovo SC (scelta del Direttore dei Lavori e inserimento della Ditta Appaltatrice, inserimento di tutti i lavori previsti dal contratto d'appalto con i relativi costi, scelta della soglia per l'emissione della notifica di pagamento, etc.)

Release:	Story ID: 4	Task ID: 4.2
Task Tag: Realizzazione del codice per creare nuovi SC		Release:
Software Engineer: Jacopo		

Description:

Realizzazione del codice del programma necessario per la creazione di un nuovo SC creato a partira dai dati inseriti dalla Stazione Appaltante

Release:	Story ID: 4	Task ID: 4.3
Task Tag: Creazione del Mockup		Release:
Software Engineer: Lorenzo		

Description:

Creazione del mockup che illustra l'interfaccia per l'inserimento dei dati necessari per la creazione di un nuovo SC da parte della Stazione Appaltante

Release:	Story ID: 4	Task ID: 4.4
Task Tag: Test di inserimento		Release:
Software Engineer: Valerio		

Description:

Test sul corretto inserimento dei dati durante la creazione di un nuovo SC e verifica sulla corretta creazione dello SC

Storia 5

Release:	Story ID: 5	Task ID: 5.1
Task Tag: Realizzazione del codice dello SC		Release:
Software Engineer: Jacopo		

Description:

Realizzazione del codice del programma necessario per l'implementazione degli SC

Release:	Story ID: 5	Task ID: 5.2
Task Tag: Realizzazione del codice per realizzare gli automatismi		Release:
Software Engineer: Jacopo		

Description:

Realizzazione del codice del programma necessario per la realizzazione degli automatismi presenti nello SC per quanto riguarda i documenti del contratto d'appalto, cioè la compilazione automatica delle voci del Registro di Contabilità e dello Stato di Avanzamento dei Lavori quando si riceve la conferma da parte della Stazione Appaltante

Release:	Story ID: 5	Task ID: 5.3

Task Tag: Realizzazione del meccanismo di invio notifiche	Release:

Software Engineer: Jacopo

Description:

Realizzazione del codice del programma necessario per la realizzazione del meccanismo che invia la notifica di pagamento alla Stazione Appaltante ogni qualvolta la soglia dei lavori svolti viene superata

Release:	Story ID: 5	Task ID: 5.4

Task Tag: Realizzazione del codice per le riserve sulle misure	Release:

Software Engineer: Jacopo

Description:

Realizzazione del codice del programma necessario per la realizzazione del meccanismo che permette l'inserimento delle riserve sulle misure registrate, in modo che le misure con riserva non facciano avanzare lo stato dei lavori, ma siano comunque registrate nel Libretto delle Misure.

Release:	Story ID: 5	Task ID: 5.5
Task Tag: Test di correttezza del codice		Release:
Software Engineer: Valerio		

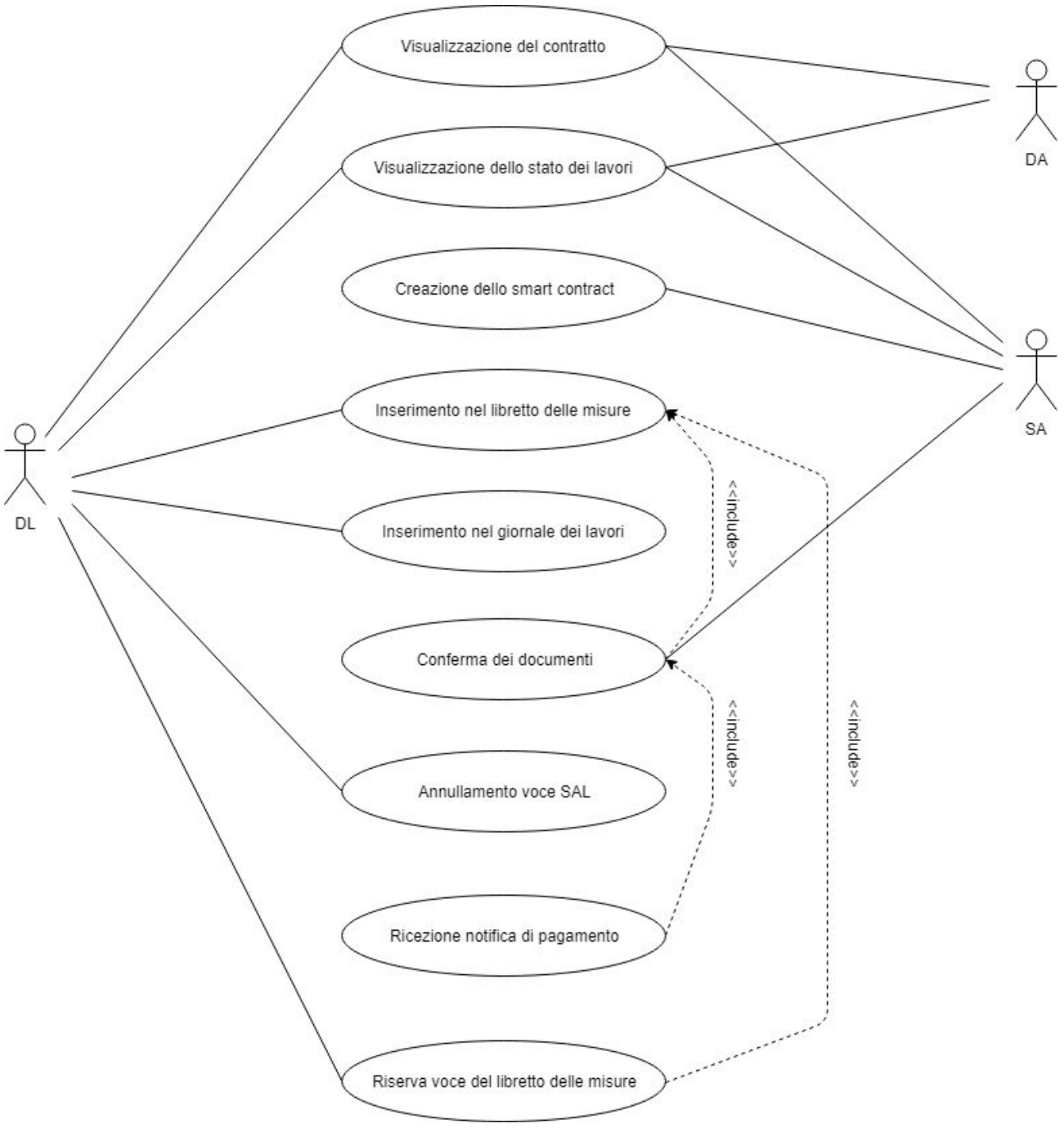
Description:

Test sulla corretta esecuzione del codice per verificare che il codice si comporti nella maniera prestabilita

3.2.6 Diagramma dei Casi d'Uso

Un caso d'uso serve per modellare i requisiti del sistema dal punto di vista dell'utente, per cui lo scopo della realizzazione del modello dei casi d'uso del sistema è quello di aiutarci nella definizione di cosa esiste al di fuori del sistema (attori, tra cui gli utenti del sistema) e che cosa dovrebbe essere fatto dal sistema (casi d'uso). La creazione di questo diagramma è resa possibile grazie ai vari incontri tenutesi con gli stakeholders ed alla documentazione ricevuta cioè grazie alla raccolta dei requisiti del sistema.

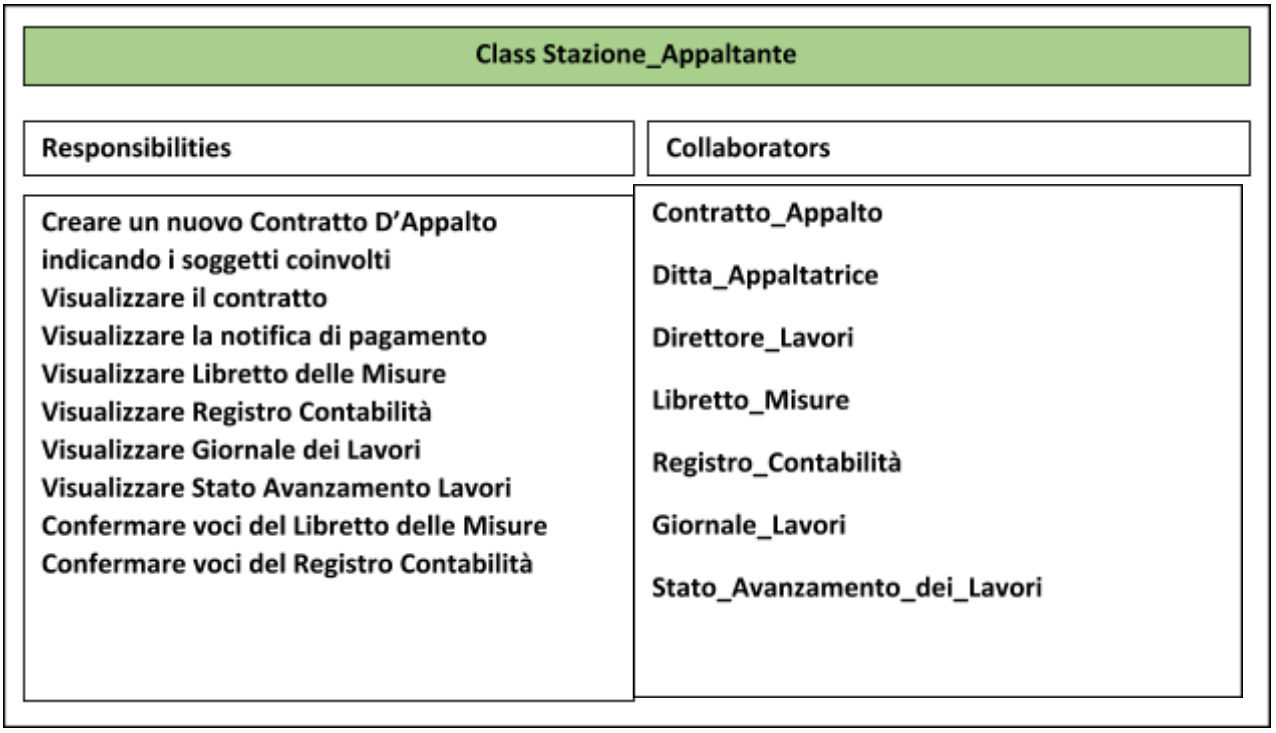
Quello riportato qui sotto è il digramma dei Casi d'Uso del nostro sistema:


Visualizzazione del contratto
Visualizzazione dello stato dei lavori
Creazione dello smart contract
Inserimento nel libretto delle misure
Inserimento nel giornale dei lavori
Conferma dei documenti
Annullamento voce SAL
Ricezione notifica di pagamento
Riserva voce del libretto delle misure
DL
DA
SA
<<include>>
<<include>>
<<include>>

4 Analisi dei requisiti

4.1 Analisi CRC

Per ogni storia utente vengono definite le classi, le responsabilità e i collaboratori. Per evidenziare le classi che compaiono all'interno delle storie utente sono stati evidenziati i nomi presenti in ogni storia e per meglio individuare le responsabilità sono stati evidenziati i verbi. Dopo aver costruito tutte le schede CRC si valuta l'eliminazione di una scheda se questa non contiene né responsabilità né collaboratori, diventando così un attributo. Di seguito sono riportate tutte le schede CRC (dove il colore verde indica una classe che serve per modellare l'interfaccia, mentre il colore azzurro indica che la classe non serve per modellare l'interfaccia):



Class Direttore_Lavori	
Responsibilities	**Collaborators**
Inserire valori delle misure effettuate Visualizzare il contratto Visualizzare la notifica di pagamento Visualizzare Libretto delle Misure Visualizzare Registro Contabilità Visualizzare Giornale dei Lavori Visualizzare Stato Avanzamento Lavori Annullare voce dello Stato Avanzamento Lavori Inserire riserva voci Libretto delle Misure	Contratto_Appalto Ditta_Appaltatrice Stazione_Appaltante Libretto_Misure Registro_Contabilità Giornale_Lavori Stato_Avanzamento_dei_Lavori

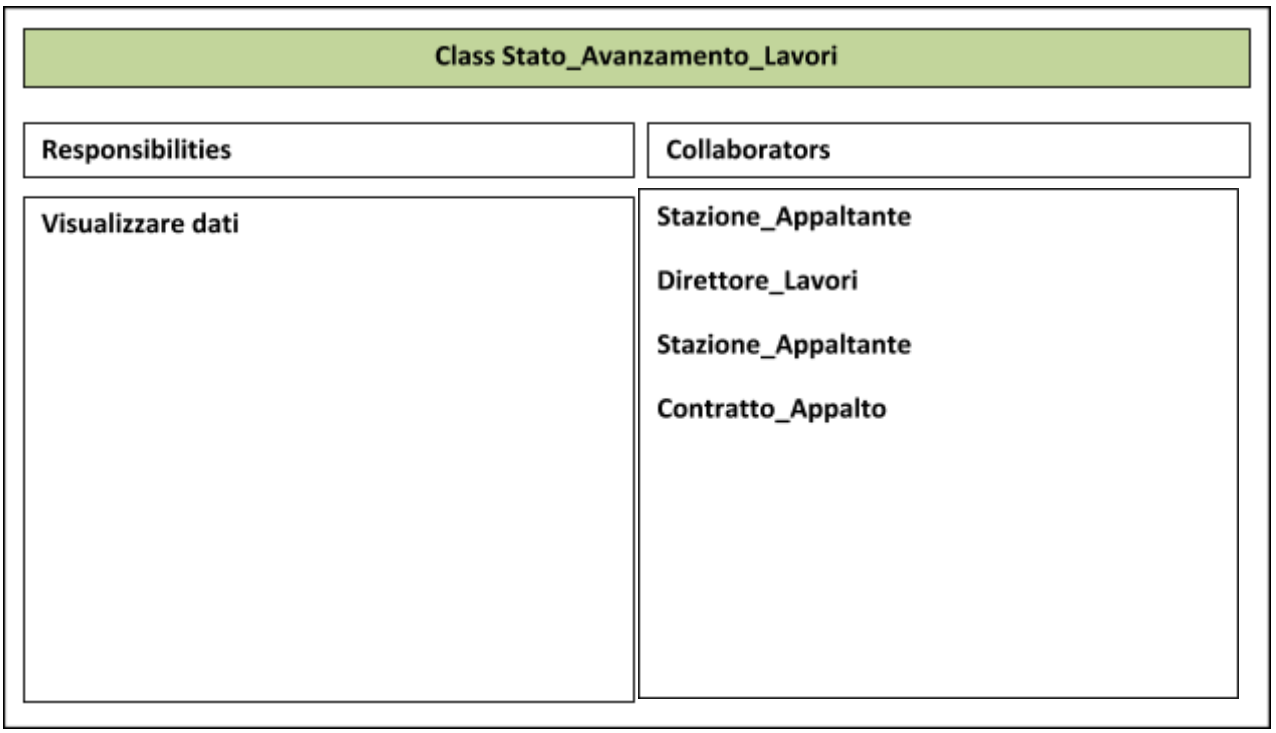
Class Ditta_Appaltatrice	
Responsibilities	**Collaborators**
Visualizzare il contratto Visualizzare la notifica di pagamento Visualizzare Libretto delle Misure Visualizzare Registro Contabilità Visualizzare Giornale dei Lavori Visualizzare Stato Avanzamento Lavori	Contratto_Appalto Stazione_Appaltante Direttore_Lavori Libretto_Misure Registro_Contabilità Giornale_Lavori Stato_Avanzamento_dei_Lavori

Class Contratto_Appalto	
Responsibilities	**Collaborators**
Permettere la creazione di un nuovo SC sulla blockchain Inviare la notifica di pagamento superata la soglia prestabilita Permettere inserimento misure Visualizzare lo stato dei lavori Visualizzare il contratto Annullare voce documento Confermare voci documento Inserire riserva voci Libretto delle Misure	Stazione_Appaltante Direttore_Lavori Stazione_Appaltante Libretto_Misure Reistro_Contabilita Giornale_Lavori Stato_Avanamento_Lavori

Class Libretto_Misure	
Responsibilities	**Collaborators**
Visualizzare dati Memorizzare dati	Stazione_Appaltante Direttore_Lavori Stazione_Appaltante Contratto_Appalto

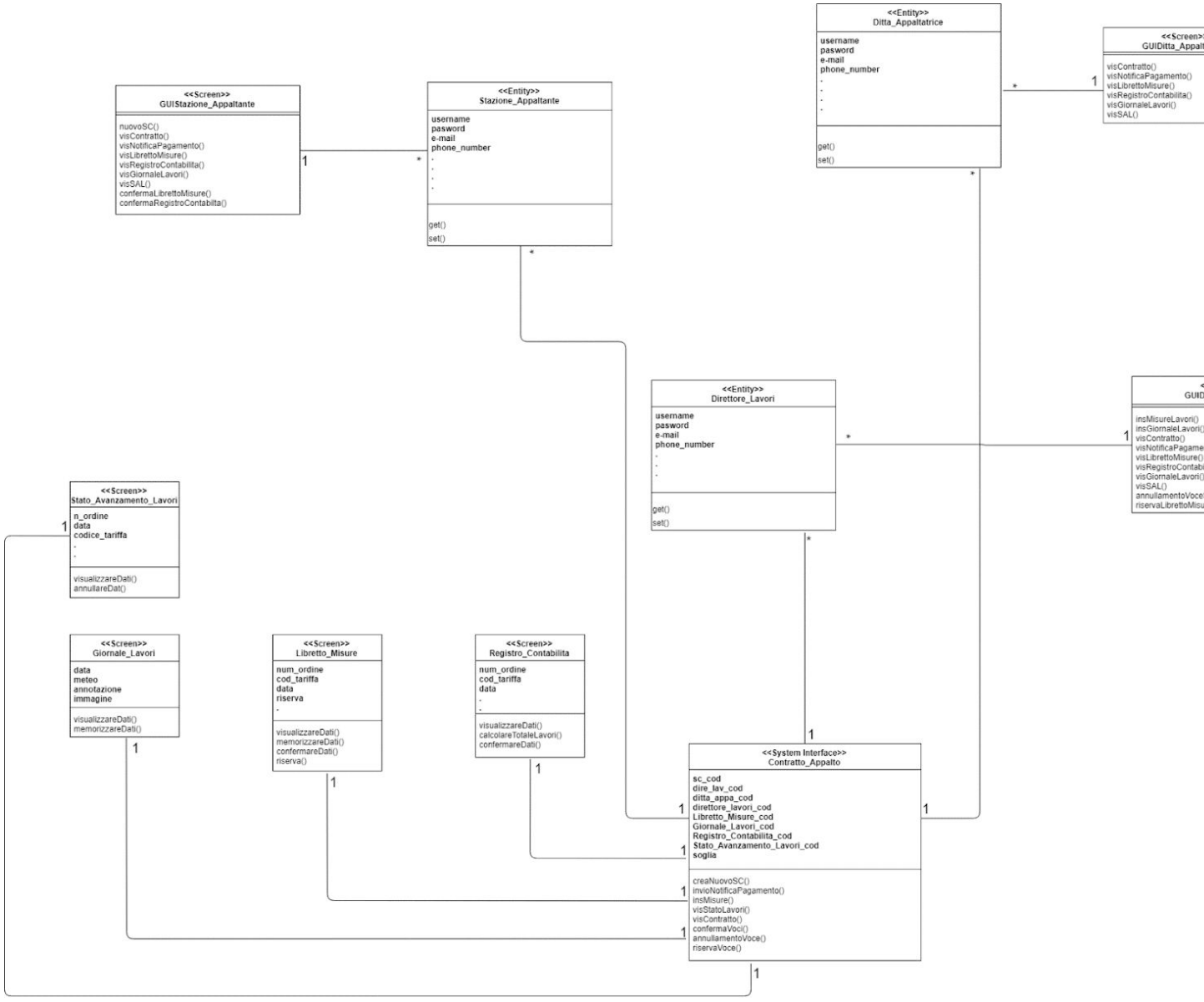
Class Registro_Contabilita	
Responsibilities	**Collaborators**
Visualizzare dati Calcolare totale lavori	Stazione_Appaltante Direttore_Lavori Stazione_Appaltante Contratto_Appalto

Class Giornale_Lavori	
Responsibilities	**Collaborators**
Visualizzare dati Memorizzare dati	Stazione_Appaltante Direttore_Lavori Stazione_Appaltante Contratto_Appalto



In ogni scheda sono riportati i nomi delle classi, le responsabilità di quella classe (i metodi che la classe dovrà implementare) ed i collaboratori. I collaboratori rappresentano le classi o attributi con cui sono collegate e in assenza di collaboratori e responsabilità possono essere ridotti ad attributi.

4.2 Analisi orientata ai dati



Per ogni CRC, si definisce una classe se presenta responsabilità e collaboratori (i cui metodi sono dati dalle responsabilità), mentre se la scheda presenta pochi o nessun collaboratore e nessuna responsabilità allora si definisce un attributo. Successivamente ogni classe così ottenuta viene decorata con uno stereotipo indicato con << >> posizionato sopra il nome della classe.
Ad esempio, per le classi appartenenti allo stereotipo UI e System/device interface ogni responsabilità è un task, lo stereotipo UI viene usato per le classi che rappresentano un mockup legato ad un utente, mentre lo stereotipo System/device interface viene usato quando il nome della classe è un attore. Ancora, lo stereotipo Entity viene associato a classi che modellano informazioni persistenti le cui responsabilità descrivono operazioni di accesso/modifica delle informazioni. Mentre gli stereotipi Actor e Control rispettivamente quando la classe è relativa ad un utente nel primo caso, nel secondo quando la classe non appartiene a nessun'altro caso precedentemente descritto. Infine, per ogni CRC per cui è stata definita una classe, per ogni classe che compare nelle collaborazioni, si inserisce una associazione tra le due classi, si stabilisce la cardinalità e vengono

stabilite delle eventuali relazioni di generalizzazione. Nel caso dei collaboratori di una classe comparisse un attributo esso diventerebbe un attributo della classe di cui è il collaboratore. Viene così costruito il modello delle classi d'analisi seguendo le istruzioni sopra riportate (il modello delle classi d'analisi così ottenuto dalle schede CRC precedentemente create è riportato nell'immagine qui sotto).

4.3 Analisi orientata ai comportamenti

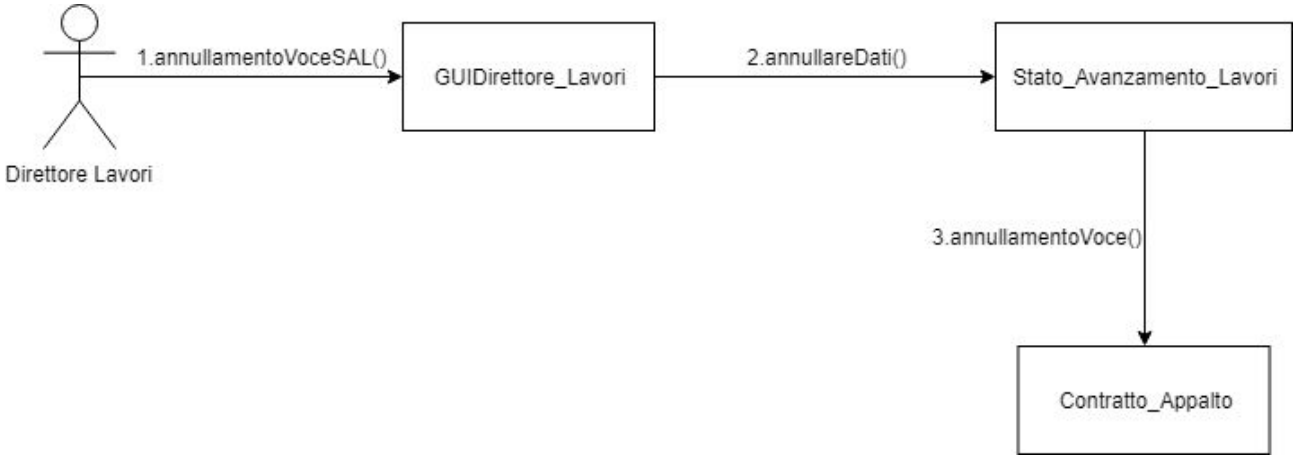
Il modello del comportamento rappresenta come devono essere svolte le operazioni associate ai dati e come i dati interagiscono tra loro. Per questo tipo di analisi possono essere utilizzati i diagrammi di analisi delle collaborazioni, delle sequenze, di stato e delle attività.

4.3.1 Diagramma delle Collaborazioni/Sequenze

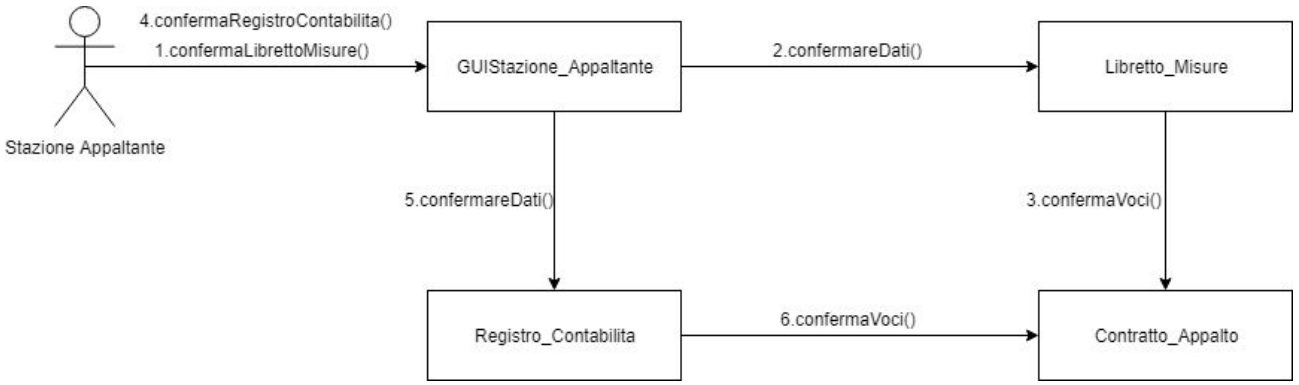
Il diagramma di collaborazione serve per modellare un contesto di interazione tra oggetti del sistema, ovvero un insieme di elementi che sono in relazione tra loro per uno scopo preciso, come l'esecuzione di una operazione. È un grafo dove i nodi rappresentano gli oggetti e gli archi rappresentano il flusso dei messaggi.

Il diagramma delle sequenze serve per modellare esplicitamente la successione temporale degli eventi durante una interazione tra oggetti del sistema. È un diagramma alternativo ai diagrammi delle collaborazioni.

Di seguito sono riportati i diagrammi di analisi delle collaborazioni che sono stati realizzati per il nostro progetto degli SC:



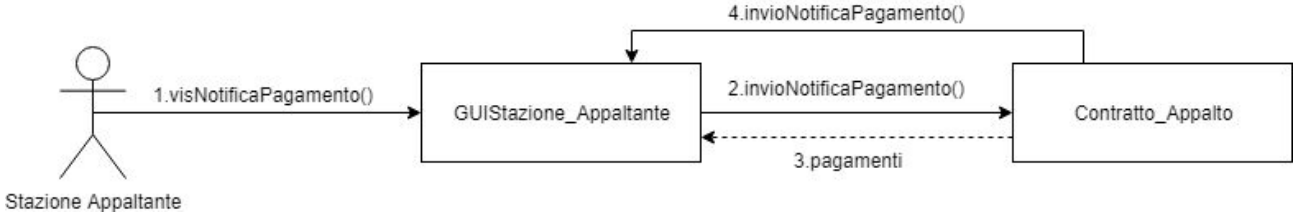
Collaboration Diagram annullamento voce SAL



Collaboration Diagram conferma

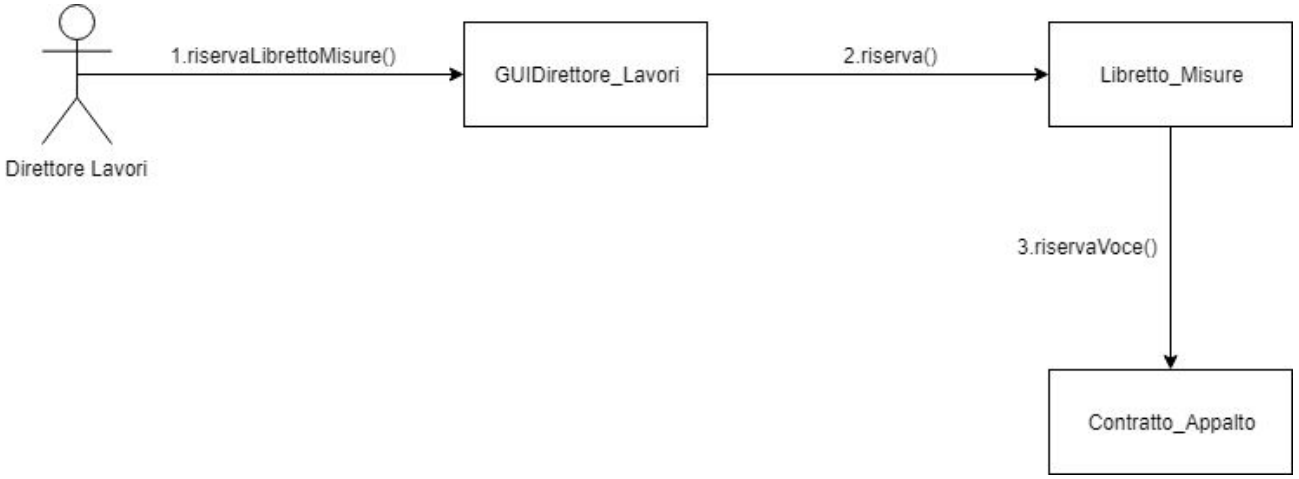
Collaboration Diagram inserimento voce nel Giornale dei Lavori

Collaboration Diagram inserimento voce nel Libretto delle Misure



Collaboration Diagram visualizzazione della notifica di pagamento

Collaboration Diagram creazione di un nuovo SC

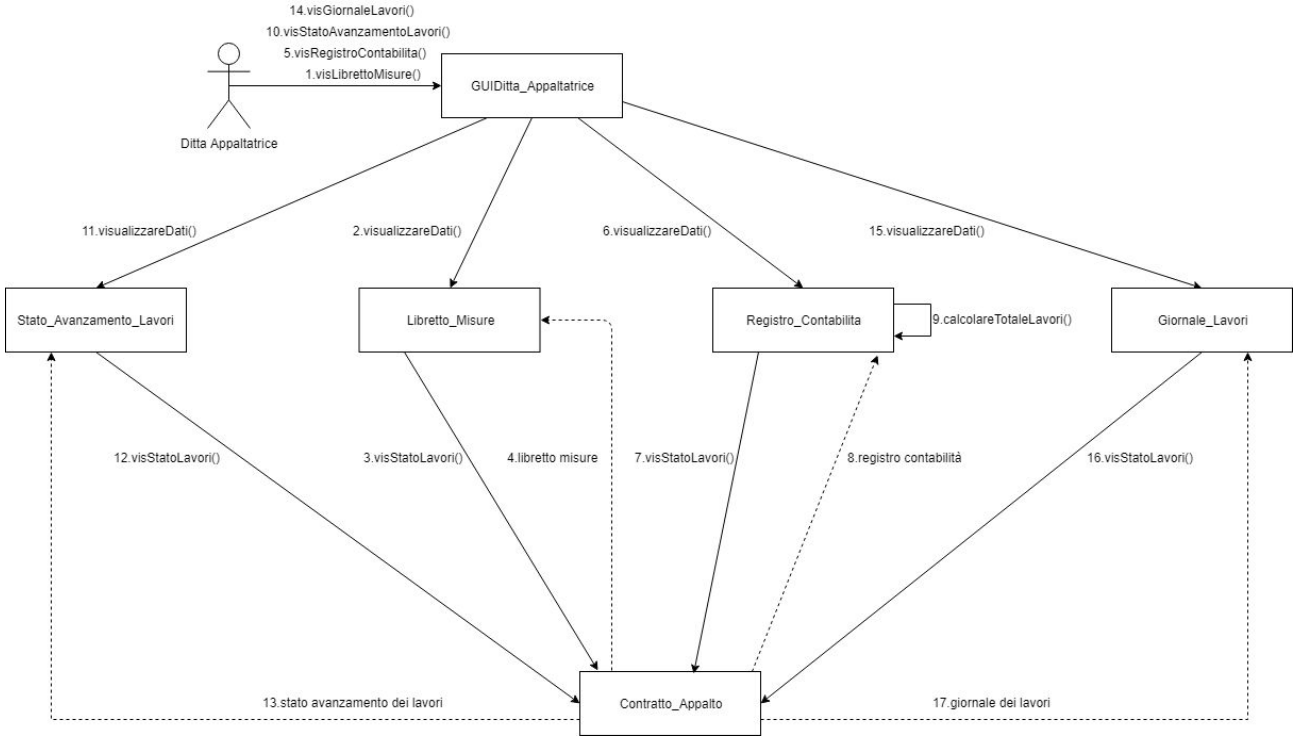


Collaboration Diagram inserimento della riserva su una voce del Libretto delle Misure

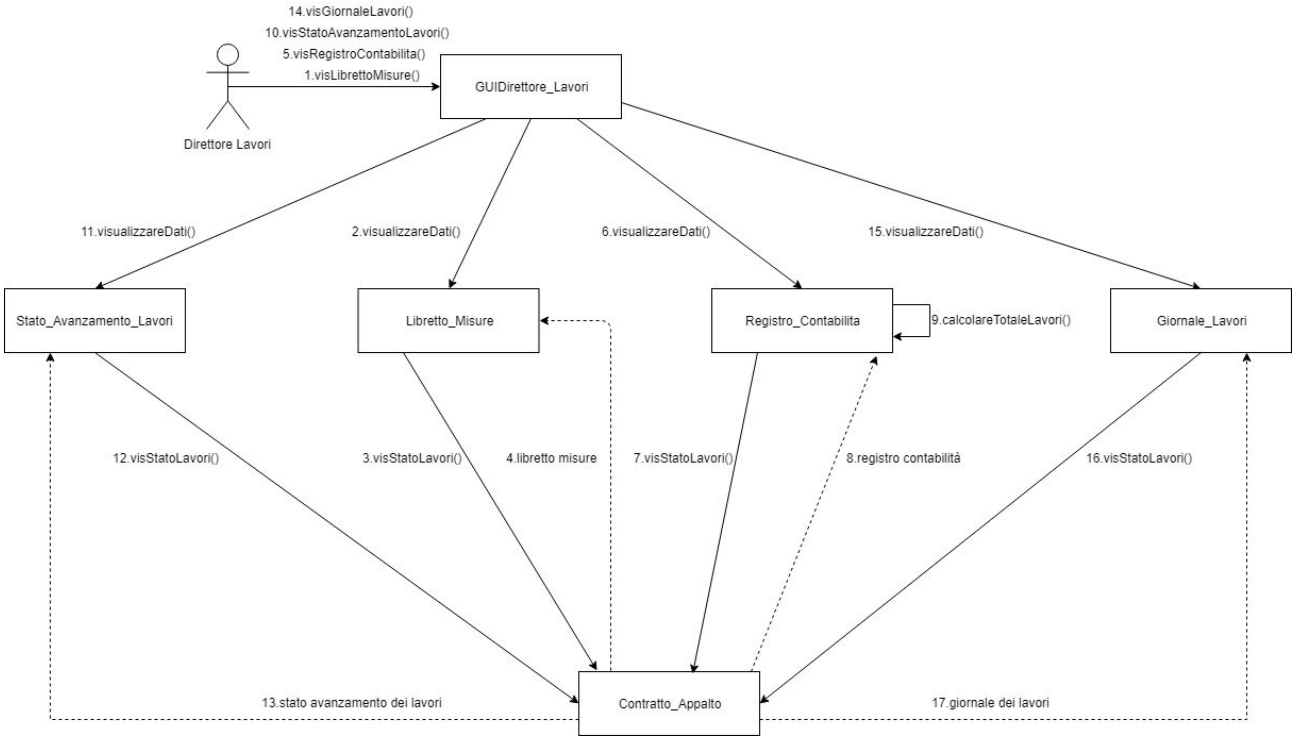
Collaboration Diagram visualizzazione del contratto per la Ditta Appaltatrice

Collaboration Diagram visualizzazione del contratto per il Direttore dei Lavori

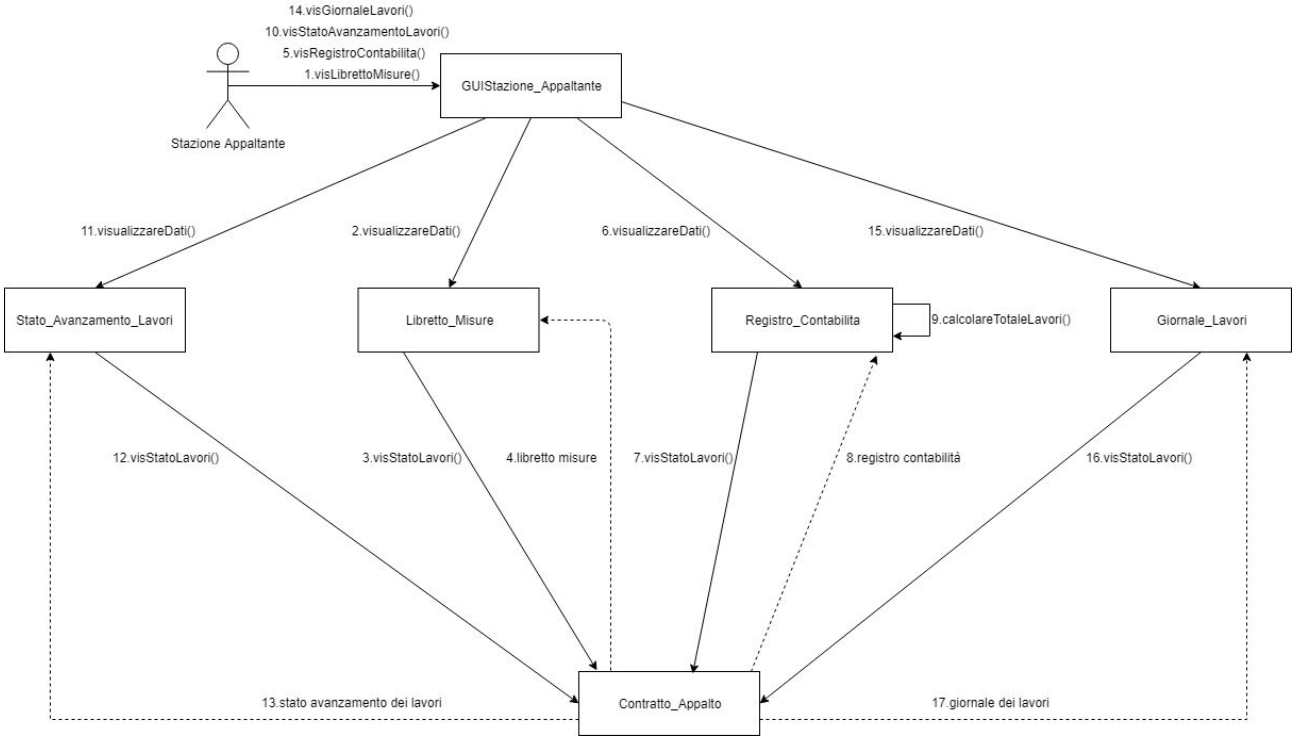
Collaboration Diagram visualizzazione del contratto per la Stazione Appaltante



Collaboration Diagram visualizzazione dello stato dei lavori per la Ditta Appaltatrice

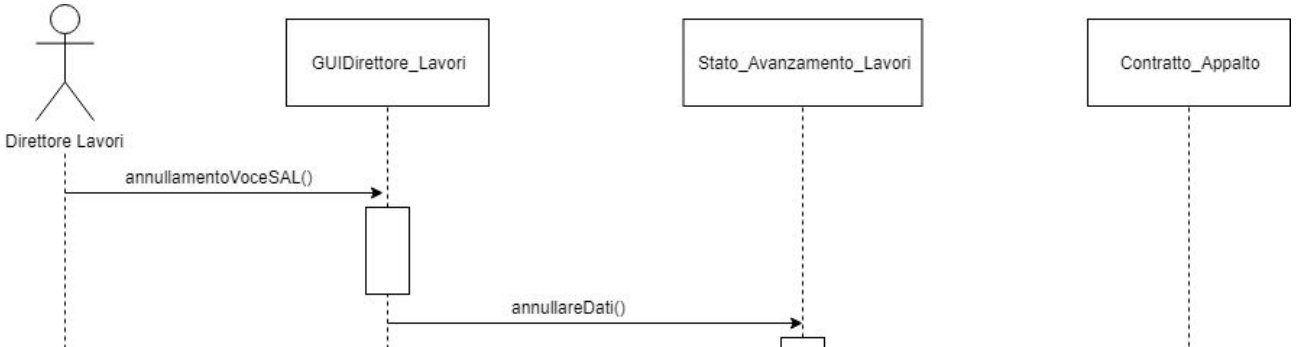


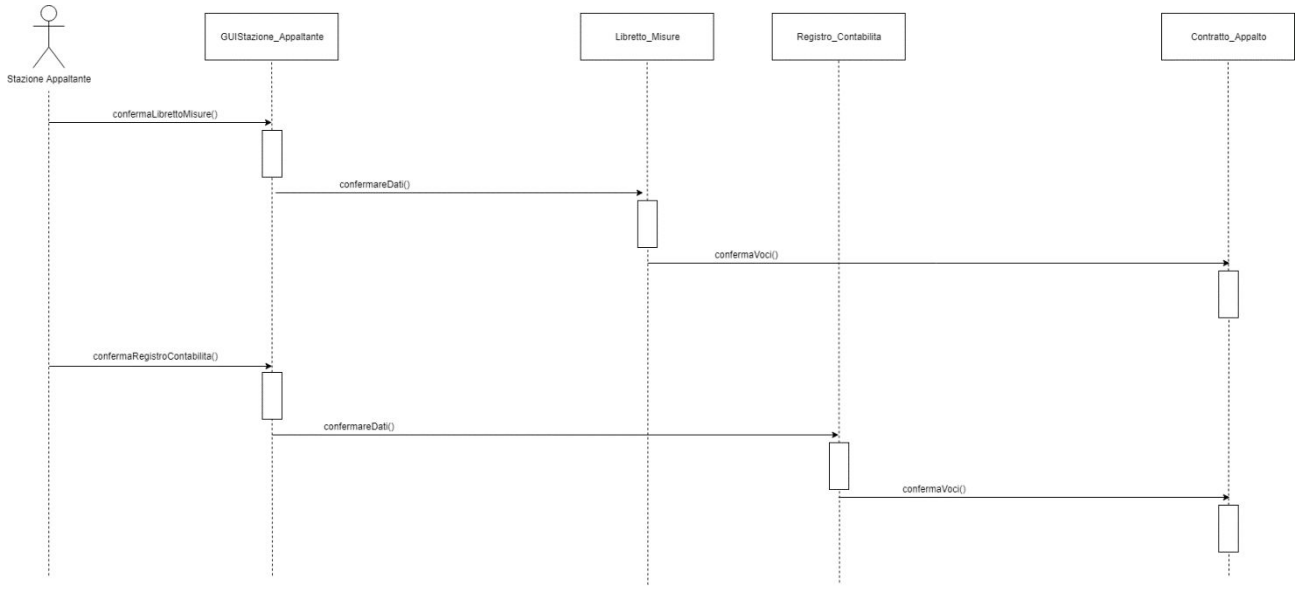
Collaboration Diagram visualizzazione dello stato dei lavori per il Direttore dei Lavori



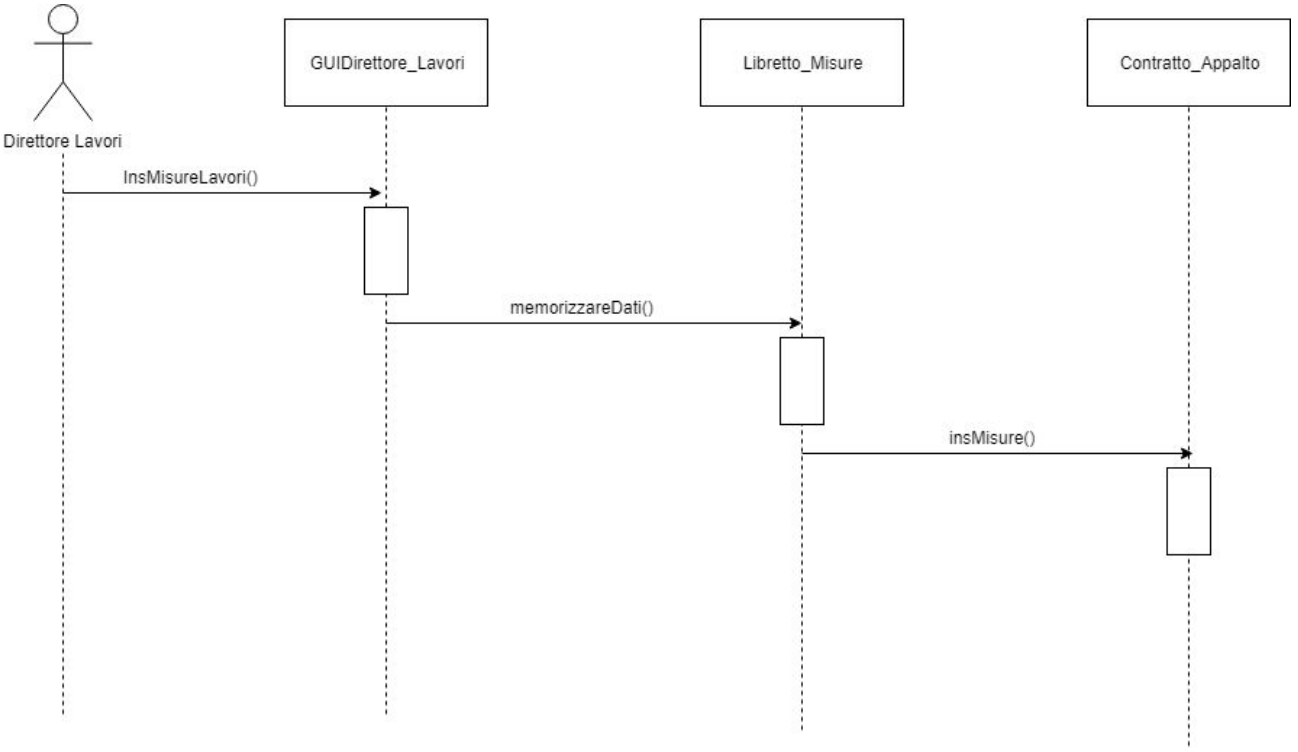
Collaboration Diagram visualizzazione dello stato dei lavori per la Stazione Appaltante

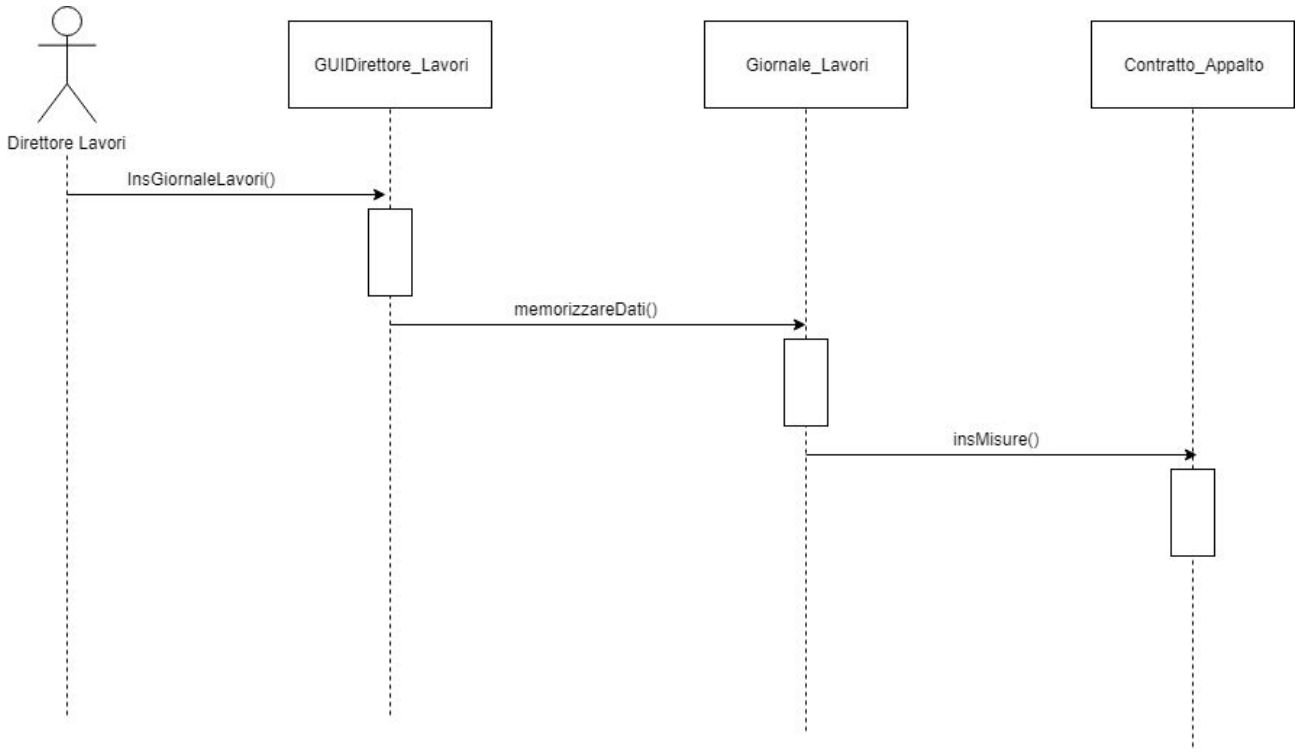
Di seguito, invece, sono riportati i diagrammi di analisi delle sequenze ottenuti:



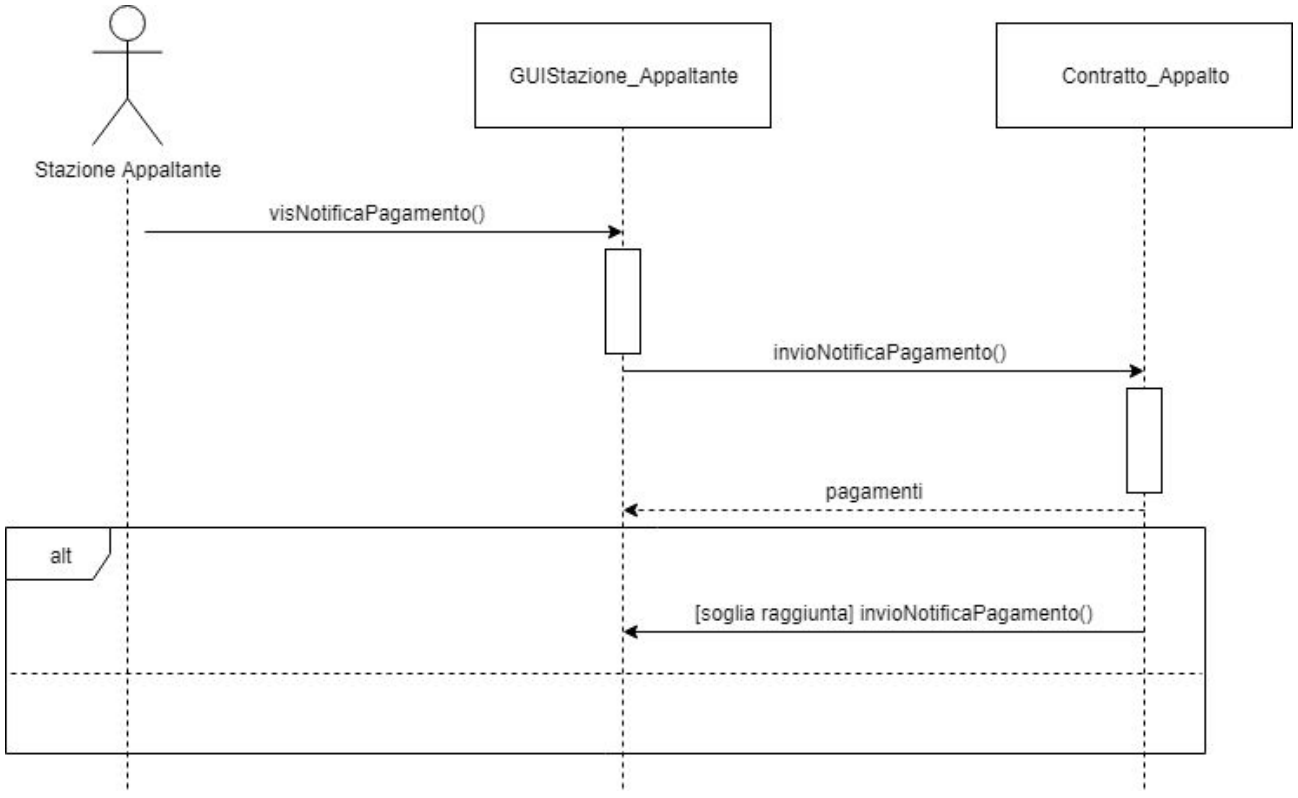


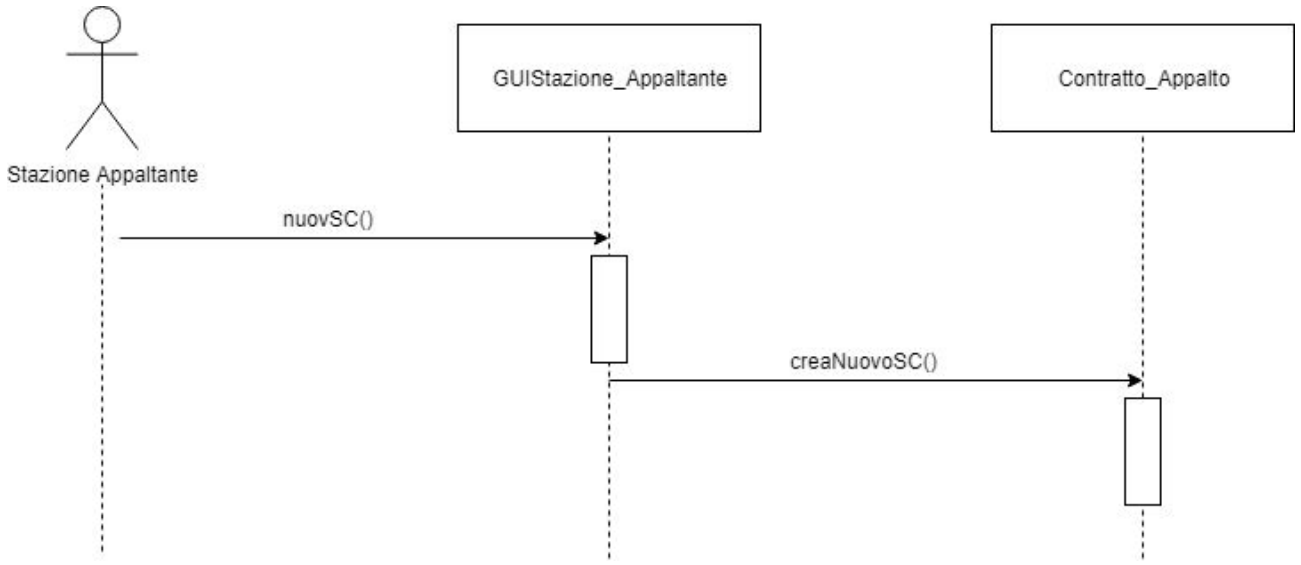
Sequence Diagram conferma



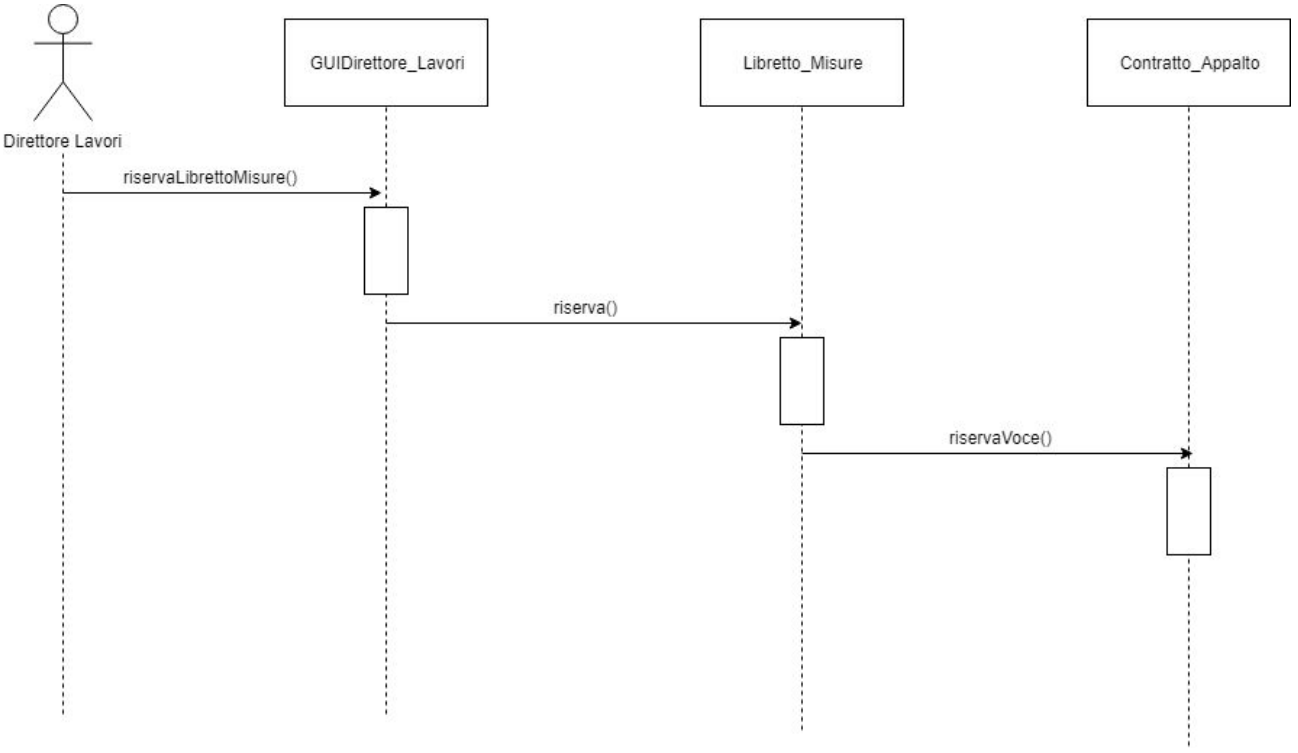


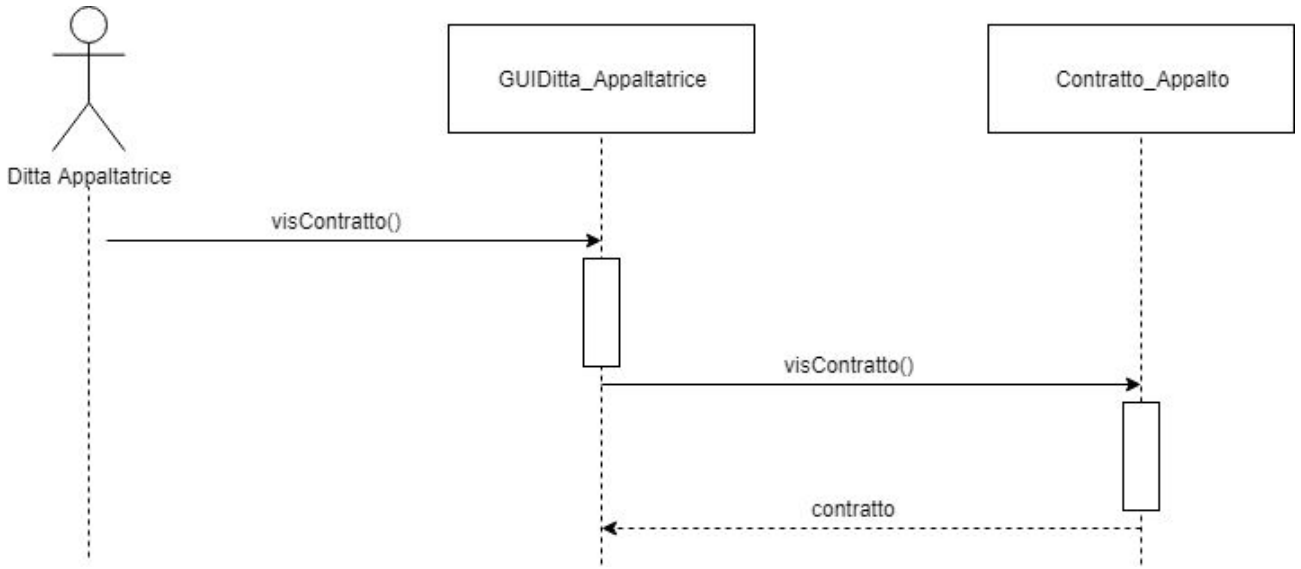
Sequence Diagram inserimento voce nel Giornale dei Lavori



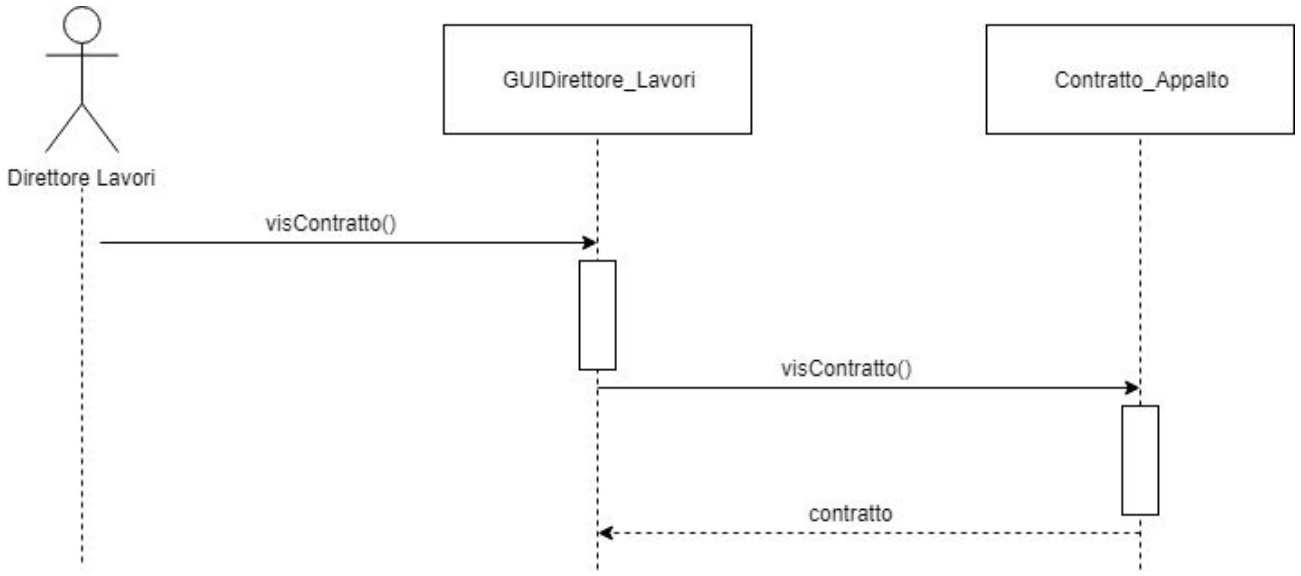


Sequence Diagram creazione di un nuovo SC

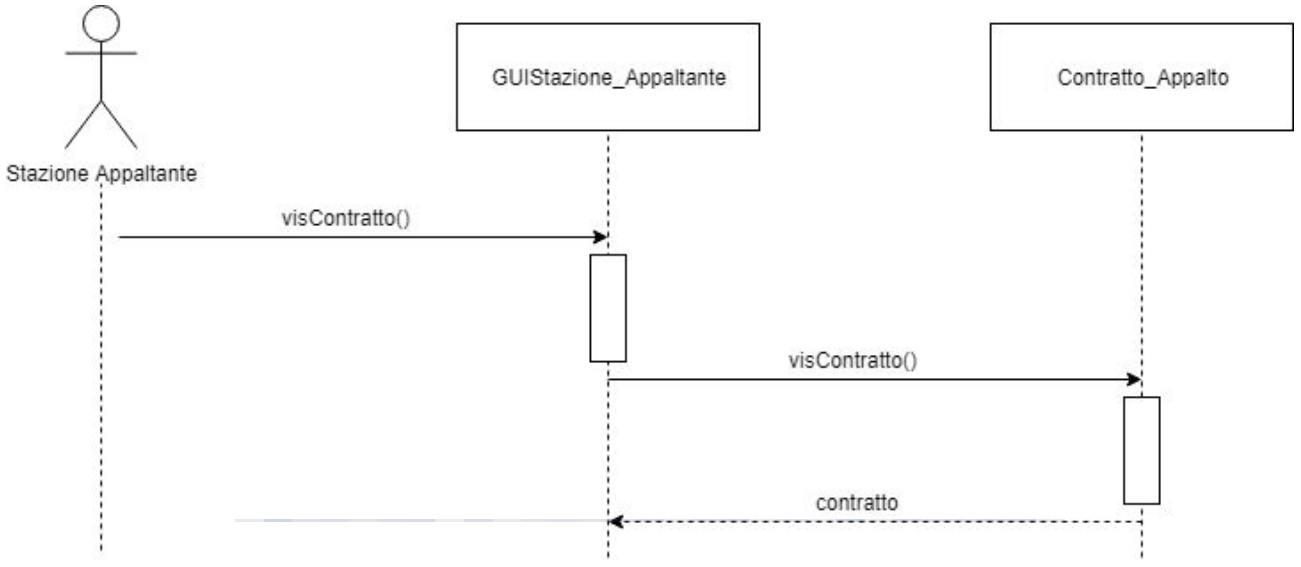


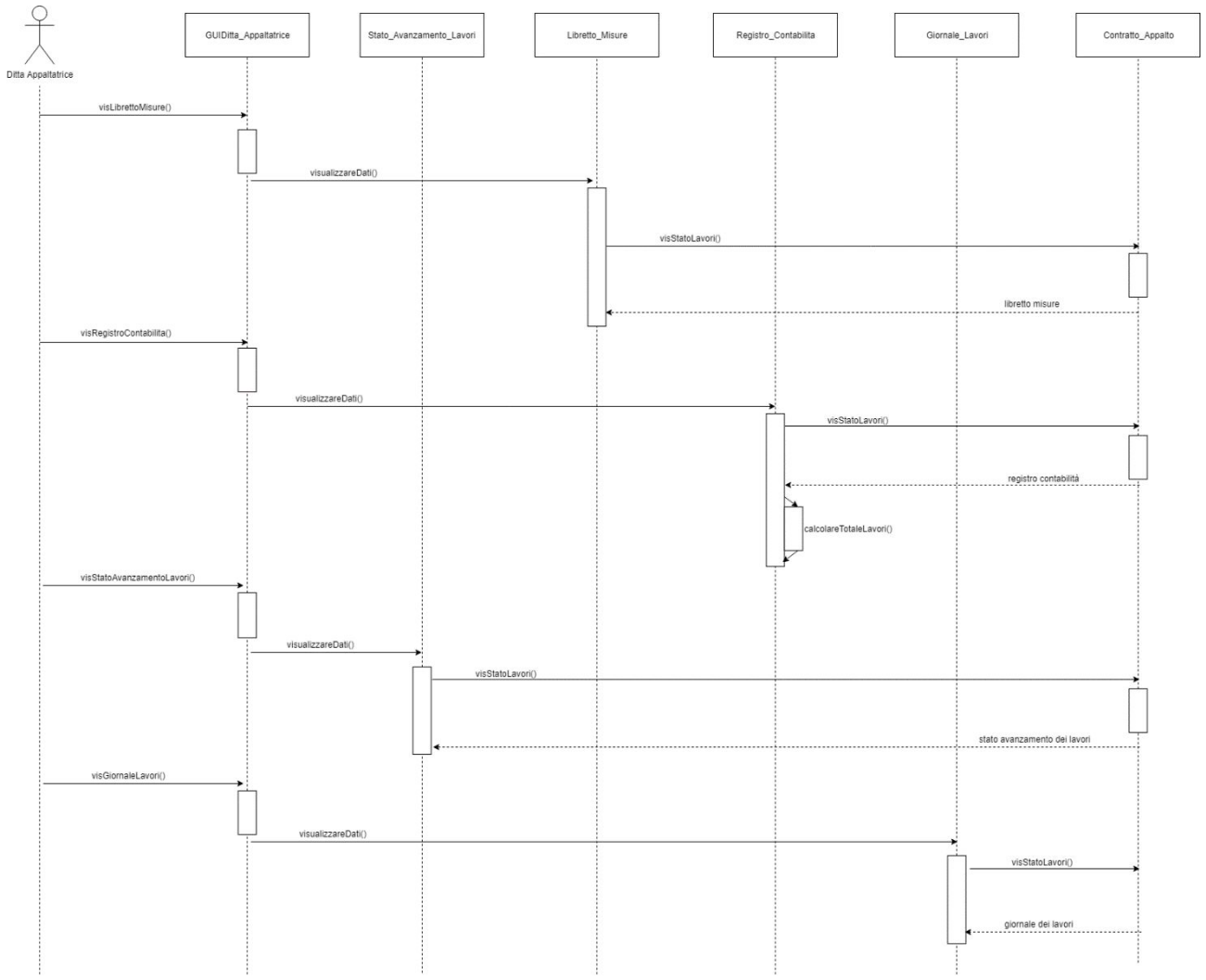


Sequence Diagram visualizzazione del contratto per la Ditta Appaltatrice

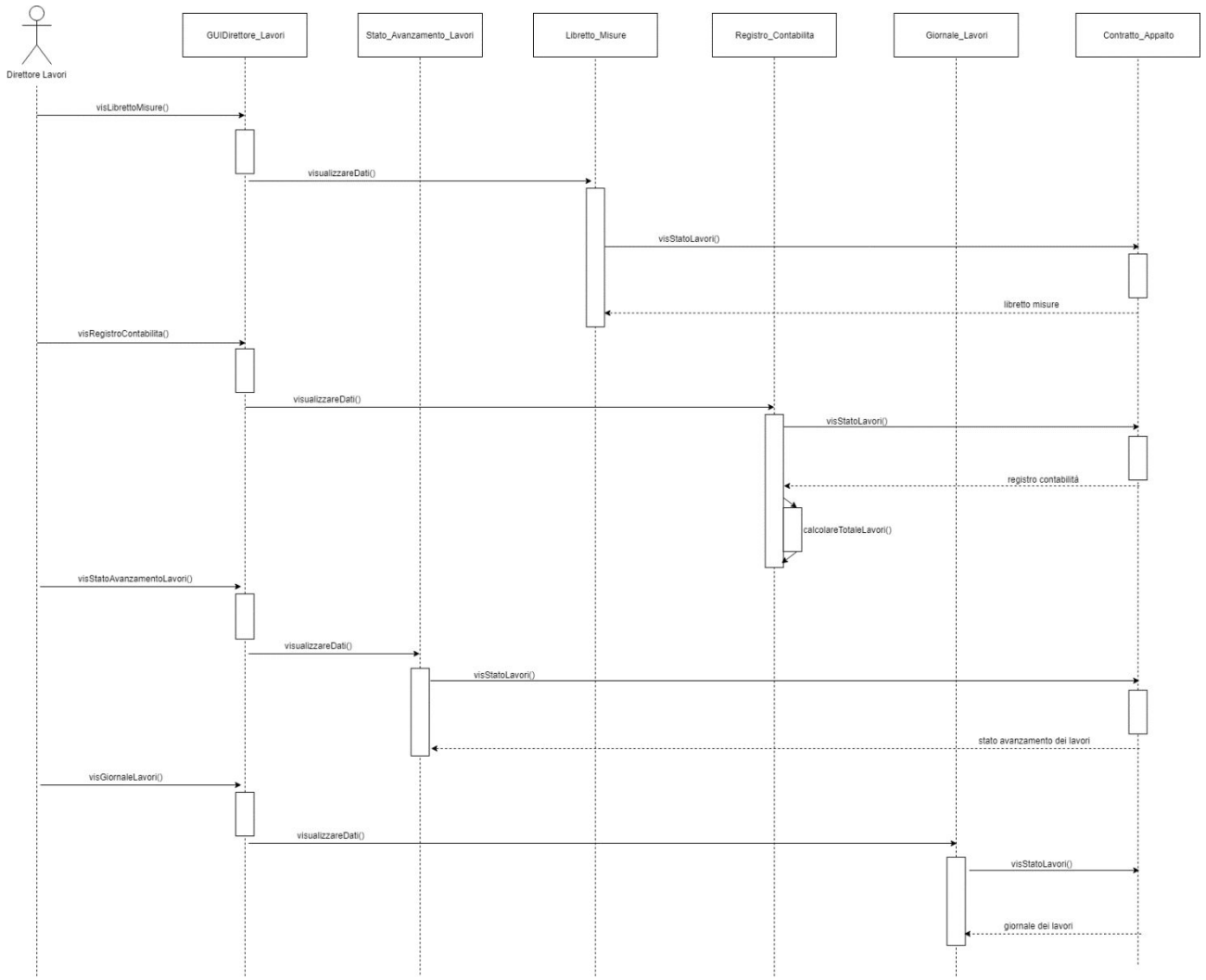


Sequence Diagram visualizzazione del contratto per il Direttore dei Lavori

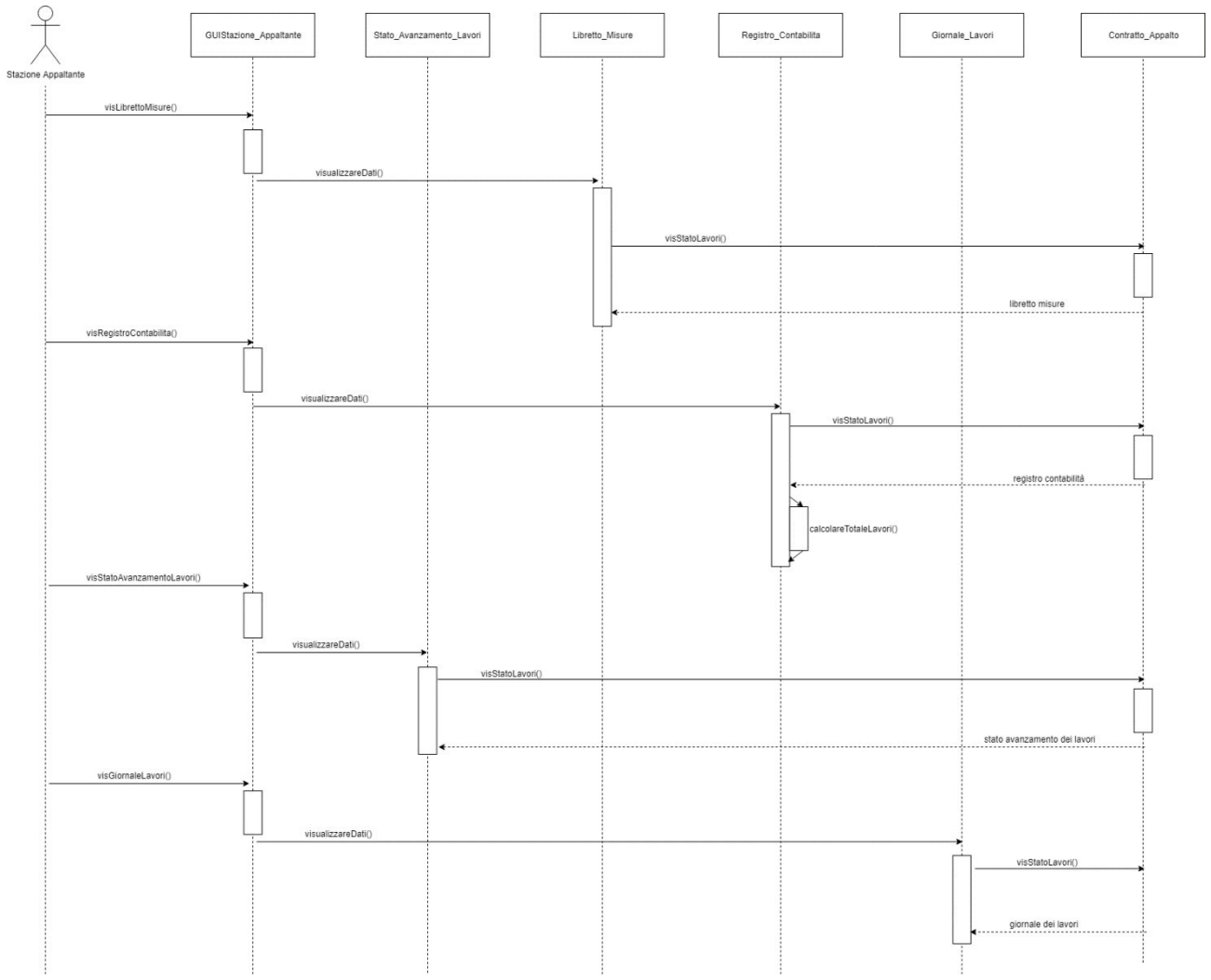




Sequence Diagram visualizzazione dello stato dei lavori per la Ditta Appaltatrice



Sequence Diagram visualizzazione dello stato dei lavori per il Direttore dei Lavori



Sequence Diagram visualizzazione dello stato dei lavori per la Stazione Appaltante

4.3.2 Modello degli Stati

Un diagramma di stato mostra come cambia una entità (attributo, istanza, classe, sottosistema, sistema, caso d'uso) durante il suo ciclo di vita. È un grafo composto da:

- stati (i nodi): rappresenta una possibile "situazione" in cui si può trovare un sistema ovvero i valori che un insieme di variabili (dette variabili di stato) assumono in un dato istante;
- transizioni (gli archi): l'evoluzione da uno stato all'altro del sistema è rappresentata mediante transizioni.

Nel nostro caso il modello degli Stati ottenuto è mostrato qui sotto:

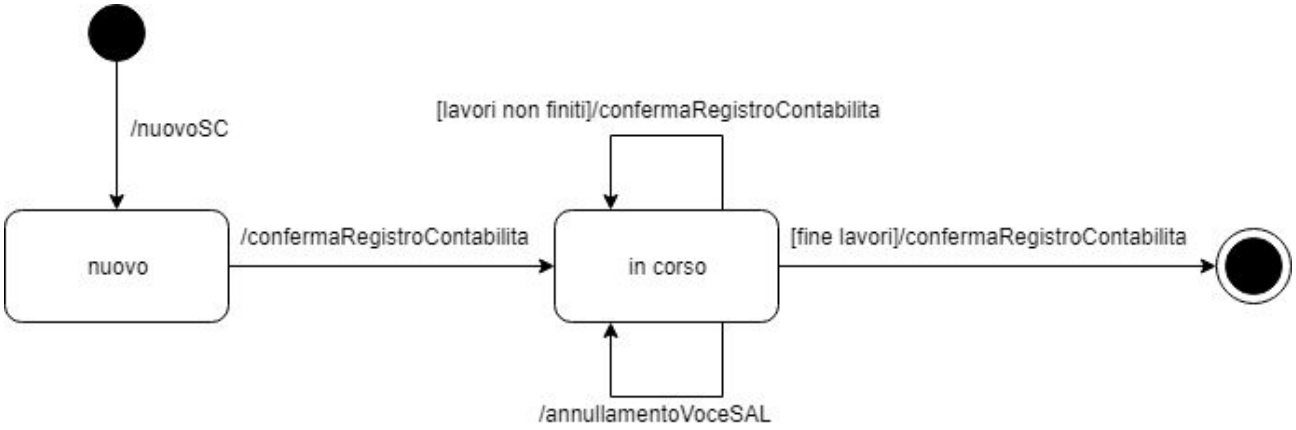
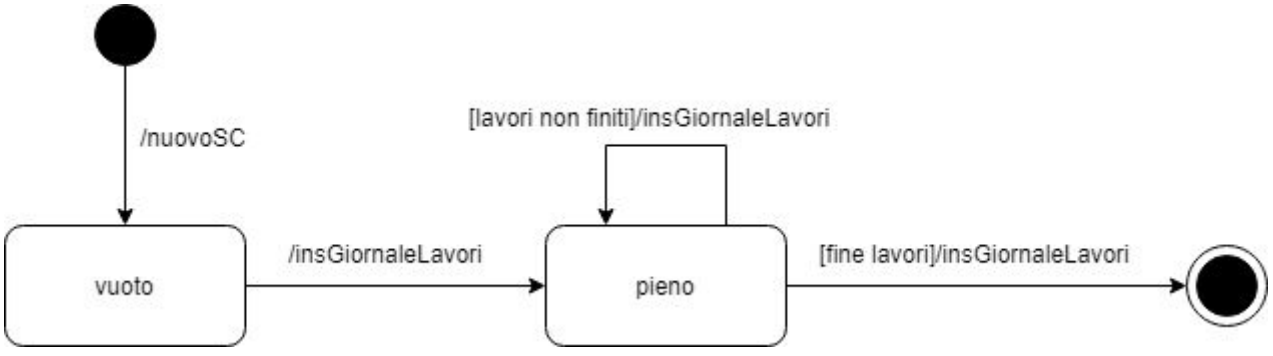
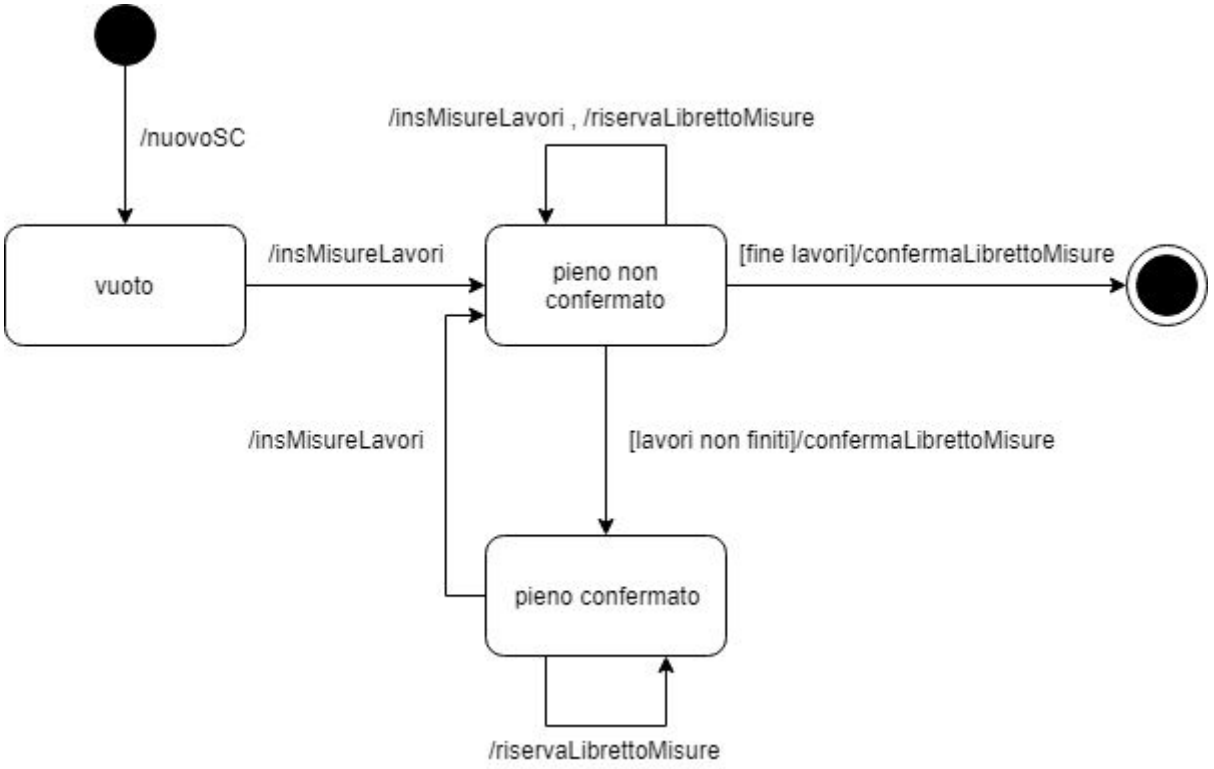
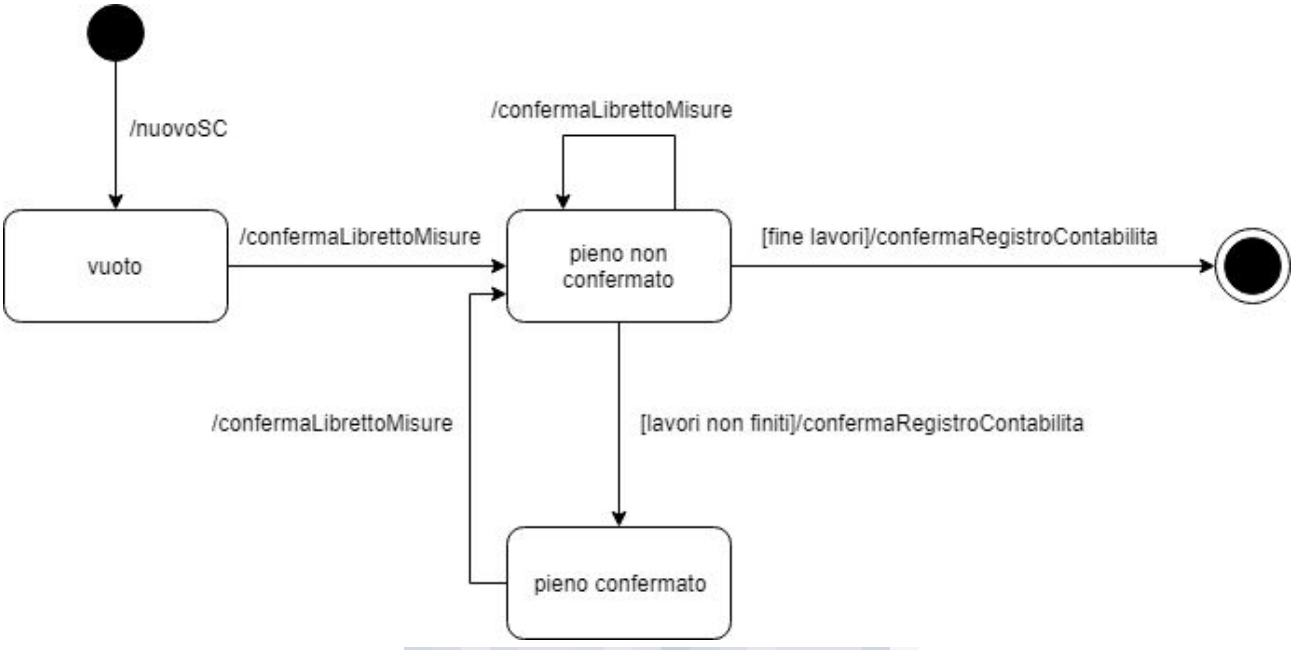
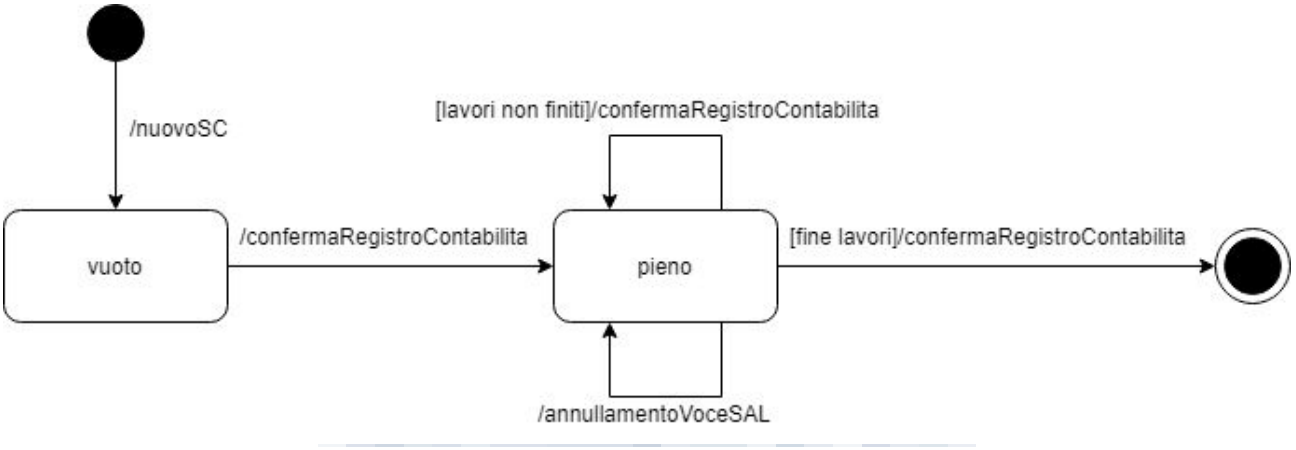


Diagramma di Stato del contratto d'appalto




/nuovoSC
/insMisureLavori , /riservaLibrettoMisure
vuoto
/insMisureLavori
pieno non confermato
[fine lavori]/confermaLibrettoMisure
/insMisureLavori
[lavori non finiti]/confermaLibrettoMisure
pieno confermato
/riservaLibrettoMisure

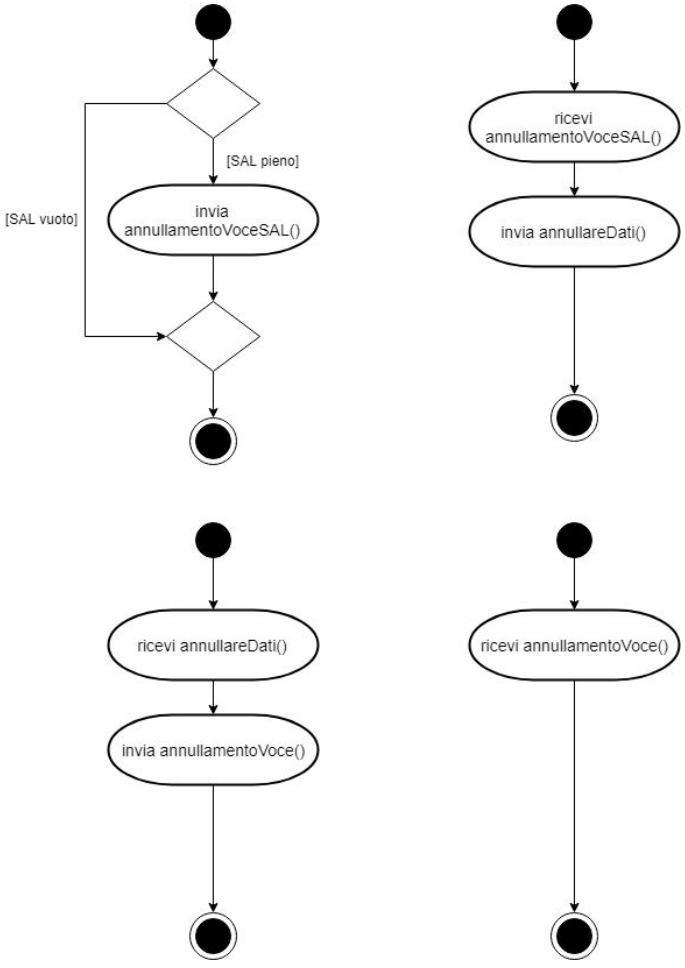

/nuovoSC
/confermaLibrettoMisure
vuoto
/confermaLibrettoMisure
pieno non confermato
[fine lavori]/confermaRegistroContabilita
/confermaLibrettoMisure
[lavori non finiti]/confermaRegistroContabilita
pieno confermato



4.3.3 Modello delle Attività

Un diagramma delle attività (è il diagramma duale del diagramma di stato) descrive il flusso di elaborazione interno di una istanza, una classe, un caso d'uso o di una operazione. È un grafo composto da attività (i nodi) e transizioni (gli archi).

Di seguito son riportati i vari diagrammi delle attività per il nostro progetto:



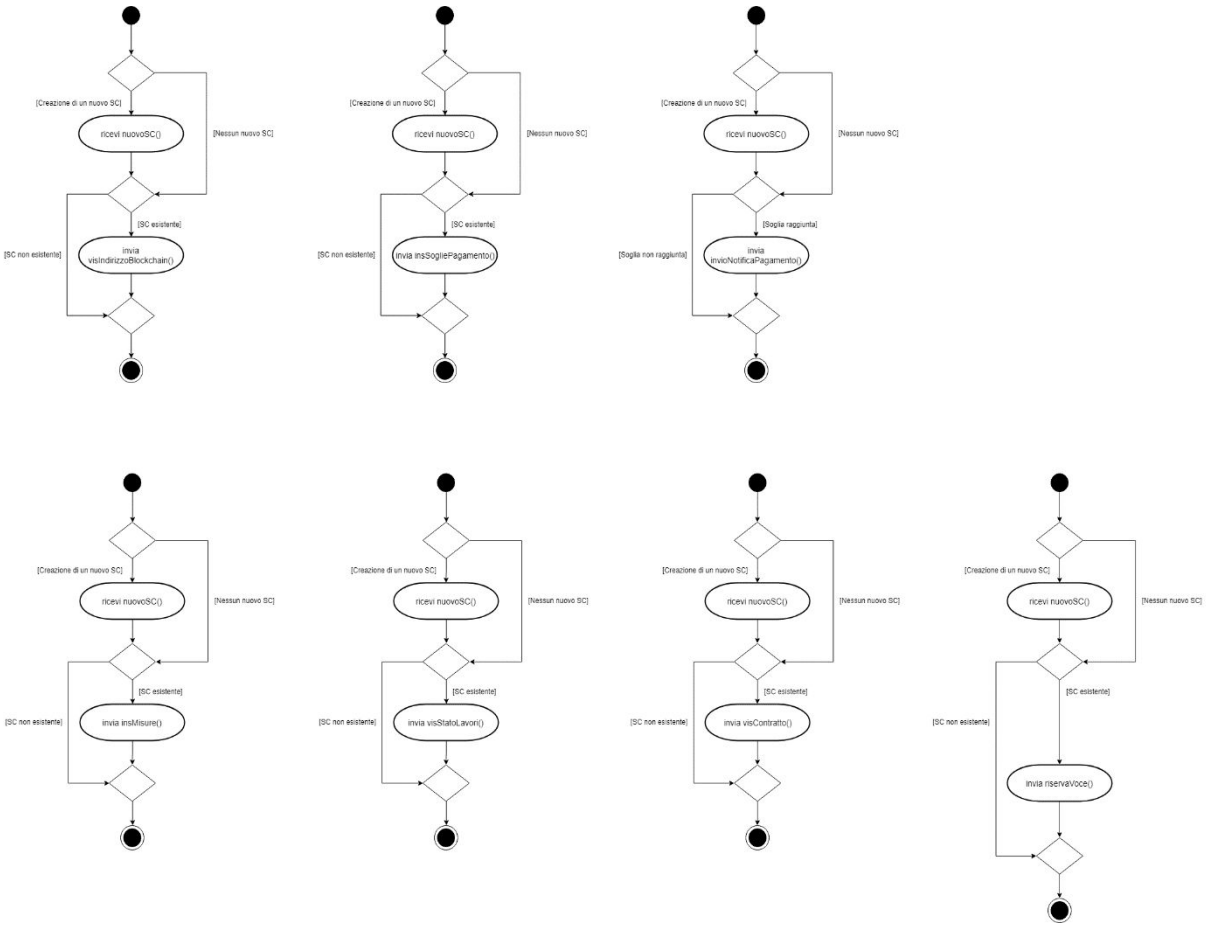


Diagramma delle Attività del contratto d'appalto

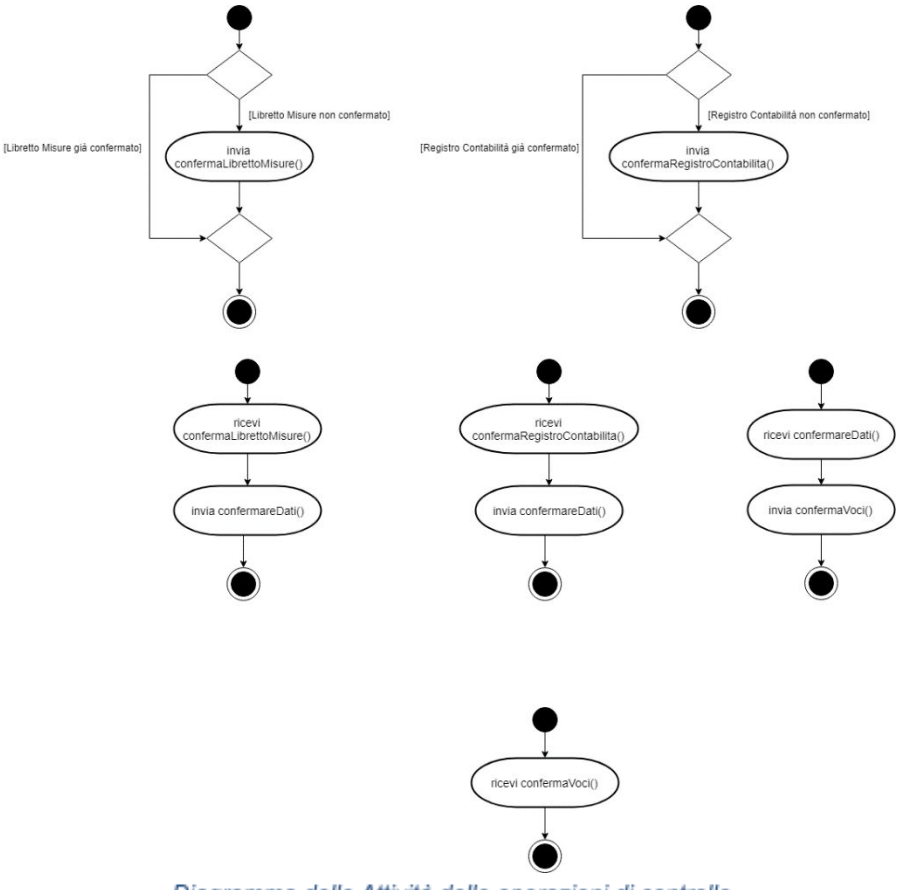


Diagramma delle Attività delle operazioni di controllo

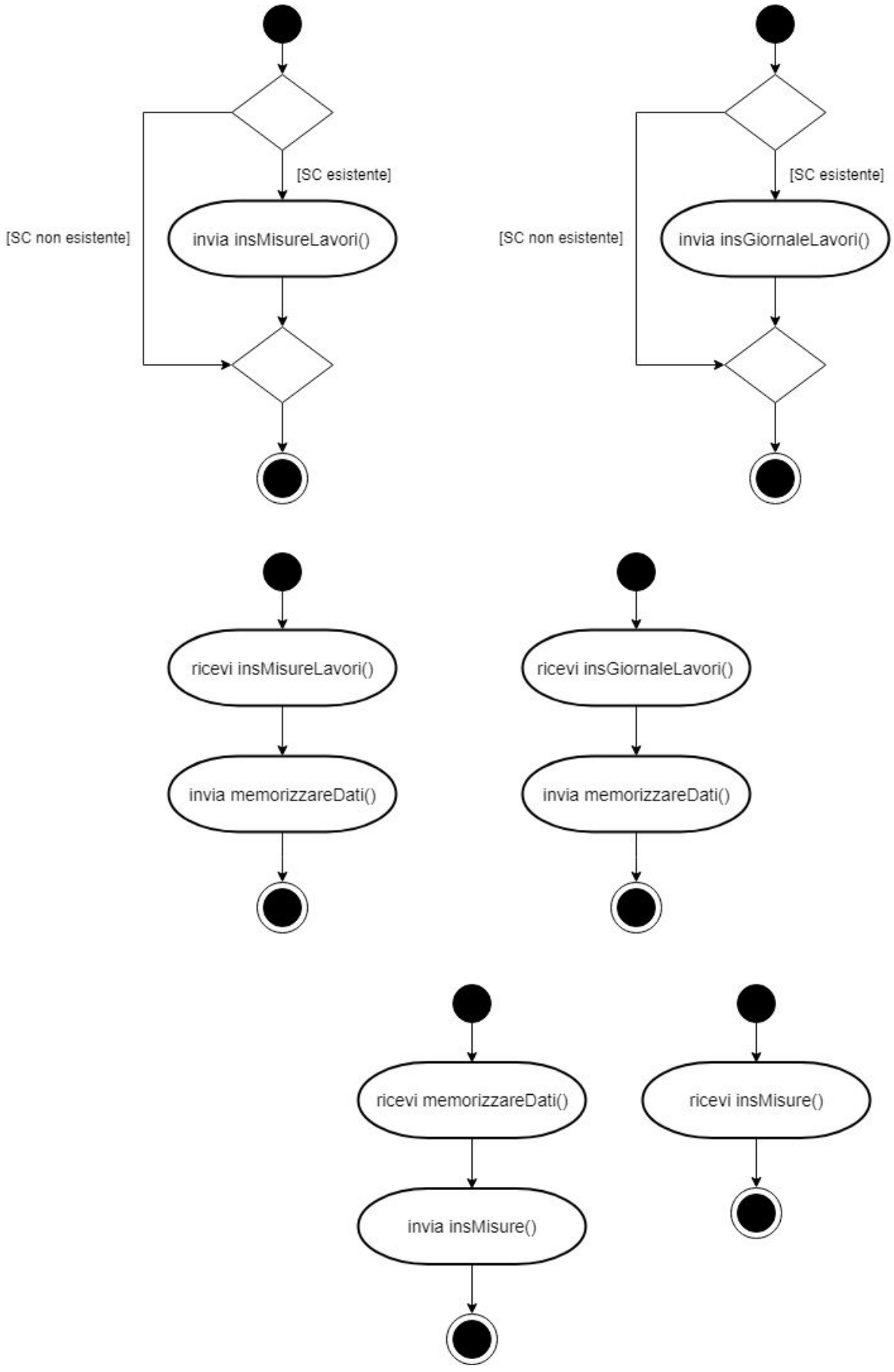


Diagramma delle Attività delle operazioni di inserimento

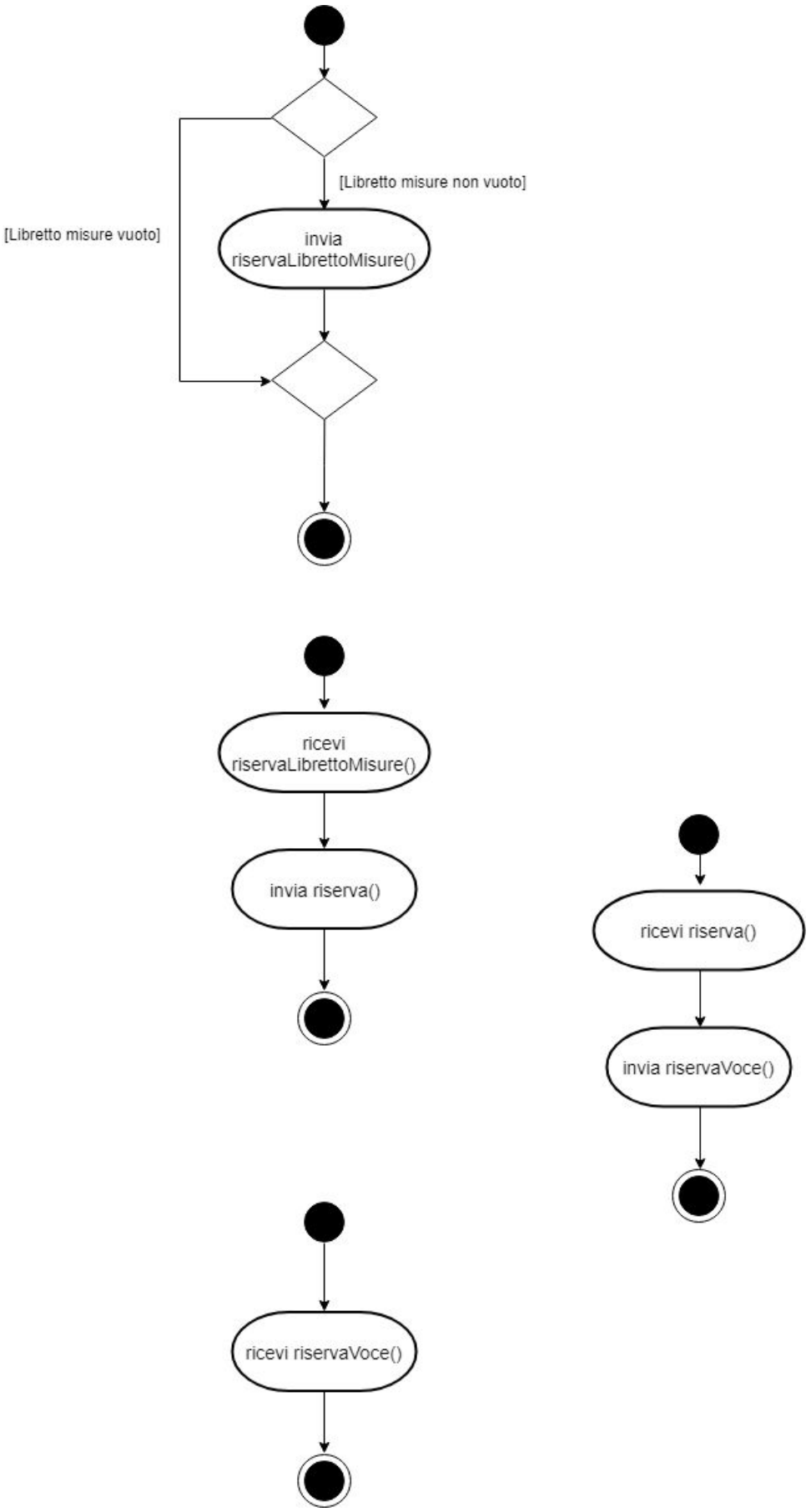


Diagramma delle Attività della riserva

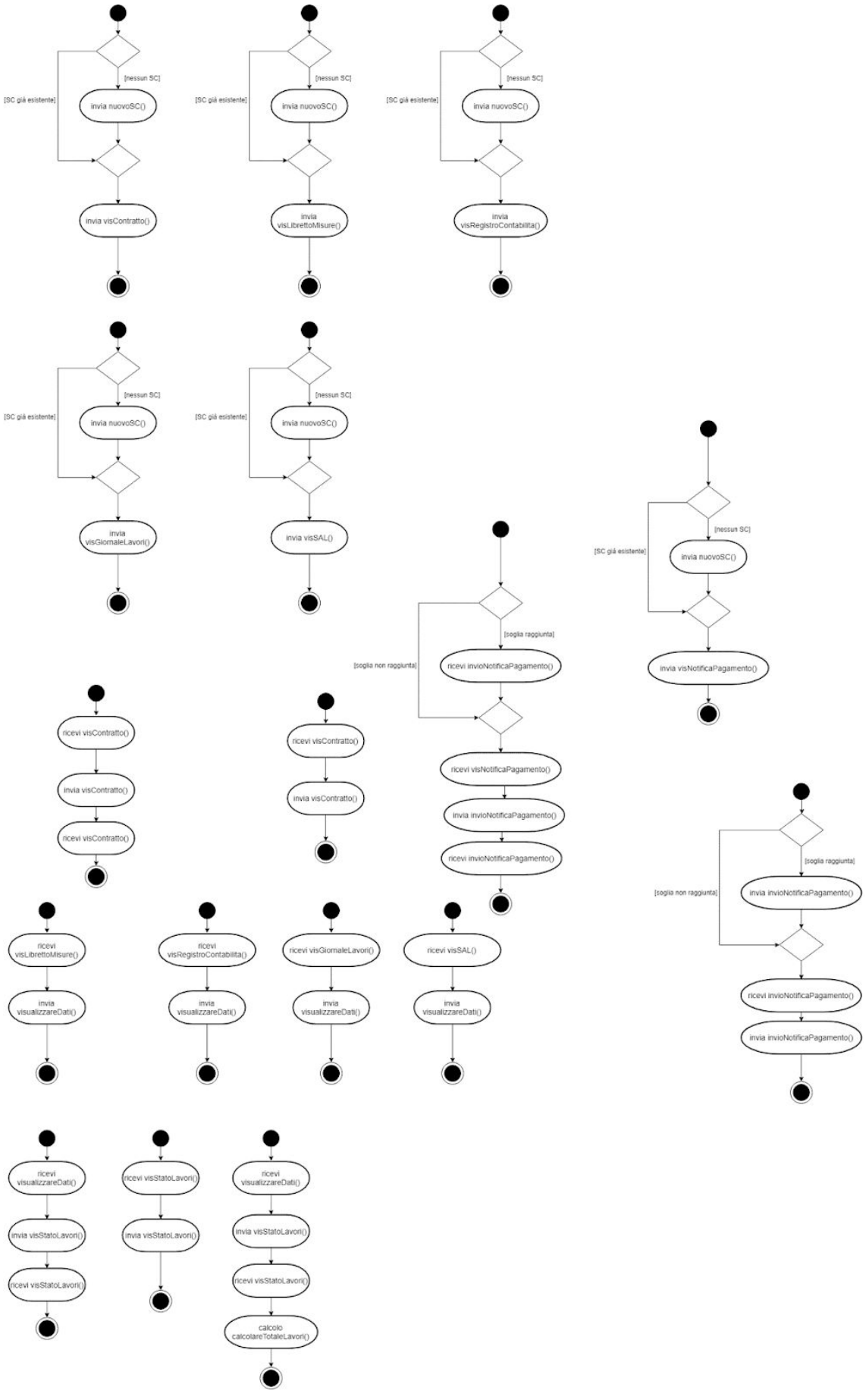


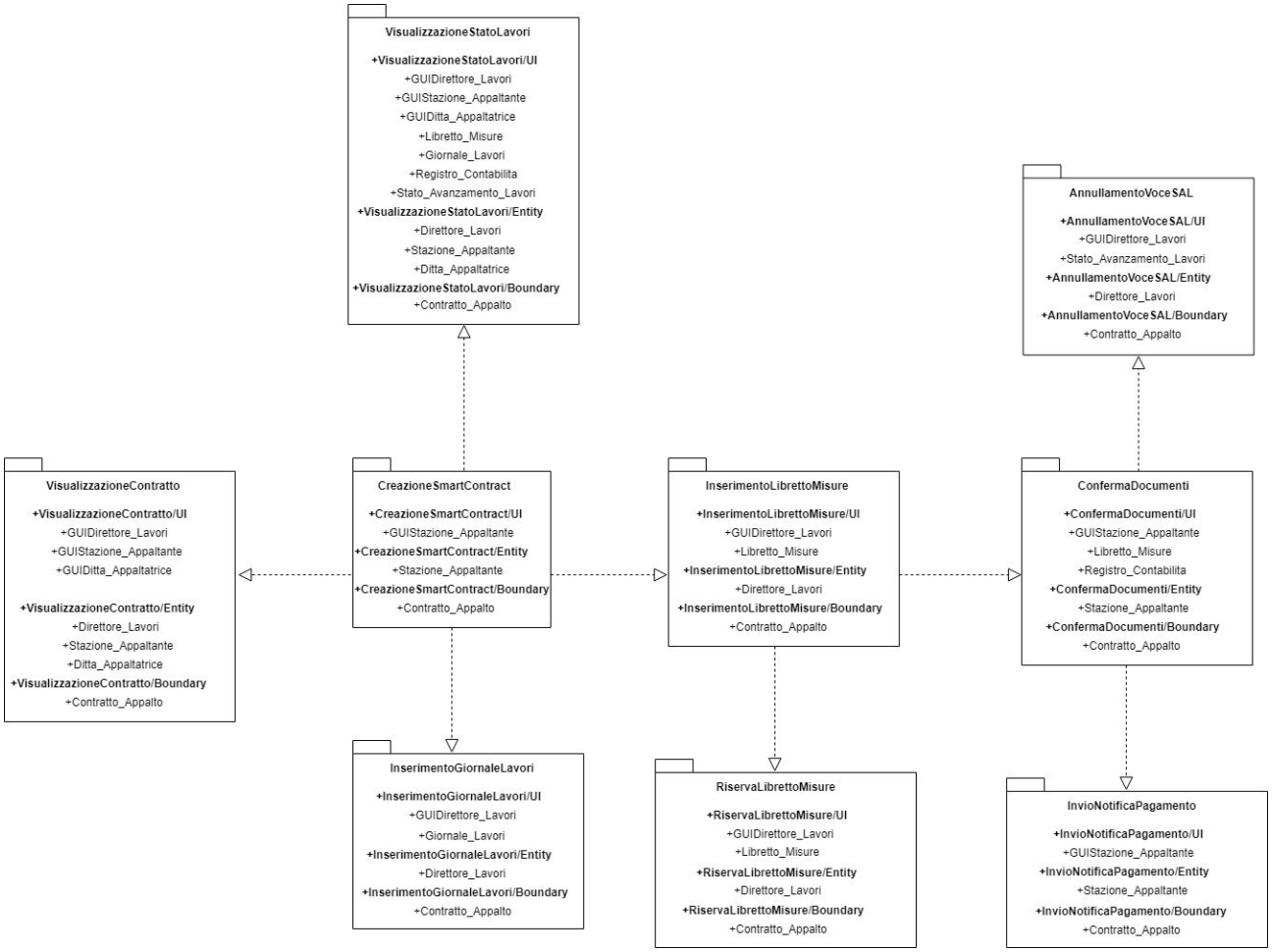
Diagramma delle Attività delle operazioni di visualizzazione

4.3.4 Diagramma dei Package

Un **package** nell'Unified Modeling Language è usato per raggruppare elementi e fornire un namespace per gli elementi raggruppati. Un package può contenere altri package, fornendo così un'organizzazione gerarchica dei package.

Praticamente tutti gli elementi UML possono essere raggruppati in package.
Così classi, oggetti, use case, componenti, nodi, istanze di nodi, ecc. possono essere tutti organizzati come package, consentendo così una maneggevole organizzazione delle miriadi di elementi che un modello UML comporta.

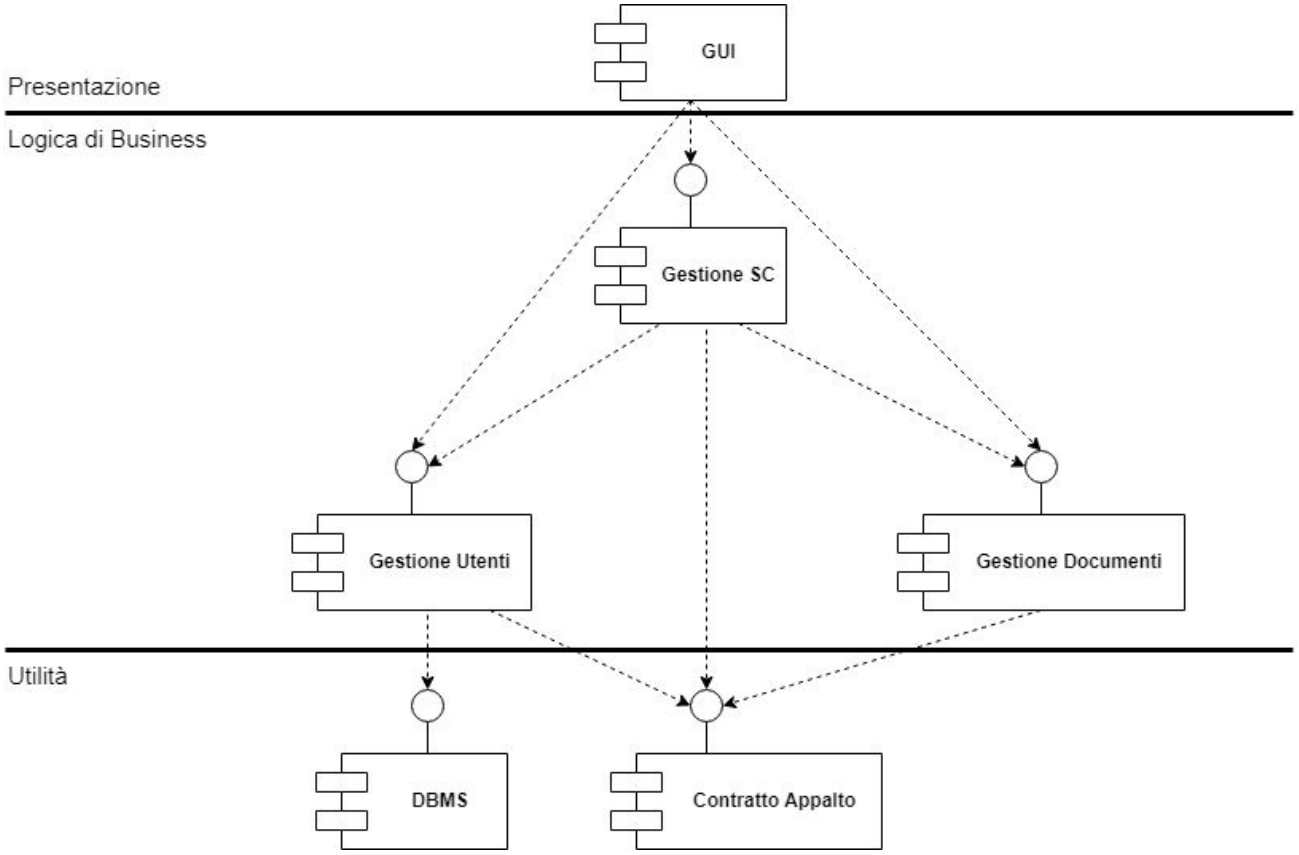
Il modello dei Package ottenuto viene mostrato qui sotto:



4.3.5 Diagramma dei Componenti

Un diagramma dei componenti è un diagramma che ha lo scopo di rappresentare la struttura interna del sistema software modellato in termini dei suoi componenti principali e delle relazioni fra di essi. Per componente si intende una unità software dotata di una precisa identità, nonché responsabilità e interfacce ben definite.

Il diagramma dei componenti del nostro programma è riportato qui di seguito:



5 Tecnologie scelte

5.1 Go-Ethereum

La prima delle tecnologie implementate per la realizzazione del sistema di cui vogliamo parlare è Go-Ethereum che consiste in una delle tre originali implementazioni (C++, Python e Go) del protocollo Ethereum scritto utilizzando il linguaggio Go. Ethereum è una piattaforma distribuita, su una rete di computers, che permette la replicazione e il processamento di dati e di piccoli programmi chiamati Smart Contract (SC) su tutti i computers appartenenti alla rete, senza un coordinatore centrale. Lo scopo è quello di estendere il concetto di Blockchain dal Bitcoin, grazie all'esecuzione degli Smart Contract da parte dei partecipanti sulle loro macchine utilizzando la "Ethereum Virtual Machine". Come Bitcoin anche Ethereum permette la realizzazione di Blockchain, costituite da blocchi di dati (transazioni e Smart Contract). I blocchi vengono creati da un qualche partecipante e distribuiti ai rimanenti che hanno il compito di validarli. Sempre come Bitcoin la rete principale di Ethereum è pubblica e aperta, cioè chiunque può scaricare o scrivere del software per connettersi alla rete per creare transazioni o Smart Contract, validandoli, senza bisogno di registrarsi ad alcuna organizzazione. In ogni caso, però, è possibile creare reti private che sono separate dalla rete principale, per cui l'accesso a tali reti è permesso solo agli utenti autorizzati. Le principali differenze tra Ethereum e Bitcoin sono:

- La maggiore velocità dei blocchi di Ethereum ad essere validati ed aggiunti alla Blocchain, impiegando una decina di secondi al contrario dei circa 60 secondi necessari a Bitcoin;
- La dimensione dei blocchi Ethereum è minore rispetto a quella dei blocchi Bitcoin (1MB), i blocchi Ethereum sono dimensionati in base alla complessità dei contratti eseguiti per cui variano leggermente tra loro (dimensione media di 2KB);
- Il codice che può essere implementato nella Blockchain ed eseguito come Smart Contract è più avanzato di quello che è possibile sviluppare usando Bitcoin.

Come già accennato in precedenza gli Smart Contract sono dei piccoli programmi conservati in una Blockchain Ethereum scritti usando il linguaggio Solidity, inoltre vengono definiti "Turing complete" in quanto sono capaci di eseguire qualsiasi computazione che può essere implementata usando gli altri linguaggi di programmazione. Possono essere eseguiti conferendogli degli Ether (ETH), la valuta di Ethereum. Per implementare uno Smart Contract bisogna creare un account con del codice all'interno e successivamente caricarlo sulla Blockchain all'interno di una transazione. Una volta che il contratto è stato caricato, per essere utilizzato, bisogna creare una transazione contenente un pagamento in ETH per il contratto e se necessario bisogna passargli tutte le informazioni di cui il contratto ha bisogno per la sua esecuzione. Ogni computer (miner) intento nell'attività di estrazione eseguirà lo SC sul proprio computer tramite la loro Ethereum Virtual Machine come parte del loro processo di estrazione e raggiungerà una conclusione riguardo all'output. In teoria, se tutti i miner sono in buna fede, tutti dovrebbero giungere allo stesso risultato

in quanto stanno eseguendo lo stesso codice del contratto con le stesse informazioni che sono state fornite. Quando un blocco viene estratto, il miner vincitore pubblicherà il blocco al resto della rete e gli altri computer valideranno di ottenere lo stesso risultato, aggiungendo poi il blocco alle loro Blockchain. Ed è così che lo stato di una Blockcain Ethereum viene aggiornato.

5.2 Truffle

Truffle è un ambiente di sviluppo e testing framework per le Blockchain che utilizzano la Ethereum Virtual Machine (EVM), con lo scopo di rendere lo sviluppo più semplice e veloce. Tra le varie funzionalità di Truffle le più rilevanti sono:

- La compilazione degli SC, il linking, il deployment e il binary management;
- Il testing automatico degli SC per velocizzare lo sviluppo;
- Framework per la migrazione degli SC.

Nell'ambito del nostro progetto Truffle è stato utilizzato per permettere la migrazione degli SC all'interno della Blockchain creata con Go-Ethereum.

5.3 Django

Django è un web framework con licenza open source per lo sviluppo di applicazioni web, scritto in linguaggio Python, seguendo il paradigma "Model-Template-View". Django fornisce un certo numero di funzionalità che facilitano lo sviluppo rapido di applicazioni per la gestione di contenuti Web. Per esempio, invece che richiedere sviluppatori per la realizzazione di controller e view per le aree di amministrazione di un sito, Django fornisce una soluzione integrata di amministrazione dei contenuti che può essere inclusa come parte di ogni sito basato su Django e che può gestire molti siti con un'unica installazione. L'applicazione per l'amministrazione permette di creare, aggiornare e eliminare contenuti rappresentati da oggetti tenendo traccia di tutte le operazioni effettuate e fornisce un'interfaccia per la gestione di utenti e gruppi di utenti (inclusa la gestione dei permessi). Inoltre fornisce un sistema di commenti, funzionalità per la creazione di feed RSS e/o Atom, "pagine semplici" che permettono di essere gestite senza dover scrivere un controller o una view appositi, e funzionalità di redirezione di URL.

5.4 Database SQLite

SQLite è una libreria software scritta in linguaggio C che implementa un DBMS SQL di tipo ACID incorporabile all'interno di applicazioni, non è un processo standalone utilizzabile di per sé ma può essere incorporato all'interno di un altro programma. Alcune delle caratteristiche offerte dalla libreria sono:

- Compattezza (meno di 500KB per l'intera libreria alla versione 3.6.14);

- La velocità, in molti casi più di MySQL e PostgreSQL;
- Il codice sorgente è liberamente disponibile, chiaro e ben commentato;
- È in grado di interpretare stringhe SQL; a differenza di altre librerie simili, supporta buona parte dello standard SQL92;
- Ha la possibilità di interpretare campi in JSON;
- L'API è semplice da utilizzare;
- Ha transazioni atomiche, consistenti, isolate e durabili (ACID), anche in caso di crash di sistema o blackout;
- È multipiattaforma.

Nel nostro caso è stato necessario per costruire un database a supporto dell'applicazione web sviluppata.

5.5 Librerie Python e JS

In questo paragrafo descriveremo in breve una serie di librerie (Python e JavaScript) utilizzate durante lo sviluppo del codice:

- Web3.py (versione 5.0.0) – è una libreria Python che ci permette di interagire con la Blockchain Ethereum dal back-end dell'applicazione e di usare quindi gli Smart Contract creati;
- py-solc-x (versione 0.5.0) – è una libreria Python che ci permette di compilare gli Smart Contract scritti in Solidity direttamente dal back-end dell'applicazione;
- Jquery (versione 2.2.4) – è una libreria JavaScript per applicazioni web che permette la selezione, la manipolazione, la gestione degli eventi e l'animazione di elementi DOM in pagine HTML, nonché implementare funzionalità AJAX. Nel nostro caso la libreria è stata usata per interagire con gli elementi grafici della pagina web e per mandare richieste AJAX al server inviando e ricevendo dati;
- django-notify-x (versione 0.1.9) – è una libreria Python che ci permette di ottenere un sistema di notifiche simile a quello di facebook per la nostra applicazione web realizzata usando Django. Grazie a questo sistema di notifiche siamo in grado di fornire un sistema di notifiche che permette di memorizzare una grande varietà di notifiche e garantisce ad un utente di ricevere una notifica non appena un'azione viene compiuta.