



Progetto Software Cybersecurity

Docente: Luca Spalazzi

Università Politecnica delle Marche

Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione

Francesco Alborino S1096444

Bodgan Petru Borc S1094507

Michele Marchetti S1096271

Chiara Mazzucchelli S1096449

Lorenzo Scoppolini Massini S1096273

Indice

1	Introduzione	4
2	Progettazione	5
2.1	Analisi dei requisiti	5
2.1.1	Requisiti iniziali	5
2.1.2	Early Requirement Analysis	6
2.1.3	Late Requirement Analysis	8
2.2	Preliminary Risk Assessment	10
2.2.1	Risk identification	10
2.3	Risk analysis	11
2.3.1	Threat identification	11
2.3.2	Abuse Case	13
2.4	Attack Tree	14
2.5	Risk Reduction	20
3	Blockchain e Smart Contracts	25
3.1	Blockchain	25
3.2	Smart Contract	27
4	Implementazione	29
4.1	Upload di un'immagine	29
4.2	Visualizzare immagini e misure	29
4.3	Dettagli implementativi	31
5	Guida all'Utilizzo	35
5.1	Eseguire l'autenticazione	35
5.2	Caricare l'immagine nella blockchain	35
5.3	Visualizzare la galleria e la singola immagine nella blockchain	35
6	Appendici	38
6.1	Linguaggio di modellazione I^*	38
6.2	Triadi per le security policy	39
6.3	Tecnologie utilizzate	39
6.3.1	Node.js	40
6.3.2	IPFS	40
6.3.3	MongoDB	40
6.3.4	Flutter	41
6.3.5	Quorum	41
6.3.6	Solidity	41
6.3.7	Truffle	42

Elenco delle figure

1	Workflow del Preliminary Risk assessment	5
2	Diagramma I^* rappresentante lo Strategic Dependency Model, creato in fase di Early Requirement Analysis	6
3	Diagramma I^* rappresentante lo Strategic Rationale Model, creato in fase di Early Requirement Analysis	7
4	Diagramma I^* rappresentante lo Strategic Rationale Model, creato in fase di Late Requirement Analysis	9
5	Abuse Case: diagramma I^* rappresentante i possibili attacchi da parte di un utente esterno	17
6	Abuse Case: diagramma I^* rappresentante i possibili attacchi da parte di un utente interno al software del direttore dei lavori	18
7	Abuse Case: diagramma I^* rappresentante i possibili attacchi da parte di un utente interno al software del drone	18
8	Attack tree relativi all'asset <i>misure</i>	19
9	Attack tree relativi all'asset <i>giornale</i>	20
10	Attack tree relativi all'asset <i>immagine</i>	21
11	Attack tree relativi all'asset <i>ottiene misure</i>	21
12	Workflow di una blockchain	25
13	Workflow del upload di un'immagine	30
14	Workflow della visualizzazione di un'immagine	31
15	Schema dei dati all'interno del database	34
16	Finestra di login del direttore dei lavori	35
17	Interfaccia del drone simulato	36
18	Galleria delle immagini nella blockchain	36
19	Dettagli e misure di una immagine all'interno della blockchain	37

Elenco delle tabelle

1	Use Case Specification dell'asset <i>Misure</i>	11
2	Use Case Specification dell'asset <i>Immagine</i>	12
3	Use Case Specification dell'asset <i>Giornale</i>	13
4	Use Case Specification dell'asset <i>Ottieni misure</i>	14
5	Asset Value Assessment	15
6	Minacce associate agli asset identificati, secondo il modello <i>Stride</i>	16
7	Valutazione delle probabilità di attacco	19
8	Metriche per la valutazione qualitativa dei rischi	22
9	Matrice probabilità-impatto	22
10	Azioni consigliate per ogni valore di rischio	23
11	<i>Mitigation Matrix</i>	24

1 Introduzione

Il seguente progetto *Software Cybersecurity* tratta la realizzazione di un software per PMI agenti nel mercato dell'edilizia. L'obiettivo del software è quello di notarizzare le immagini provenienti da un cantiere edile. A questo scopo nel cantiere vengono effettuate le operazioni di acquisizione delle immagini per ottenere le misure, controllarle e documentarle. Spesso le immagini sono ottenute tramite le riprese di uno o più droni, queste sono mandate in seguito a un servizio di fotogrammetria attraverso il quale vengono calcolate delle misure geometriche. Tutte le immagini acquisite e le relative misure vengono salvate nel Giornale dei Lavori dove saranno disponibili per essere consultate in seguito dal direttore dei lavori.

Per ottenere delle buone prestazioni in ambito di sicurezza del software, deve essere garantita:

- l'autenticità delle immagini, ovvero dove e quando queste sono state acquisite;
- l'autenticità delle misure e a quali immagini si riferiscono;
- l'integrità delle immagini e delle misure;
- il non ripudio della paternità di immagini e misure

Nelle seguenti pagine verrà mostrata la progettazione e l'implementazione di un software in grado di fare quanto richiesto. Il lavoro è suddiviso come segue: nel Capitolo 2 vengono analizzati i requisti che il software dovrà rispettare, il Capitolo 3 contiene una breve introduzione alle Blockchain e agli smart contract spiegando anche come questi verranno utilizzati nel progetto, nel Capitolo 4 è stata seguita l'implementazione del software mentre nel Capitolo 5 è mostrata la guida all'utilizzo con le interfacce utente. Infine il Capitolo 6 conclude il tutto con un approfondimento sulle metodologie e le tecnologie utilizzate.

2 Progettazione

In questo capitolo è affrontata la progettazione del software per la gestione del giornale dei lavori. I passi seguiti sono mostrati nella Figura 1: si parte con l'analisi dei requisiti, si identificano gli asset e i relativi rischi ad essi associati, si identificano le azioni per poterli arginare, con l'obiettivo di trovare una soluzione che renda il software sicuro, bilanciando costi ed affidabilità.

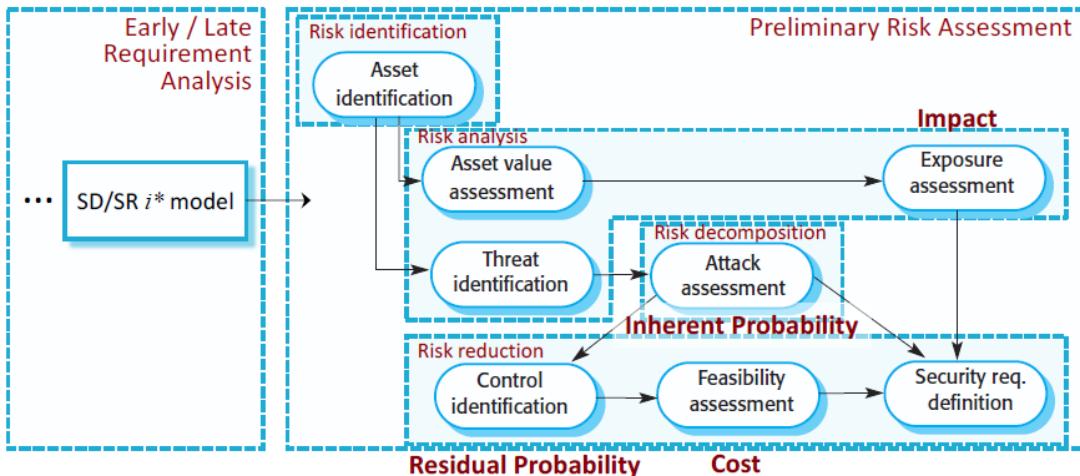


Figura 1: Workflow del Preliminary Risk assestment

2.1 Analisi dei requisiti

2.1.1 Requisiti iniziali

Questo progetto è incentrato sulla realizzazione del giornale dei lavori in formato digitale. Quest'ultimo è un documento contabile con cui monitorare passo dopo passo l'andamento tecnico ed economico di un'opera, in questo caso specifico di un'opera edilizia. Il responsabile di tale giornale è il direttore dei lavori, che ha la responsabilità del controllo contabile amministrativo e redige gli atti contabili, utili per poter monitorare l'avanzamento dei lavori. In generale, vengono considerati atti contabili il giornale dei lavori, il libretto delle misure delle lavorazioni e delle provviste, il registro della contabilità, lo stato di avanzamento dei lavori e i certificati di pagamento.

L'ambito specifico di cui si occupa questo progetto è quello edilizio, che presenta delle particolarità in quanto, tra le azioni che vengono eseguite in cantiere edile, c'è l'acquisizione di immagini, con l'obiettivo di misurare, monitorare, controllare e documentare l'avanzamento dei lavori. Per fare ciò, vengono spesso impiegati dei droni in quanto permettono di acquisire immagini dall'alto. Tali immagini vengono poi elaborate da un servizio di fotogrammetria, che impiega una tecnica di misurazione che permette, a partire da un'immagine, di ricavarne dati utili ai lavori. Le immagini fotografate dal drone devono essere salvate nel giornale dei lavori, mentre le misure ottenute dovrebbero essere riportate nel libro delle misure, ma in questo progetto, per semplicità, verranno anch'esse inserite nel giornale dei lavori.

Il progetto si concentra proprio su questo aspetto, ovvero quello di gestire il salvataggio delle immagine fornite dal drone e le relative misure. Essendo un documento contabile ufficiale, ci sono dei requisiti e delle

norme che il giornale dei lavori deve necessariamente rispettare. E', innanzi tutto, necessario verificare l'autenticità delle immagini, quindi deve essere nota la loro provenienza, in particolare il dove e il quando sono state acquisite. Nell'ambito di questo progetto non viene trattata l'acquisizione e la ricezione di immagini e dati relativi alla loro provenienza, ma queste informazioni vengono considerate noti a priori, per cui si parte dal presupposto che l'immagine e i relativi dati forniti siano autentici. Lo stesso discorso vale per i dati metrici forniti dal servizio di fotogrammetria, di cui si vuole conoscere dove e quando sono stati calcolati e a partire da quali immagini.

Un altro requisito importante da rispettare è l'integrità di immagini, dati e misure, ovvero niente di ciò che viene riportato sul giornale dei lavori deve poter essere modificato o distrutto. Come requisito non funzionale viene richiesto l'uso di un servizio di fotogrammetria specifico. In questo progetto non viene utilizzato il software professionale, che permetterebbe di elaborare le immagini provenienti dal drone e di produrre in output una notevole quantità di dati metrici da riportare nel giornale dei lavori. È adottato invece un software free che genera delle altre informazioni a partire dall'immagine data come dei tags che indicano la probabilità della presenza di alcuni oggetti nelle immagini.

Per modellare i requisiti richiesti è stato inizialmente utilizzato il linguaggio di modellazione I^* , la cui spiegazione è riportata nell'Appendice 6.1.

2.1.2 Early Requirement Analysis

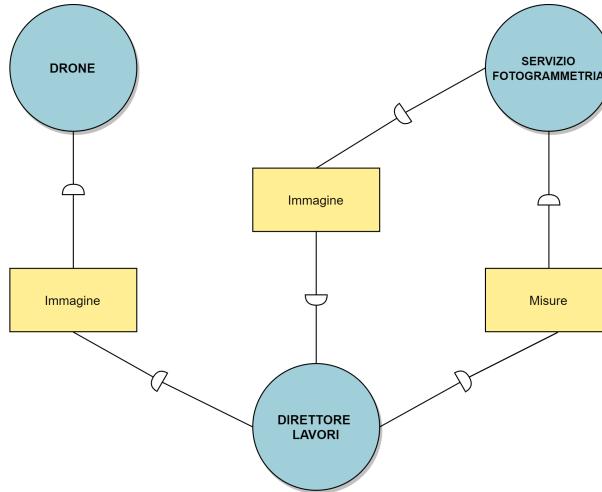


Figura 2: Diagramma I^* rappresentante lo Strategic Dependency Model, creato in fase di Early Requirement Analysis

Per poter comprendere gli attori coinvolti nel processo e i legami presenti, si ricorre alla *Early Requirement Analysis*, letteralmente *Analisi Preliminare dei Requisiti*. In questa fase si comprende il problema che si sta affrontando, studiando e analizzando il caso reale, in modo tale da individuare i componenti principali che hanno un ruolo chiave nel processo preso in analisi. Il risultato sono dei diagrammi I^* che mettono in evidenza gli attori rilevanti e le loro dipendenze per generare o ottenere risorse e per raggiungere un determinato obiettivo. Si creano, in particolare, due diagrammi: lo *Strategic Dependency model* (SD) e lo *Strategic Rationale model* (SR).

L'SD descrive una rete di relazioni di dipendenza tra vari attori in un contesto organizzativo, utilizzando un insieme di nodi, che rappresentano gli attori stessi, e collegamenti che mostrano le dipendenze tra attori.

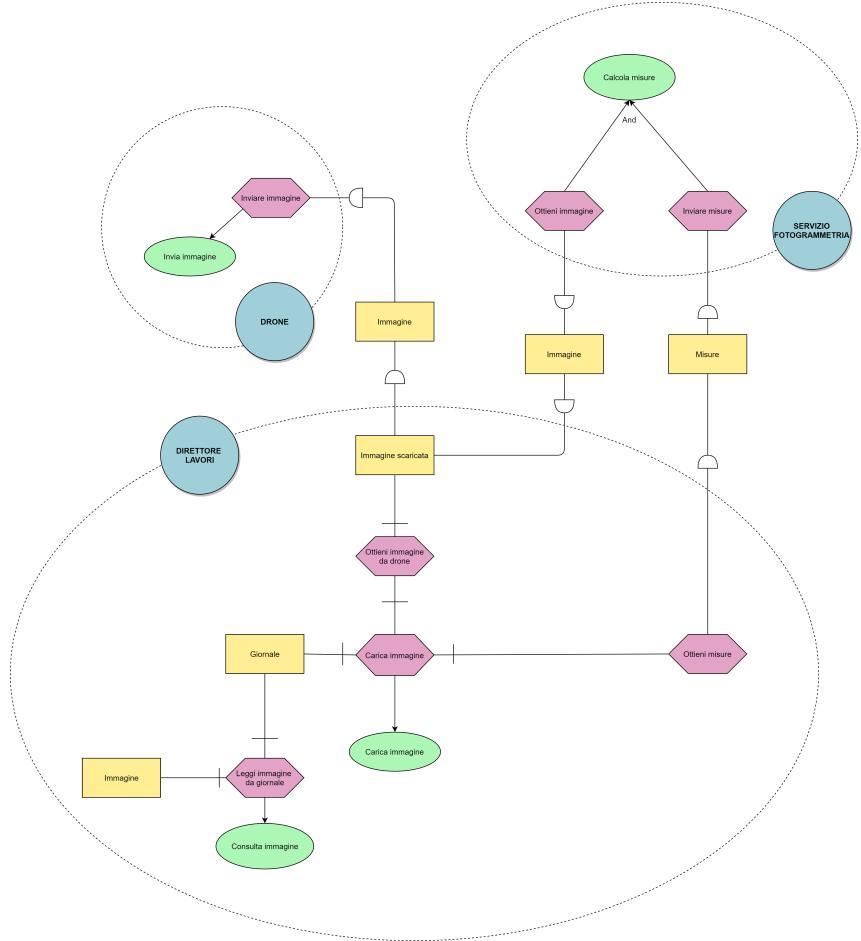


Figura 3: Diagramma I^* rappresentante lo Strategic Rationale Model, creato in fase di Early Requirement Analysis

L'SR permette di modellare le ragioni associate a ciascun attore e le sue dipendenze e fornisce informazioni su come gli attori raggiungono i loro obiettivi. Questo modello include solo gli elementi che vengono considerati abbastanza importanti da influenzare i risultati di un obiettivo. Il modello SR mostra le dipendenze degli attori includendo, al suo interno, il modello SD. In particolare, il modello SR specifica gli obiettivi, i softgoal, i compiti e le risorse, quindi, rispetto al modello SD, il modello SR fornisce un livello di analisi più dettagliato.

I diagrammi SD e SR riguardanti questo progetto sono mostrati, rispettivamente, in Figura 2 e Figura 3. Da tali diagrammi si può notare che ci sono tre attori :

- *Direttore dei lavori*: è il capo del cantiere e il responsabile del corretto aggiornamento del giornale dei lavori
- *Drone*: strumento utilizzato per l'acquisizione delle immagini del cantiere

- *Servizio di fotogrammetria*: servizio software di terzi utilizzato per ricavare le misure utili ai lavori del cantiere

Come si può notare nello schema SR (Figura 3) il direttore dei lavori è l'unico utente che ha i permessi per caricare una immagine sul giornale dei lavori e successivamente poterla consultare assieme alle sue misure. Queste, come mostrato nel grafico vengono calcolate dal servizio di fotogrammetria. Il drone ha come scopo unico quello di inviare le immagini acquisite al direttore dei lavori

2.1.3 Late Requirement Analysis

Questa fase di analisi si basa sull'introduzione del software all'interno del sistema, in quanto tutte le azioni vengono effettivamente compiute da un software. Quest'ultimo viene considerato come un attore e vengono creati i legami di dipendenza con gli altri componenti. In questa fase si genera un diagramma I^* , ovvero lo Strategic Rationale Model mostrato in Figura 4, creato a partire dallo SR presentato nell'early analysis (Figura 3) e aggiungendo il software come attore. Nel diagramma il software è stato scomposto in tre componenti più piccoli per poter analizzare con più dettaglio le dipendenze e i legami presenti:

- **Software del direttore dei lavori**: componente che si occupa della fase di caricamento dell'immagine sul giornale con le relative misure ottenute dal software del servizio di fotogrammetria.
- **Software del drone**: componente che si occupa di inviare al software del direttore dei lavori le immagini acquisite dal drone
- **Software del servizio di fotogrammetria**: componente che si occupa di ricevere l'immagine per consegnarla al servizio di fotogrammetria per calcolarne le misure metriche. Successivamente restituisce le misure ottenute al software del direttore dei lavori.

Con l'introduzione di tali software si può notare che il direttore dei lavori si occupa solamente della lettura e della visualizzazione delle informazioni contenute all'interno del giornale dei lavori. In questo modo le operazioni di caricamento delle immagini con le rispettive misure non sono più di sua responsabilità ma vengono svolte dal software stesso, riducendo così le possibili azioni da parte dell'utente.

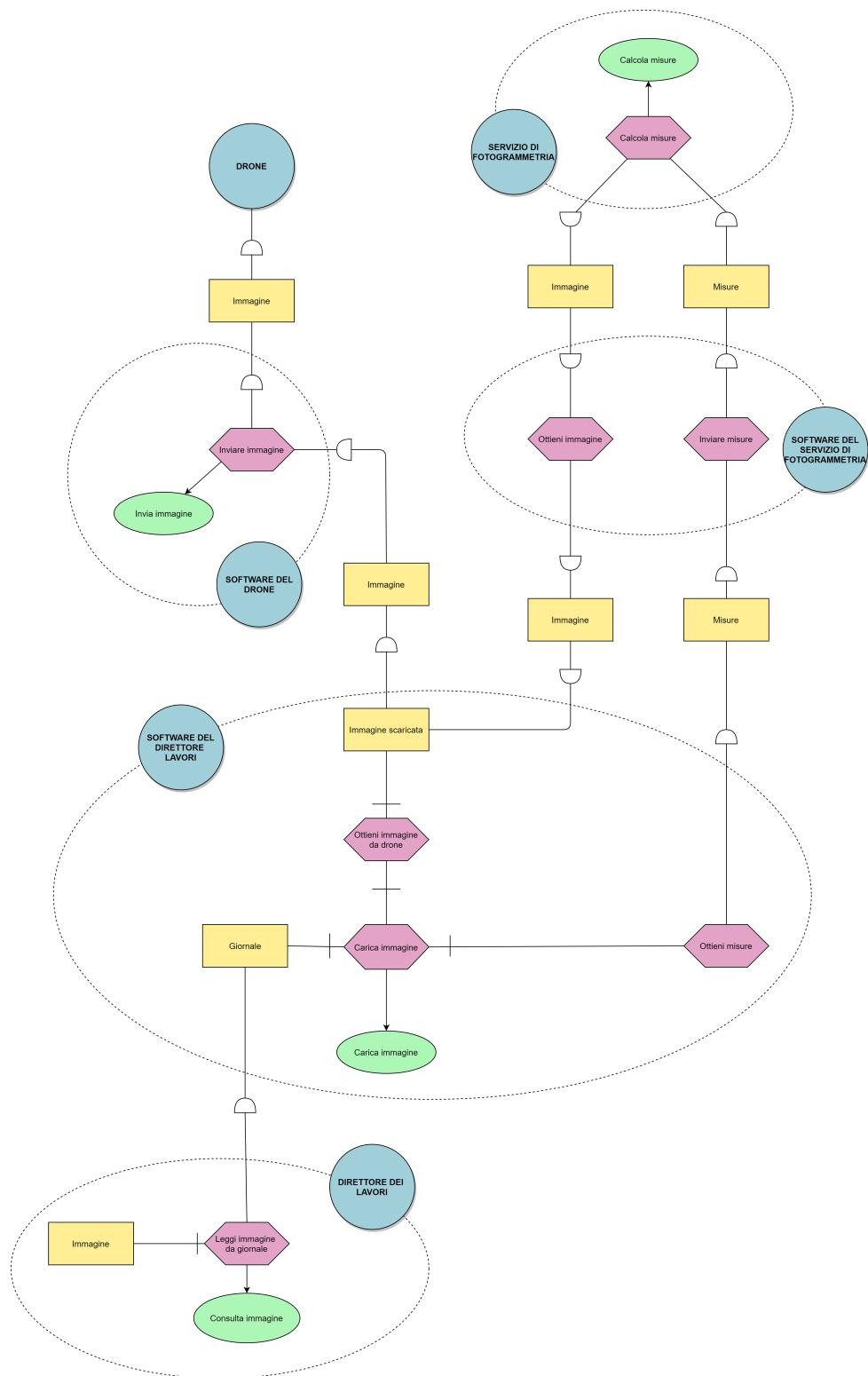


Figura 4: Diagramma I^* rappresentante lo Strategic Rationale Model, creato in fase di Late Requirement Analysis

2.2 Preliminary Risk Assessment

In questa fase si individuano gli asset del sistema con le relative policy di sicurezza e organizzative per poi essere in grado di analizzare i rischi a cui sono esposte.

Dall'analisi dei diagrammi I^* si è in grado di identificare gli asset del sistema. Questa identificazione permette di approfondire al meglio le misure di sicurezza da apportare nei confronti di questi asset, per rendere il software sicuro. Come mostrato nelle Tabelle 1, 2, 3 e 4 gli asset identificati sono quattro:

- **Misure:** sono i dati metrici relativi a un'immagine, ottenuti tramite un servizio di fotogrammetria
- **Immagine:** foto scattata dal drone al cantiere, con relativi metadati
- **Giornale:** documento che contiene immagini e relative misure, per tenere traccia dell'avanzamento dei lavori
- **Ottieni misure:** processo che permette di richiedere le misure di un'immagine al servizio di fotogrammetria

2.2.1 Risk identification

Nelle Tabelle 1, 2, 3 e 4 sono riportati, oltre alla descrizione degli asset, i loro flussi all'interno dell'applicazione, i loro requisiti non funzionali e le policy organizzative (utili per comprendere le operazioni che gli utenti possono o non possono effettuare nei confronti di uno specifico asset). Queste descrizioni più accurate sono utilizzate per poter identificare i rischi legati a ciascun asset, con l'obiettivo di trattarli e mitigarli.

Gli asset possono essere sia tangibili che intangibili. In questo caso sono stati individuati come asset tangibili *immagine*, *misure* e *giornale*, che sono elementi estremamente importanti, in quanto presenti in tutte interazioni emerse dai diagrammi I^* . *Ottienere risorse* è invece un asset non tangibile poiché è un bene immateriale, ma comunque non meno importante in quanto si occupa della corretta ricezione delle misure. Per quanto riguarda le security policy, sono stati utilizzati i requisiti delle triadi conosciute, riassunti nell'Appendice 6.2.

USE CASE NAME: MISURE	
Actors	Software del direttore dei lavori, Software del servizio di fotogrammetria, Servizio di fotogrammetria
Description	Misure relative a un'immagine richieste dal software del direttore dei lavori al servizio di fotogrammetria
Data	Misure
Stimulus and Preconditions	Le misure vengono calcolate dal servizio di fotogrammetria. Le misure possono essere richieste dal software del direttore dei lavori
Basic Flow	<ol style="list-style-type: none"> 1. Il servizio di fotogrammetria calcola le misure 2. Il software del servizio di fotogrammetria invia le misure al software del direttore dei lavori 3. Il software del direttore dei lavori aggiorna il giornale dei lavori, salvandoci le misure
Exception Flow	<ol style="list-style-type: none"> 1. Il software del servizio di fotogrammetria invia le misure al software del direttore dei lavori 2. Il software del direttore dei lavori non riceve le misure dal servizio di fotogrammetria
Response and Postconditions	Il software del direttore dei lavori ottiene le misure
Security policy	Disponibilità, Responsabilità
Organization policy	Solo il software del direttore dei lavori ha l'autorizzazione a richiedere le misure relative ad un'immagine e a salvarle nel giornale dei lavori

Tabella 1: Use Case Specification dell'asset *Misure*

2.3 Risk analysis

Dopo aver individuato tutti gli asset con le relative policy di sicurezza, è opportuno esprimere il valore rappresentato da ogni policy qualitativamente per capire come agire e su quali policy concentrarsi maggiormente. Per adoperare una rappresentazione qualitativa è stato necessario indicare il valore di ogni policy di sicurezza tramite una scala di Likert a 5 elementi. Queste informazioni sono rappresentate nella Tabella 5, che rappresenta l'asset value assessment dove accanto al valore di ogni policy è riportata una breve considerazione che ne motiva la scelta. Il valore individuato per ogni policy sarà poi utilizzato per valutare l'impatto dei rischi all'interno della Mitigation Matrix in Tabella 11.

2.3.1 Threat identification

Dopo aver individuato gli asset e le relative policy da rispettare, si può procedere con l'identificazione delle minacce che insorgono legate alla violazione di questi requisiti di sicurezza. Questa analisi viene condotta utilizzando il metodo *Stride*, che consiste nell'associare una minaccia ad ogni obiettivo di sicurezza. Il modello *Stride* prevede le seguenti minacce:

- *Spoofing*: violazione dell'autenticazione
- *Tampering*: violazione dell'integrità
- *Repudiation*: violazione del non ripudio, per cui un utente potrebbe negare di aver compiuto un'azione svolta

USE CASE NAME: IMMAGINE	
Actors	Drone, Software del drone, Direttore dei lavori, Servizio fotogrammetria, Software del servizio di fotogrammetria
Description	<p>A. L'immagine può essere inviata dal software del drone al software del direttore dei lavori</p> <p>B. L'immagine può essere inviata dal software del direttore dei lavori al software del servizio di fotogrammetria</p> <p>C. Il servizio di fotogrammetria estrae le misure dall'immagine</p> <p>D. L'immagine può essere visualizzata nel giornale dei lavori dal direttore dei lavori</p>
Data	Immagine
Stimulus and Preconditions	<p>A. L'immagine deve essere scattata dal drone</p> <p>B. Il software del direttore dei lavori richiede le misure associate all'immagine</p> <p>C. Il servizio di fotogrammetria riceve l'immagine che deve elaborare</p> <p>D. Il direttore dei lavori ha bisogno di consultare il giornale dei lavori</p>
Basic Flow	<p>A. L'immagine viene inviata dal software del drone al software del direttore dei lavori</p> <p>B. L'immagine può essere inviata dal software del direttore dei lavori al software del servizio di fotogrammetria; il servizio di fotogrammetria calcola le misure relative all'immagine</p> <p>C. Il software del direttore dei lavori salva l'immagine sul giornale dei lavori</p> <p>D. L'immagine viene consultata dal direttore dei lavori</p>
Exception Flow	<p>A. L'immagine viene inviata dal software del drone al software del direttore dei lavori; l'immagine non viene ricevuta dal software del direttore dei lavori</p> <p>B. L'immagine può essere inviata dal software del direttore dei lavori al software del servizio di fotogrammetria; il servizio di fotogrammetria non è disponibile</p> <p>C. L'immagine può essere recuperata nel giornale dei lavori dal direttore dei lavori; il giornale dei lavori è compromesso</p>
Response and Postconditions	L'immagine viene caricata sul giornale dei lavori
Security policy	Disponibilità, Responsabilità
Organization policy	Solo il software del direttore dei lavori può aggiornare o modificare le immagini sul giornale dei lavori. Solo il direttore dei lavori può consultare le immagini dal giornale dei lavori. All'immagine devono essere associati i relativi metadati tra cui data e luogo di cattura

Tabella 2: Use Case Specification dell'asset *Immagine*

- *Information disclosure*: violazione della confidenzialità
- *Denial of service*: violazione della disponibilità
- *Elevation of privilege*: violazione dell'autorizzazione

Inoltre sono state considerate anche ulteriori minacce come:

USE CASE NAME: GIORNALE	
Actors	Software del direttore dei lavori, Direttore dei lavori
Description	Il software del direttore dei lavori salva periodicamente immagini (e metadati) con le relative misure provenienti dal servizio di fotogrammetria, per tenere traccia dell'avanzamento dei lavori
Data	Immagini, misure
Stimulus and Preconditions	Il software del direttore dei lavori riceve l'immagine dal software del drone e le relative misure dal software del servizio di fotogrammetria
Basic Flow	<ol style="list-style-type: none"> 1. Il software del direttore dei lavori invia l'immagine (con metadati) e le relative misure al giornale 2. Il direttore dei lavori richiede di visualizzare un'immagine con le relative misure
Exception Flow	<ol style="list-style-type: none"> 1. Non arrivano immagini dal drone 2. Il software del direttore dei lavori chiede le misure e le misure non vengono restituite 3. Il software del direttore dei lavori carica l'immagine ma questa non viene salvata 4. Il direttore dei lavori vuole visualizzare un'immagine ma non riesce a farlo
Response and Postconditions	<ol style="list-style-type: none"> 1. Conferma che l'immagine è stata caricata correttamente nel giornale 2. Informare il direttore nel caso ci siano problemi con il calcolo delle misure 3. Informare il direttore nel caso ci fosse un malfunzionamento del drone
Security policy	Integrità, Disponibilità, Autenticità, Responsabilità
Organization policy	Il giornale deve fornire i suoi servizi 24/7. Solo il direttore dei lavori può consultare il giornale dei lavori. Solo il software del direttore dei lavori può aggiornare e modificare il giornale dei lavori.

Tabella 3: Use Case Specification dell'asset *Giornale*

- *Danger*: violazione della safety
- *Unreliability*: violazione dell'affidabilità
- *Absence of resilience*: violazione della resilienza

Nella Tabella 6 vengono riportare le minacce identificate per ciascun asset, secondo il modello *Stride*, e gli attacchi associati a tali minacce.

Come si può notare dalla tabella, tra le minacce più ricorrenti si ha la violazione dell'integrità che potrebbe potenzialmente colpire tutti e quattro gli asset in quanto un eventuale cyber attacco o un guasto del sistema andrebbe ad intaccare l'integrità dei dati e delle informazioni. Inoltre, un'altra minaccia che può potenzialmente colpire tutti gli asset è la violazione della disponibilità, in quanto il sistema deve essere sempre attivo.

2.3.2 Abuse Case

Questa fase ha l'obiettivo di capire come un soggetto è portato a violare un obiettivo di sicurezza. In particolare, l'*Abuse Case* descrive come un eventuale attaccante possa compiere delle azioni malevoli per

USE CASE NAME: OTTIENI MISURE	
Actors	Software del servizio di fotogrammetria, Software del direttore dei lavori
Description	E' il processo che permette al software del direttore dei lavori di ottenere le misure dal software del servizio di fotogrammetria
Data	Misure
Stimulus and Preconditions	L'immagine deve essere in possesso del software del direttore dei lavori e deve essere inviata al software del servizio di fotogrammetria
Basic Flow	<ol style="list-style-type: none"> 1. L'immagine deve essere in possesso del software del direttore dei lavori 2. L'immagine deve essere inviata al software del servizio di fotogrammetria 3. Le misure devono essere calcolate dal servizio di fotogrammetria
Exception Flow	<ol style="list-style-type: none"> 1. L'immagine viene inviata dal software del direttore dei lavori al software del servizio di fotogrammetria, ma questa non viene ricevuta 2. Le misure vengono calcolate dal servizio di fotogrammetria ma non vengono ricevute dal software del direttore dei lavori
Response and Postconditions	Il software del direttore dei lavori ottiene le misure dell'immagine inviata
Security policy	Disponibilità, Responsabilità
Organization policy	Solo il software del direttore dei lavori può inviare l'immagine per ricevere le misure

Tabella 4: Use Case Specification dell'asset *Ottieni misure*

arrivare a violare uno o più obiettivi di sicurezza. In Figura 5 è riportato il modello I^* che rappresenta la possibile presenza di attaccanti esterni nei confronti di tutti gli asset, mentre nelle Figure 6 e 7 sono riportati, rispettivamente, i possibili attacchi condotti da un utente interno al software del direttore dei lavori e a quello del drone.

Non sono state prese in considerazione le vulnerabilità relative al servizio di fotogrammetria in quanto nella fase di sviluppo non è stato possibile integrarlo. Nel prodotto finale è stata impiegata una API alternativa sostitutiva con l'unico scopo di simularne il funzionamento.

Tutte le possibili azioni che possono essere condotte dall'attaccante e che violano i requisiti di sicurezza sono state inserite nel diagramma attraverso delle goal dependency: il legame tra i vari attori è espresso attraverso dei flussi di dipendenza identificati da degli obiettivi che, in questo caso, sono riferiti alla violazione delle policy di sicurezza.

Questa progettazione non prevede l'analisi del *Misuse Case*, ovvero l'analisi dei possibili attacchi involontari causati da un utente "sbadato", in quanto il direttore dei lavori può solo leggere dal giornale dei lavori e, quindi, non può compiere alcuna azione che conduce a un errore involontario del sistema.

2.4 Attack Tree

Le analisi appena condotte sono state essenziali per la costruzione degli *Attack Tree*, ovvero dei diagrammi che mostrano come un asset o, in generale, un obiettivo possa essere attaccato. In particolare, sono dei diagrammi multi-livello che permettono di visualizzare il flusso di attacco: la radice contiene il potenziale attacco e i suoi figli contengono le condizioni che portano al suo verificarsi. Inoltre, come si può vedere dalle figure, negli archi sono state riportate delle percentuali che rappresentano la probabilità del verificarsi di

Asset	Policy	Valore	Considerazione
Misure	Integrità	5	L'integrità delle misure è un fattore molto importante, in quanto una volta acquisite non dovranno più essere modificate.
	Autenticità	3	L'autenticità delle misure, ovvero la loro correttezza, è necessaria allo scopo del nostro progetto.
	Disponibilità	2	Le misure, una volta ottenute, potranno essere visualizzate in qualsiasi momento.
Giornale	Integrità	5	Il giornale dei lavori, essendo un documento ufficiale non potrà mai essere modificato, saranno possibili solo aggiunte.
	Autenticità	4	Le informazioni riportate dovranno essere autentiche, quindi una volta inserite se ne accerta l'autenticità e non possono essere modificate.
	Disponibilità	3	Le informazioni contenute nel giornale dei lavori devono poter essere accedute in breve tempo e restare disponibili ad ulteriori visualizzazioni.
	Responsabilità	3	Gli utenti che svolgono attività sul giornale dovranno sempre essere tracciati.
Immagine	Disponibilità	3	L'immagine dovrà sempre essere disponibile.
	Responsabilità	4	L'autore dell'inserimento di una immagine dovrà sempre essere tracciabile.
	Integrità & Affidabilità	\	Il software relativo al drone è lasciato in gestione a terze parti alle quali si trasferisce la responsabilità sulle garanzie di integrità e affidabilità.
Ottieni Misure	Disponibilità	2	Deve essere sempre possibile richiedere al servizio dei fotogrammetria l'ottenimento delle misure dalle immagini.
	Integrità & Affidabilità	\	Utilizzando un servizio di fotogrammetria a pagamento di terze parti, si trasferisce la responsabilità sulle garanzie di integrità e affidabilità a questi ultimi.
	Responsabilità	4	La richiesta per l'ottenimento delle misure di una immagine dovrà sempre essere tracciata.

Tabella 5: Asset Value Assessment

Asset	Spoofing	Tamper-ing	Repudia-tion	Informa-tion Dis-clusure	DoS	Elevation of Privilege	Danger	Unreliabi-lity	Absence of Resilience	Attack
Misure		X			X		X			Contraffazione Eccessivo numero di richieste
Giornale	X	X			X			X		Contraffazione Cancellazione Eccessivo numero di richieste
Immagine		X	X	X	X					Ripudio Sottrazione e abuso dei cookies Eccessivo numero di richieste
Ottieni misure		X			X			X		Azione non autorizzata Eccessivo numero di richieste

Tabella 6: Minacce associate agli asset identificati, secondo il modello *Stride*

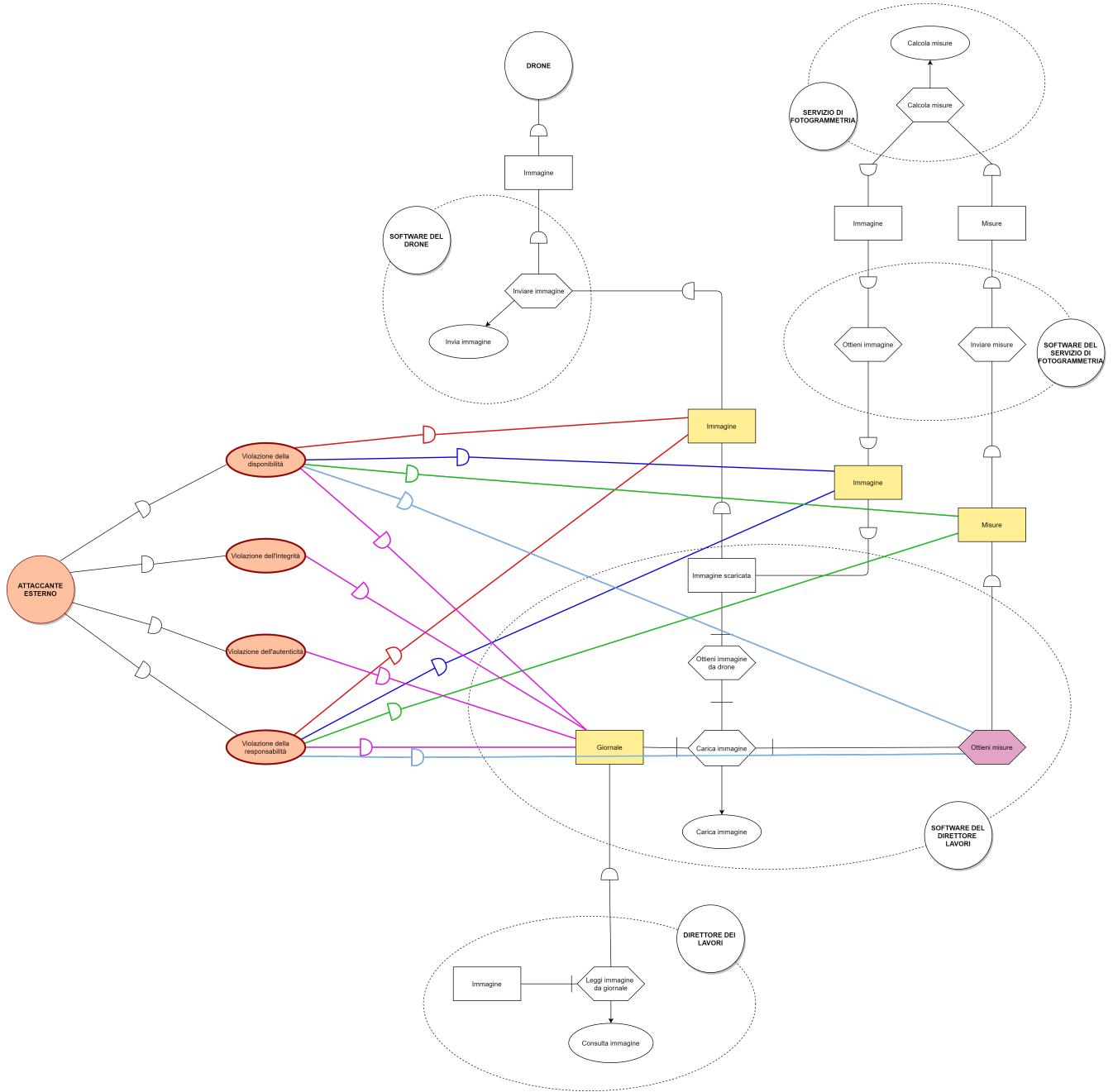


Figura 5: Abuse Case: diagramma I^* rappresentante i possibili attacchi da parte di un utente esterno

quel determinato tipo di attacco (ad esempio come si può vedere in Figura 8 b, il 'Ripudio' ha probabilità 12% che è maggiore uguale alla somma delle probabilità dei figli poichè ci possono essere degli attacchi ancora non considerati in quanto non noti) in un periodo stimato di 4 anni. Queste probabilità, per essere utilizzate in fase di calcolo del rischio del relativo attacco nella Mitigation Matrix in Tabella 11 a pagina 24, dovranno essere convertite in una scala a 5 valori secondo la matrice in Tabella 7.

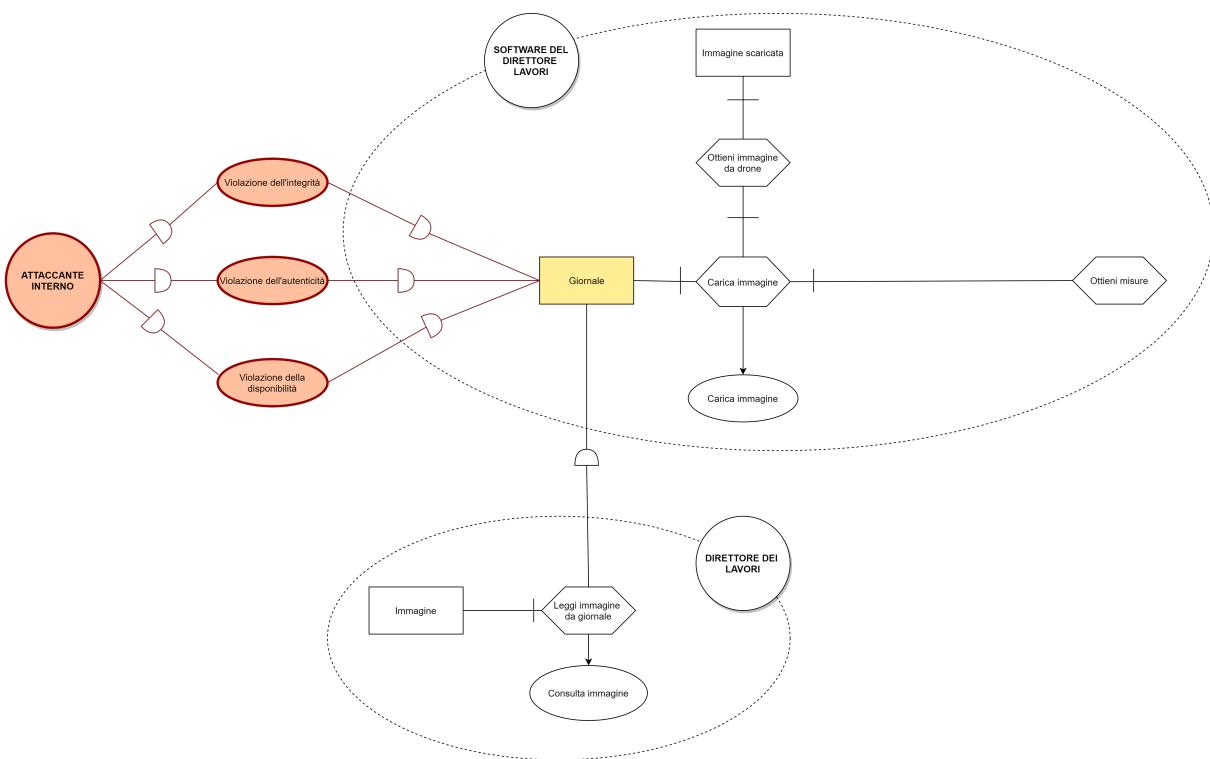


Figura 6: Abuse Case: diagramma I^* rappresentante i possibili attacchi da parte di un utente interno al software del direttore dei lavori

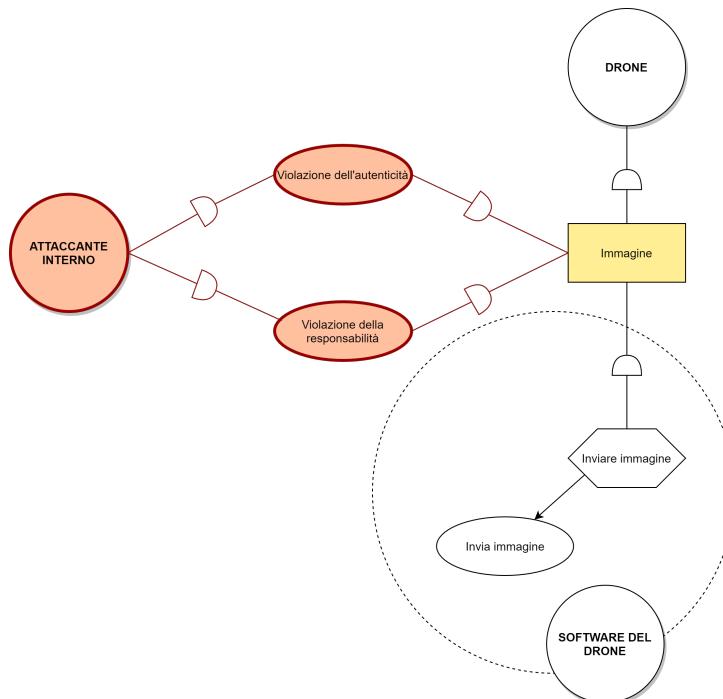
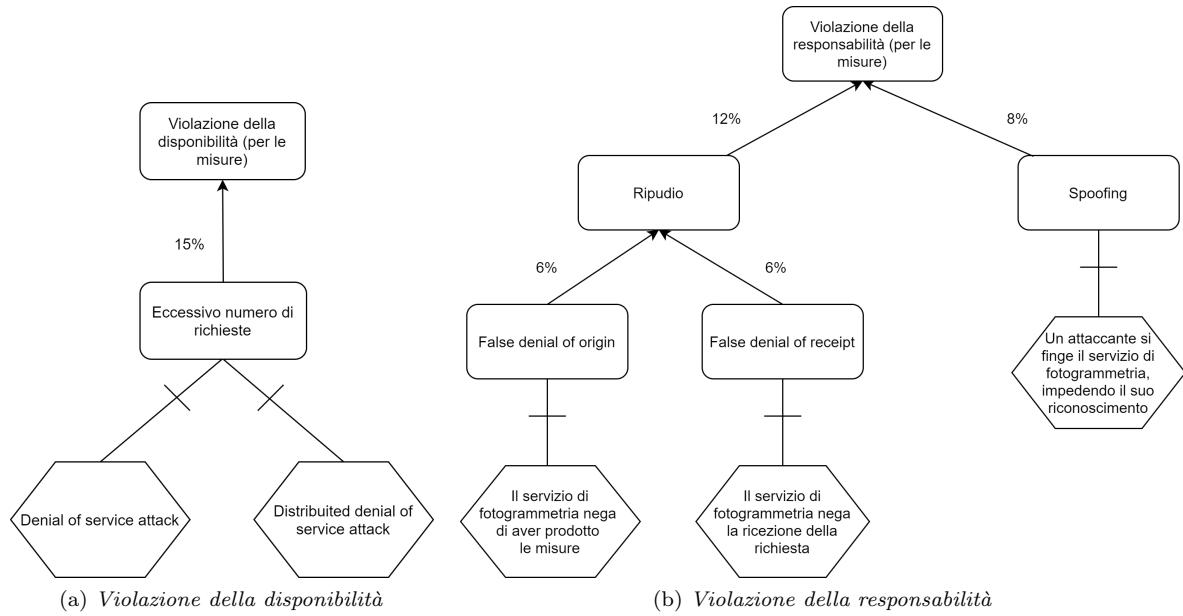


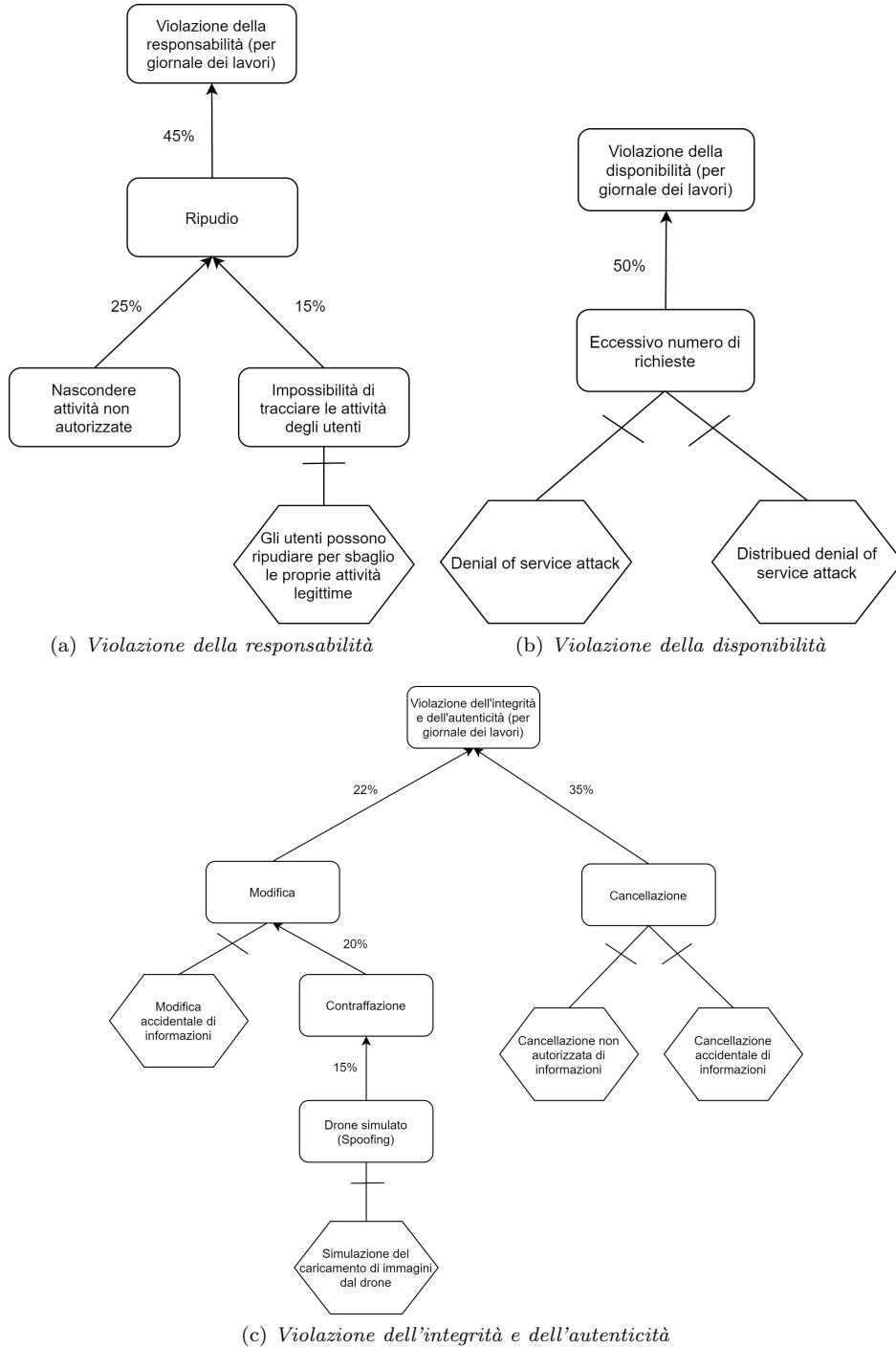
Figura 7: Abuse Case: diagramma I^* rappresentante i possibili attacchi da parte di un utente interno al software del drone

Percentuale Probabilità	Valore Probabilità
0% - 20%	1
21% - 40%	2
41% - 60%	3
61% - 80%	4
81% - 100%	5

Tabella 7: Valutazione delle probabilità di attacco

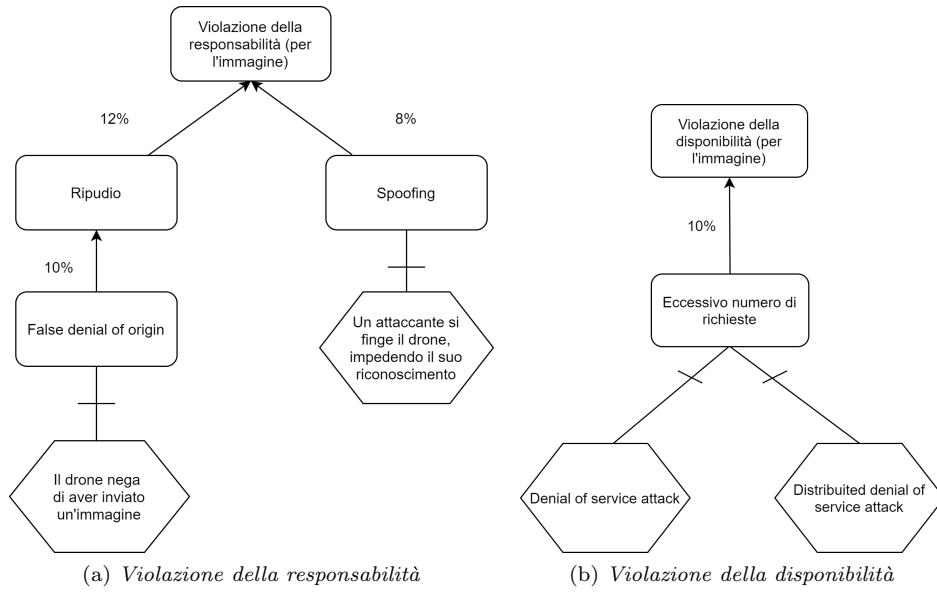
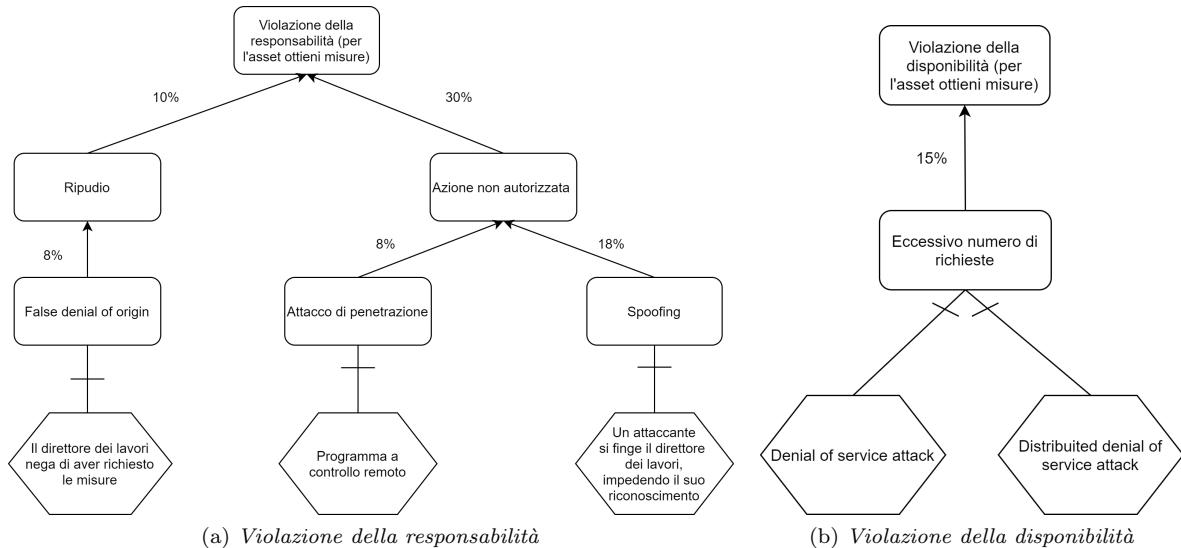
Tutto ciò è utile per poter comprendere come si possano verificare gli attacchi e, di conseguenza, identificare quali parti del sistema sono più soggette ad accesso o modifiche non autorizzate. Le Figure 8, 9, 10 e 11 rappresentano, rispettivamente, gli attack tree degli asset *misure*, *giornale*, *immagine* e *ottiene misure*.

Figura 8: Attack tree relativi all'asset *misure*

Figura 9: Attack tree relativi all'asset *giornale*

2.5 Risk Reduction

Nelle sezioni precedenti sono stati individuati gli asset e ne sono state valutate caratteristiche, vulnerabilità e potenziali attacchi. Si procede, ora, con l'analisi in dettaglio del rischio, con l'obiettivo di scegliere l'azione da mettere in pratica per garantire un certo livello di sicurezza in modo da mitigare

Figura 10: Attack tree relativi all'asset *immagine*Figura 11: Attack tree relativi all'asset *ottieni misure*

il rischio, cercando di raggiungere un buon compromesso tra sicurezza e costo. Si riprende in analisi anche il modello *Stride*, mostrata precedentemente in Tabella 6, così da prendere in esame le possibili minacce per ogni asset identificato e valutarne, per ognuno, probabilità e impatto. Per ogni possibile attacco individuato, infatti, saranno specificati probabilità di accadimento (ricavata in precedenza dagli attack tree) e il relativo impatto (tenendo in considerazione l'asset value assessment della Tabella 5) con possibili tecniche di controllo utili per limitare gli effetti delle minacce.

La valutazione del rischio avviene in maniera qualitativa, in quanto risulta troppo complesso, in questa fase, riuscire ad effettuare una valutazione quantitativa precisa e indicativa. Per effettuare questa valutazione, è utilizzata una scala di Likert a 5 valori. Nella Tabella 8 sono riportate le metriche per la

valutazione della probabilità e dell'impatto: per entrambi, sono definiti 5 possibili valori che indicano, rispettivamente, la probabilità di accadimento di una minaccia e la gravità dell'impatto in caso si verifichi realmente. Ognuno di questi valori qualitativi è associato ad un numero che ne indica il posizionamento nella scala.

Metriche per la valutazione dei rischi				
Scala per la probabilità di accadimento				
Molto Improbabile	Improbabile	Possibile	Probabile	Molto Probabile
1	2	3	4	5
Scala per l'impatto				
Lieve	Tollerabile	Accettabile	Moderato	Grave
1	2	3	4	5

Tabella 8: Metriche per la valutazione qualitativa dei rischi

Dopo aver valutato in maniera qualitativa probabilità e impatto tramite la metodologia proposta, è necessario associare ad ogni minaccia un valore unico di rischio totale che, sebbene sia ancora qualitativo, tiene in considerazione e riassume i valori identificati di probabilità e impatto. I valori di rischio sono visibili in Tabella 9: in base ai valori identificati precedentemente, tramite questa matrice è possibile associare ad ogni minaccia il suo valore di rischio. Normalmente, questo valore è calcolato eseguendo la moltiplicazione tra i valori di probabilità e impatto ma in questo caso si è associato, alle varie combinazioni di valori possibili, dei numeri per poter creare una scala di rischio su un range che si estende da 1 a 100 per ottenere maggior accuratezza.

Matrice probabilità-impatto					
Probabilità- Impatto	Lieve (1)	Tollerabile (2)	Accettabile (3)	Moderato (4)	Grave (5)
Molto Probabi- le (5)	10	30	50	80	100
Probabile (4)	8	24	40	64	80
Possibile (3)	5	15	25	40	50
Improbabile (2)	3	9	15	24	30
Molto Impro- babile (1)	1	3	5	8	10

Tabella 9: Matrice probabilità-impatto

Le celle della tabella sono colorate in quanto, ad ogni valore, è associato un valore qualitativo del rischio e una tipologia di azione prevista. Tale associazione è proposta nella Tabella 10: i valori complessivi dei rischi sono divisi in 5 range e, per ognuno di essi, è indicato la tipologia di azione che si vuole mettere in pratica.

La *Mitigation Matrix*, mostrata in Tabella 11, riassume, per ogni asset, le minacce associate, con relativa probabilità di accadimento, impatto e valore del rischio, calcolato come appena illustrato. Inoltre, la tabella presenta, per ogni possibile attacco, le tecniche di controllo proposte per limitare o arginare gli effetti delle corrispondenti minacce. Tali tecniche vengono valutate, sempre qualitativamente, indicando la loro efficacia (indicata con dei +), il loro costo (utilizzando una scala di valori 1-5) e come si intende metterla in pratica.

Azioni consigliate		
Range del rischio	Valore assoluto del rischio	Tipologia di azione prevista
Minore di 10	Molto basso	Accettazione
Da 10 a 20	Basso	Monitoraggio
Da 20 a 35	Medio	Mitigazione
Da 35 a 70	Alto	Mitigazione
Maggiore di 70	Molto alto	Evitare

Tabella 10: Azioni consigliate per ogni valore di rischio

Non tutte queste tecniche di controllo verranno effettivamente applicate: per ognuna di esse si valuta, infatti, la loro convenienza, confrontando il costo con l'efficacia e la pericolosità del rischio. In particolare, si consulta la Tabella 10, che presenta le linee guida di comportamento in base alla gravità della minaccia. Se il rischio ha valore basso, l'azione proposta è quella di accettazione o di monitoraggio, per cui non si metterà in pratica alcuna tecnica di controllo in quanto sarebbe solo un costo aggiuntivo per migliorare di poco la sicurezza di un asset per il quale l'attacco è poco probabile o reca un danno molto lieve. Se invece il rischio ha valore alto, la tabella delle azioni consigliate propone di mitigare o di evitare tale minaccia. In tal caso, viene scelta una delle tecniche di controllo proposte (nella Tabella 11 è indicata colorando la cella corrispondente in grigio) e si calcola il valore del rischio residuo che persiste dopo aver applicato tale tecnica. Se sono state presentate più tecniche per controllare lo stesso attacco, si sceglierà valutando costo ed efficacia. Per calcolare il rischio residuo sono state ricalcolati probabilità e impatto di ogni rischio considerando il contributo della tecnica di controllo scelta: per semplicità, in tabella è riportato solo in valore di rischio finale, calcolato utilizzando la Tabella 9.

Asset	Attacco	Probabilità	Impatto	Rischio	Controllo	Efficacia	Costo	Fattibilità	Rischio residuo
Misure	Violazione della responsabilità	1	4	8	Firma digitale	+++	2	Tecnicamente fattibile	8
	Eccessivo numero di richieste	1	2	3	Token di riconoscimento	++	3	Tecnicamente fattibile	3
Giornale	Contraffazione	2	5	30	Impedire le modifiche	+++	4	Tecnicamente fattibile ma richiede specifiche tecnologie di memorizzazione come le Blockchain	1
	Cancellazione	2	4	24	Impedire le cancellazioni	++	4	Tecnicamente fattibile ma richiede specifiche tecnologie di memorizzazione come le Blockchain	1
	Eccessivo numero di richieste	3	3	25	Distribuzione	++	4	Tecnicamente fattibile ma potrebbe richiedere costi di manutenzione	15
	Ripudio	3	3	25	Captcha	++	1	Tecnicamente fattibile ma potrebbe incontrare la resistenza degli utenti	1
Immagine	Sottrazione e abuso dei cookies	1	4	8	Aggiornamento dei cookies	+++	2	Tecnicamente fattibile	8
	Eccessivo numero di richieste	3	2	15	Token di accesso	++	3	Tecnicamente fattibile	15
Ottieni misure	Violazione della responsabilità	2	4	24	Offuscamento	+	3	Tecnicamente fattibile	8
	Eccessivo numero di richieste	1	2	3	Monitoraggio	+++	4	Tecnicamente fattibile	3
					Captcha	++	1	Tecnicamente fattibile ma potrebbe incontrare la resistenza degli utenti	

Tabella 11: *Mitigation Matrix*

3 Blockchain e Smart Contracts

La *blockchain* è una particolare tecnologia *Distributed Ledger*, letteralmente "registro distribuito", ovvero un sistema condiviso, replicato, sincronizzato e decentralizzato. Questa tecnologia è nata per garantire la sicurezza e il rispetto di una transazione, senza che i due utenti coinvolti si debbano per forza fidare l'uno dell'altro e senza coinvolgere una terza figura. Il concetto su cui si fonda la tecnologia *Distributed Ledger* è quello che è più sicuro interpellare un numero elevato di testimoni in quanto è più improbabile che si accordino per inserire delle clausole fasulle che gravino su una controparte e ne favoriscano l'altra. Ogni testimone della rete *Distributed Ledger* conserva una copia dello stesso registro: quando la rete dei testimoni approva una nuova transazione, questa viene registrata su tutte le copie del registro. Le tecnologie *Distributed Ledger* si distinguono tra loro per tre elementi, ovvero la struttura dati, il protocollo utilizzato per la registrazione della transazione e l'algoritmo del consenso che regola i meccanismi di accordo per l'approvazione della transazione.

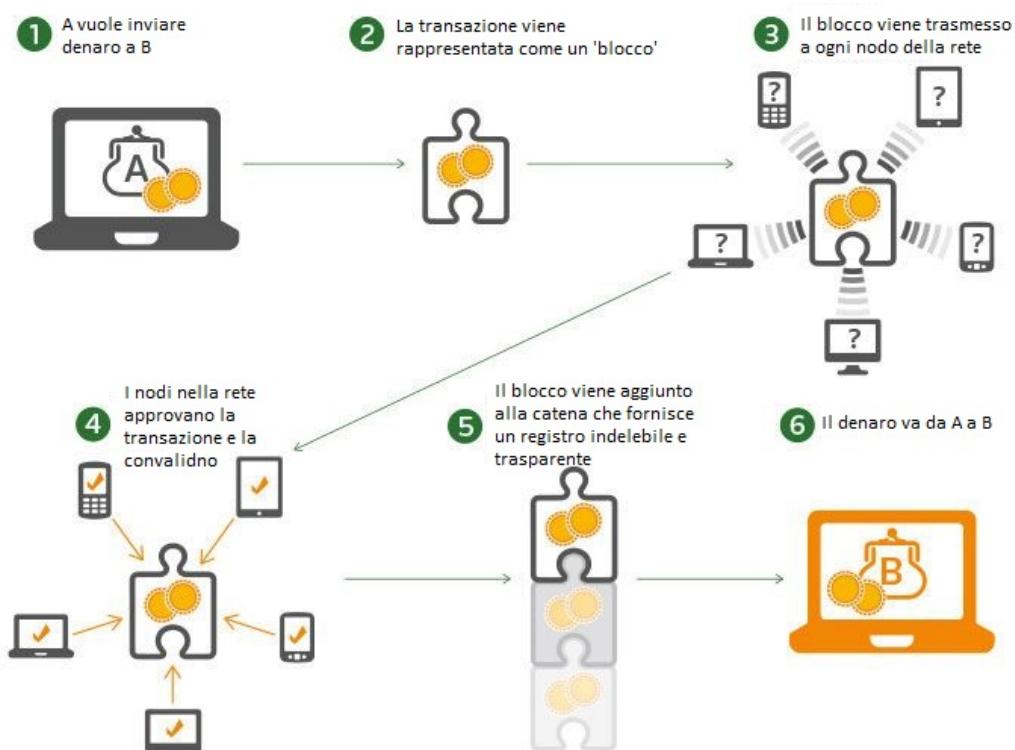


Figura 12: Workflow di una blockchain

3.1 Blockchain

In questo progetto è utilizzata, in particolare, la blockchain, nata per la registrazione di transazioni basate su criptovalute. In Figura 12 è mostrato uno schema di esempio che indica la struttura di una blockchain generica per uso finanziario. La tecnologia blockchain ha un ruolo chiave nel progetto: essa permette una trasparenza unica nel suo genere registrando l'elenco delle operazioni che vengono effettuate. Questo

permetterà di avere giornali dei lavori attendibili, in modo tale da risolvere molti dei problemi legali che si verificano allo stato attuale.

Nella sua applicazione iniziale, la blockchain era stata ideata per uno scambio di denaro decentralizzato, rivoluzionando il modo di eseguire transazioni in quanto non era più necessario il coinvolgimento di un ente terzo, come la banca, ed in quanto eliminava il problema del *double spending*. Tale tecnologia ha avuto un impatto dirompente perché non esiste più un ente governativo centrale, dal momento che la rete è diventata *peer-to-peer*: l'autorizzazione alla transazione e lo scambio di valuta avviene per acclamazione di maggioranza, utilizzando algoritmi del consenso. Successivamente, la blockchain è stata applicata anche in altri ambiti in quanto comporta notevoli vantaggi in termini di sicurezza rispetto alle soluzioni più tradizionali.

L'ascesa che ha avuto nell'ultimo decennio, ha condotto la blockchain a porsi come valida alternativa a database e sistemi di cloud storage e, attraverso l'utilizzo dei cosiddetti *Smart Contract*, ai contratti tradizionali. Ad ogni modo, nel corso dell'elaborato viene giustificata tale scelta, esprimendone i vantaggi. Il campo dell'edilizia, a cui in questo caso è rivolta, si presta bene alle sue caratteristiche considerando l'alta sensibilità degli appalti al rispetto della parte legale e alla necessità di trasparenza e immodificabilità che si ricerca.

Dal momento che le criptovalute si basano su un registro pubblico e sequenziale, la blockchain è un registro pubblico sequenziale e immutabile, basato sull'uso di tecniche crittografiche. Questo registro non è posseduto unicamente da un utente o da un'entità, ma ogni nodo conserva la sua copia, per cui sono essenziali i meccanismi di sincronizzazione affinché il registro rimanga coerente e aggiornato nonostante sia decentralizzato.

Per poter garantire la tracciabilità di tutte le operazioni svolte, è necessario che tutti gli utenti che interagiscono con la blockchain siano identificabili. Ogni utente avrà, quindi, un indirizzo pubblico tramite il quale può ricevere o originare una transazione: tale indirizzo può essere ricavato a partire da una chiave pubblica contenuta nel *wallet* dell'utente stesso. Ogni utente è, infatti, in possesso di un portafoglio, o *wallet*, che contiene la valuta digitale e la coppia di chiavi pubblica-privata. Per generare una transazione, l'utente deve dimostrare di possedere la chiave privata, che è confidenziale, corrispondente al proprio indirizzo pubblico.

La criptovaluta esiste nella forma di sequenza di transazioni annotate nella blockchain, nella seguente forma:

1. *Input*: la sorgente della valuta. Descrive le transazioni eseguite precedentemente che hanno consentito all'utente di percepire l'ammontare che intende spendere.
2. *Amount*: la valuta che l'utente vuole spendere.
3. *Output*: l'indirizzo pubblico di destinazione.

Ogni transazione crittografata viene scritta in un blocco e, successivamente, deve essere convalidata e approvata dai partecipanti della blockchain. In seguito il blocco viene inserito nella blockchain. Per individuare il Primary Node con cui il client si interfaccia sono possibili 2 strategie:

- ***proof of work*** in cui tutti i nodi competono tra loro per trovare la soluzione in una sfida crittografica ad alto costo computazionale. Il primo nodo che risolve correttamente tale sfida viene nominato Primary Node.

- ***proof of stake*** in cui tutti i nodi partecipano ad un'asta e viene nominato Primary Node il maggior offerente. Per bilanciare il sistema ed impedire la vincita da parte degli stessi nodi sono stati introdotti due meccanismi: il primo è un fattore di casualità per la scelta del vincitore, il secondo è un sistema di aging con il quale il vincitore viene sfavorito nelle aste successive.

In entrambi i casi, dopo la conclusione di ogni transazione, il Primary Node riceve una ricompensa in criptovaluta.

La blockchain è una struttura dati condivisa e immutabile, definita come un registro digitale le cui voci sono raggruppate in blocchi, concatenate in ordine cronologico, e la cui integrità è garantita dall'uso della crittografia. È un database in cui i dispositivi di archivio non sono tutti connessi attraverso un processore comune, non comunicano tutti con un solo server, ma sono connessi tramite una rete *peer-to-peer*. Questo database gestisce una lista crescente di transazioni, contenute nei blocchi, ognuno caratterizzato da una marca temporale (timestamp) e da un collegamento a un blocco precedente.

Le blockchain possono essere di tre tipologie:

- ***Pubbliche***: sono quelle per cui non vi sono restrizioni o condizioni di accesso, ovvero gli utenti non hanno bisogno di alcun tipo di autorizzazione per poter accedere alla rete, eseguire delle transazioni e partecipare alla convalida dei blocchi.
- ***Permissioned***: una blockchain è permissioned quando è soggetta ad un'autorità centrale che determina chi vi può accedere; inoltre assegna anche i ruoli agli utenti e definisce le regole di visibilità dei dati.
- ***Private***: sono reti non visibili controllate da un'organizzazione ritenuta affidabile, che determina chi vi può accedere e chi può leggere i dati. L'organizzazione può anche modificare le regole di funzionamento della blockchain stessa. Questa tipologia è quella che garantisce il più alto livello di privacy. Inoltre è la più veloce e la più economica.

3.2 Smart Contract

Gli *smart contract* sono definiti come “contractual type arrangement”, ovvero un’incorporazione di clausole contrattuali, scritte in linguaggio informatico e contenute in software o protocolli informatici. Essi possiedono la caratteristica di eseguirsi automaticamente sulla base di determinate condizioni predeterminate dalle parti.

Con questo termine intendiamo, quindi, l'esecuzione di programmi *general purpose* su blockchain e non è interessante, per lo scopo di questo progetto, essere sicuri del fatto che lo smart contract sia un contratto nel senso stretto del termine. Questo perché il concetto standard di contratto non è più rilevante in questo ambito ma si predilige l'idea di contratto come programma o algoritmo immutabile e non ripudiabile.

Uno smart contract è scritto in una transazione, salvata poi nella blockchain, per questo è tracciabile, non modificabile e con le clausole che non possono essere violate: il contratto è, infatti, un algoritmo che si esegue in automatico e per il quale non ci possono essere interpretazioni diverse e soggettive. Gli smart contract, quindi, permettono di raggiungere un livello di sicurezza più elevato rispetto ai contratti tradizionali e permettono di ridurre i costi di transazione.

Gli smart contract presentano diversi benefici rispetto ai contratti standard, ma combinato con l'uso della blockchain, comportano dei vantaggi aggiuntivi:

- Certezza dell'esecuzione: essendo un algoritmo, ogni clausola verrà soddisfatta ed eseguita se si verificano le condizioni che la attivano.
- Trasparenza: se ci sono obbligazioni contrattuali, queste e i loro risultati sono visibili a tutti i partecipanti alla rete e non solo alle parti coinvolte nel contratto.
- Immutabilità: il contratto non può essere annullato e ogni transazione registrata non può essere né modificata né cancellata.
- Possibilità di trovare sempre un accordo: anche in assenza di fiducia tra le parti interessate, è possibile trovare un accordo in quanto la fiducia ora si sposta al codice e alla rete della blockchain.

4 Implementazione

Seguendo i requisiti definiti in fase di progettazione, in questa sezione sarà esposta l'implementazione in maniera dettagliata tale da permettere di comprendere il workflow del sistema. Saranno inoltre presentate brevemente le tecnologie adottate.

Il sistema si basa su un servizio Rest sviluppato con *Express* in Node.js. Quest'ultimo ha lo scopo di interfacciarsi con tutte le altre componenti utilizzate. L'architettura complessiva può essere suddivisa a partire dalle due funzioni principali che essa supporta: l'upload di un'immagine e la lettura di un'immagine.

4.1 Upload di un'immagine

Questa funzione, schematizzata in Figura 13, permette di caricare nella blockchain le immagini acquisite dal drone con le relative misure e metadati. Il processo risulta completamente automatizzato quindi non richiede l'intervento di nessun utente.

Di seguito sono riportati i passaggi eseguiti durante il processo di upload di un'immagine sulla blockchain:

Fase 1 La prima fase viene eseguita dal drone, che dopo aver catturato un'immagine, la invia al servizio Rest realizzato con Node.js. Insieme all'immagine vengono trasmessi anche i relativi metadati e un token di riconoscimento.

Fase 2 Il servizio Rest, appena riceve una nuova immagine, ne verifica l'origine inviando il token di riconoscimento all'apposito database sviluppato con *MongoDB*.

Fase 3 Il database valida il token e restituisce il responso al servizio Rest.

Fase 4 Il servizio Rest invia l'immagine al software di fotogrammetria.

Fase 5 Il servizio di fotogrammetria calcola le misure relative all'immagine ricevuta e le restituisce al servizio Rest.

Fase 6 Il servizio Rest invia l'immagine al file system IPFS.

Fase 7 Il servizio Rest invia alla blockchain l'hash dell'immagine e le relative misure e metadati.

Come si può notare nella **Fase 6** l'immagine non viene caricata direttamente sulla blockchain ma viene salvata sul file system distribuito IPFS. Questa scelta è giustificata da ragioni prestazionali in quanto risulta inefficiente salvare nella blockchain file multimediali quali immagini.

4.2 Visualizzare immagini e misure

Questa funzione, schematizzata in Figura 14, permette al direttore dei lavori di consultare il contenuto della blockchain, in particolare le immagini acquisite dal drone con le rispettive misure e metadati. Il processo richiede all'utente una fase di autenticazione attraverso una form di login che permette di ottenere il token di riconoscimento.

Di seguito sono riportati i passaggi eseguiti durante il processo di visualizzazione della galleria immagini contenuta nella blockchain:

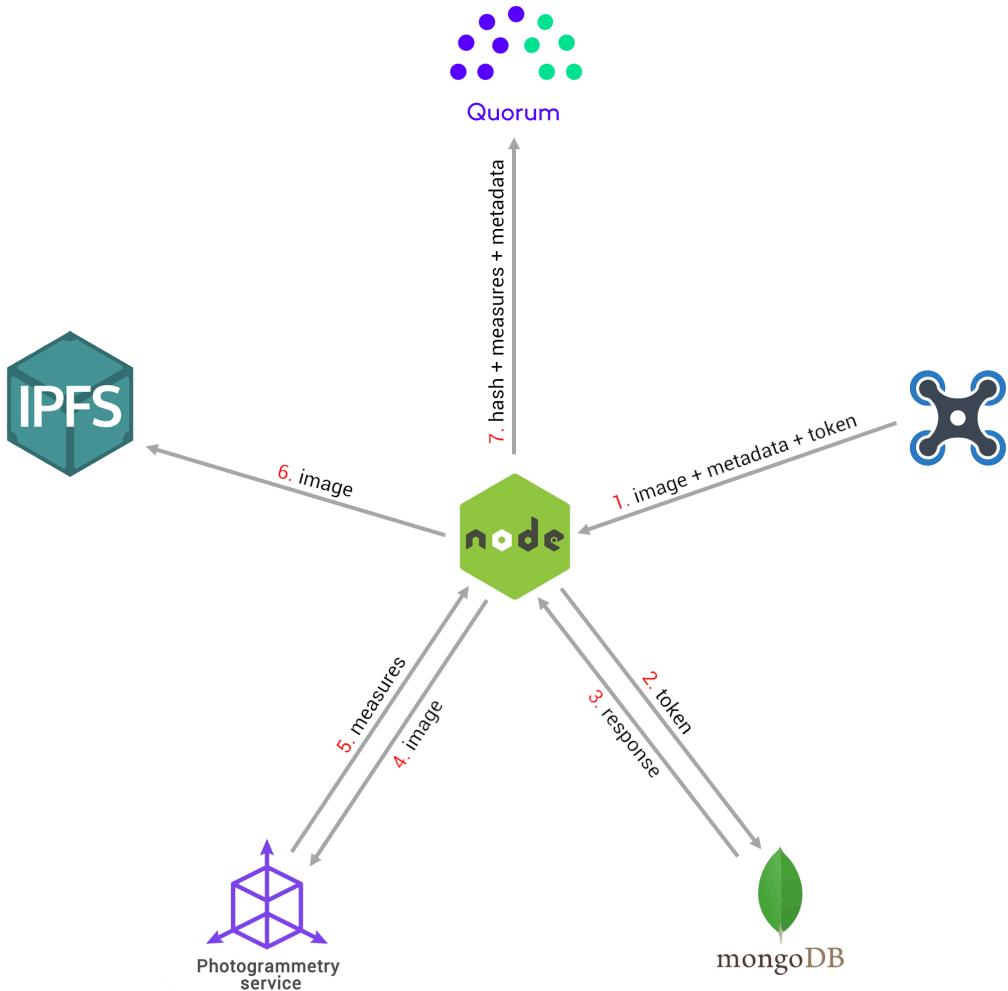


Figura 13: Workflow del upload di un'immagine

Fase 1 La prima fase viene eseguita dal direttore dei lavori, che attraverso un'interfaccia grafica, invia al servizio Rest, la richiesta per visualizzare le immagini del cantiere.

Fase 2 Il servizio Rest, appena riceve una nuova richiesta, ne verifica l'origine inviando il token di riconoscimento all'apposito database.

Fase 3 Il database valida il token e restituisce il responso al servizio Rest.

Fase 4 Il servizio Rest invia la richiesta alla blockchain.

Fase 5 La blockchain restituisce al servizio Rest i contenuti richiesti.

Fase 6 La Rest restituisce al capo dei lavori la risposta della blockchain comprendente: hash, misure e metadati.

Fase 7 Attraverso gli hash, il software del direttore dei lavori, è in grado di richiedere a IPFS le corrispondenti immagini.

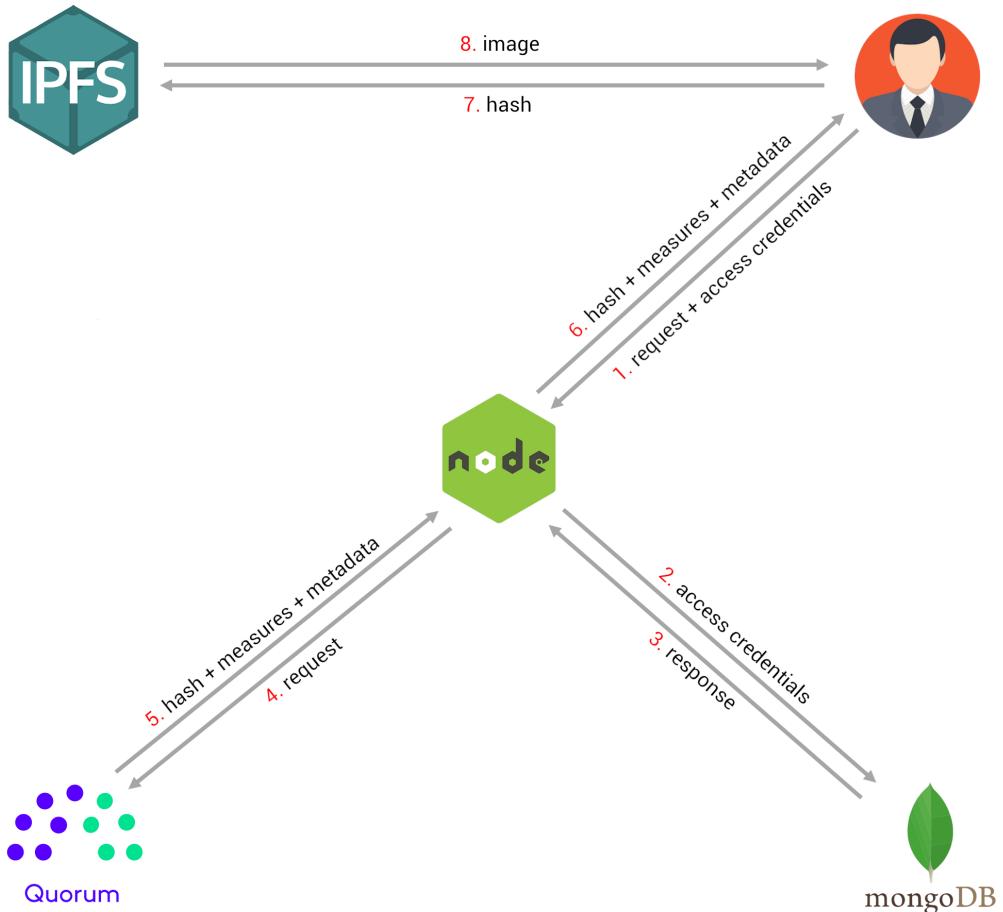


Figura 14: Workflow della visualizzazione di un'immagine

Fase 8 Il software del capo dei lavori mostra le immagini ottenute con i metadati e le misure associate.

4.3 Dettagli implementativi

Nel seguente paragrafo sono mostrate le tecnologie adottate per la realizzazione di tutti i componenti del software e le loro funzionalità offerte, una descrizione più accurata di tali tecnologie sarà offerta nella Appendice 6.3. In particolare sono stati individuati 5 componenti principali: il *drone*, il *database MongoDB*, il *servizio di fotogrammetria*, il *software del capo dei lavori* e la *Blockchain con gli Smart contract*:

- A. Per simulare l'invio di immagini da parte del *drone* è stata creata una pagina HTML facendo uso di un CSS framework chiamato Bulma¹ che permette di effettuare le operazioni richieste, ovvero l'upload delle immagini.

¹<https://bulma.io/>

- B. Per memorizzare le credenziali di accesso si è fatto uso di *MongoDB* che è un sistema di gestione di database (DBMS) open source orientato ai documenti e che supporta varie forme di dati. La libreria utilizzata per connettersi al database è *mongoose* che ha permesso di far comunicare Node.js con il database definendo lo schema degli oggetti e facendone il mapping. Come si può vedere dalla Figura 15 il database contiene lo username e la password degli utenti, dove quest'ultima è stata criptata attraverso una funzione di hash messa a disposizione dalla libreria *bcrypt*.
- C. Per simulare il funzionamento del *servizio di fotogrammetria* è stata utilizzata una API fornita da *Imagga*. Questo servizio mette a disposizione diverse soluzioni di Image Recognition, in particolare nel progetto si è fatto uso del Tagging, ovvero riconoscere automaticamente attraverso algoritmi di AI elementi visibili nell'immagine ed esprimerne la relativa confidenza.
- D. Il *software del capo dei lavori* è stato sviluppato in *Flutter*, un UI toolkit per la creazione di siti web. Nella schermata di login è stato inserito un captcha per limitare attacchi di tipo DoS.
- E. La *blockchain* è implementata in Quorum ed è costituita da 4 nodi. Per compilare gli Smart Contract è stato utilizzato il framework *Truffle* che permette di migrare i contratti nella blockchain di Quorum e prende in maniera autonoma il contract address e il contract ABI. Il primo è l'indirizzo del contratto all'interno della blockchain, il secondo è il modo standard per interagire con i contratti in Etherium sia tra la blockchain e l'ambiente esterno sia all'interno tra gli stessi contratti. Configurata la blockchain il contratto diventa definitivo e il resto viene eseguito tramite le funzioni del contratto (in Solidity). Per collegare la blockchain di Etherium con le funzionalità di Node.js è stata utilizzata la libreria *Web3* che permette di metterli in comunicazione.

Di seguito è riportato il contratto all'interno della blockchain.

```

pragma solidity ^0.7.0;
pragma experimental ABIEncoderV2;
// SPDX-License-Identifier: MIT

contract Image {

    struct ImageStruct{
        uint[] confidences;
        string[] tags;
        //image json information
        string imageInfo;
        //gps json information
        string gpsInfo;
    }

    string[] public imagesAddress;
    mapping (string => ImageStruct) blocks;
    //ImageStruct[] public blocks;

    function set(

```

```

        string memory _imagesAddress,
        uint[] memory _confidences,
        string[] memory _tags,
        string memory _imageInfo,
        string memory _gpsInfo
    ) public {
        if(!_contains(_imagesAddress)){
            imagesAddress.push(_imagesAddress);
            blocks[_imagesAddress] = ImageStruct({
                confidences: _confidences,
                tags: _tags,
                imageInfo: _imageInfo,
                gpsInfo: _gpsInfo
            });
        }
    }

    function getAllAddress() public view returns (string[] memory){
        return imagesAddress;
    }

    function getByAddress(string memory _address) public view returns (
        uint[] memory,
        string[] memory,
        string memory,
        string memory
    ){
        return (
            blocks[_address].confidences,
            blocks[_address].tags,
            blocks[_address].imageInfo,
            blocks[_address].gpsInfo
        );
    }

    function length() public view returns (uint){
        return imagesAddress.length;
    }

    function contains(string memory _address) public view returns (bool){
        for(uint i = 0; i < length(); i++){
            if(compareStringsbyBytes(imagesAddress[i], _address)) return true;
        }
        return false;
    }
}

```

```
function compareStringsbyBytes(string memory s1, string memory s2) public
    pure returns(bool){
    return keccak256(abi.encodePacked(s1)) == keccak256(abi.encodePacked(s2));
}
```

```
_id: ObjectId("5f2d72d755f52814543de5fd")
username: "drone"
password: "$2b$10$s0zfcxkVkvDuNpw/H9nQuO8/fz2W3PWyK2wMRWVuOV4XBmMCGxdgu"
__v: 0
```

```
_id: ObjectId("5f461e515e3d895abc89fea8")
username: "Admin05"
password: "$2b$10$ox.vNUxLgbjrIA6qCUm00OpHFVx19w34JKnrDsJCC7atYsVWsbdKYq"
__v: 0
```

Figura 15: Schema dei dati all'interno del database

5 Guida all'Utilizzo

In questo paragrafo vengono fornite le istruzioni necessarie all'utente per poter accedere al giornale dei lavori, caricando le immagini da salvare sulla blockchain con le relative misure dal servizio di fotogrammetria o visualizzando le immagini e misure dalla blockchain.

5.1 Eseguire l'autenticazione

Per potersi autenticare e effettuare il login è necessario risolvere un captcha ruotando l'immagine in modo da portarla al giusto orientamento (tenere premuto con il cursore sull'immagine captcha mentre la si ruota fino alla giusta posizione). Per impedire l'accesso da parte di bot, dopo 5 tentativi errati si perde la possibilità di effettuare il login la quale schermata è mostrata in Figura 16. Inoltre, dopo aver effettuato il login, è possibile fare il logout per terminare e chiudere la sessione.

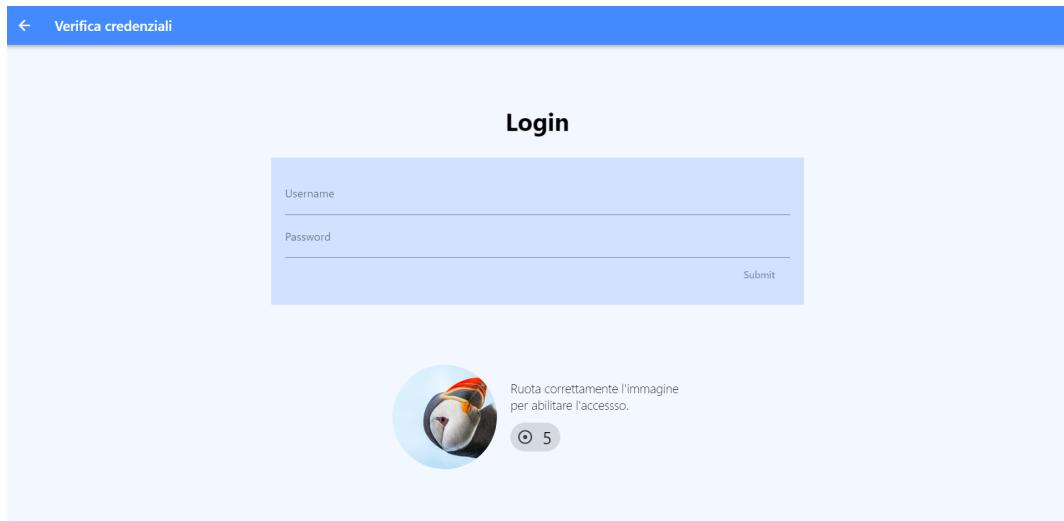


Figura 16: Finestra di login del direttore dei lavori

5.2 Caricare l'immagine nella blockchain

Per caricare un'immagine sulla blockchain basta selezionarne una dal disco locale e caricarla (questo procedimento simula l'invio dell'immagine da parte del drone che è protetto da un token di riconoscimento). È supportato solo il formato *.jpg*. Questa interfaccia è mostrata nella Figura 17.

5.3 Visualizzare la galleria e la singola immagine nella blockchain

Dopo essersi autenticati verrà visualizzata la galleria delle immagini presenti sulla blockchain come mostrato dalla Figura 18. In seguito, per visualizzare i dettagli di un'immagine, è sufficiente selezionare il bottone corrispondente e verranno mostrati i rispettivi metadati e le misure (TAGS) ottenute grazie al servizio di fotogrammetria. La Figura 19 mostra le informazioni associate a un'immagine nella blockchain.

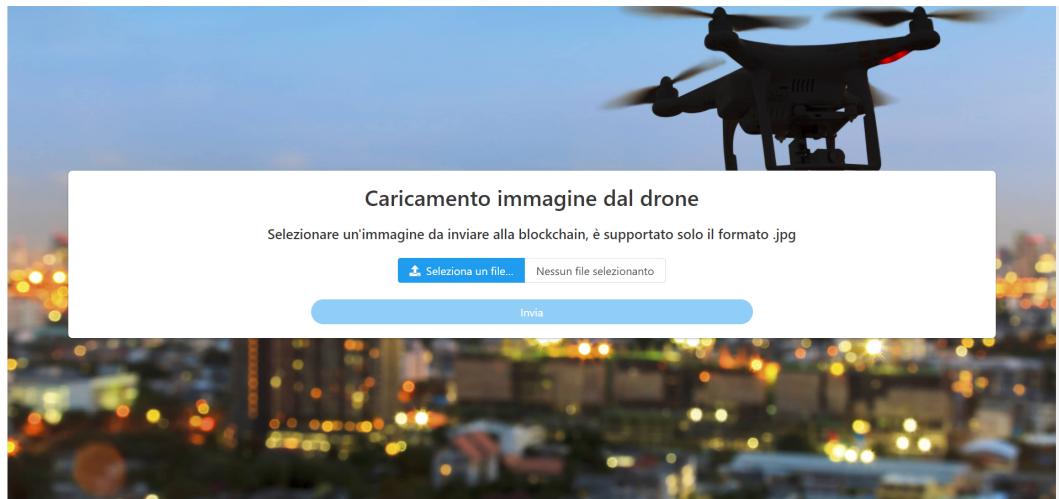


Figura 17: Interfaccia del drone simulato

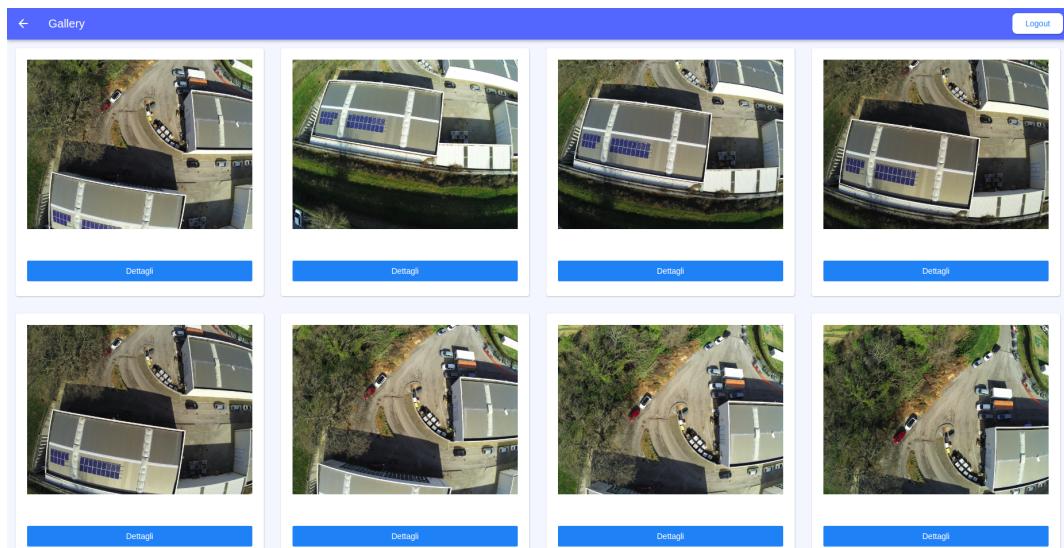


Figura 18: Galleria delle immagini nella blockchain

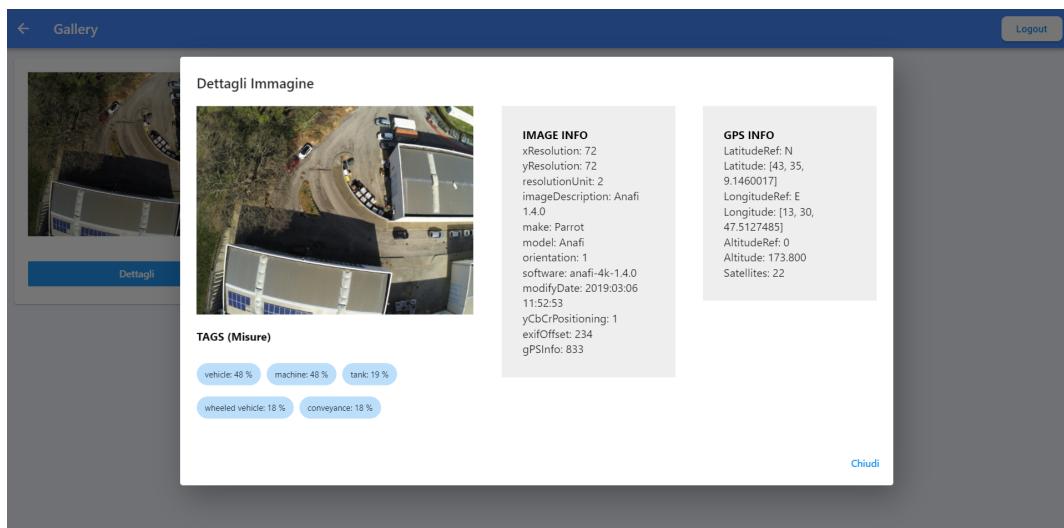


Figura 19: Dettagli e misure di una immagine all'interno della blockchain

6 Appendici

6.1 Linguaggio di modellazione I^*

I^* (i-star) è un linguaggio di modellazione dei requisiti più ricco rispetto al popolare UML, dato che quest'ultimo non è in grado di esprimere al meglio tutte le relazioni tra le componenti del progetto da realizzare, in quanto non possiede tutti i costrutti necessari.

Gli elementi che caratterizzano il linguaggio I^* sono i seguenti:

- **Attore:** viene indicato con un tondo ed è usato per rappresentare un'entità che compie un'azione per raggiungere un obiettivo. Oltre l'attore è possibile esprimere l'agente (individuo o sistema specifico) e il ruolo (figura ricoperta nel contesto di modellazione).
- **Softgoal:** è indicato con un simbolo a forma di nuvola e rappresenta un obiettivo che non ha una definizione specifica ed esaustiva e per cui non vi è un criterio preciso che permetta di stabilire se effettivamente sia stato raggiunto o meno.
- **Hardgoal:** viene rappresentato tramite un ovale e corrisponde ad un obiettivo, questa volta più preciso, per cui si è in grado di stabilire dei criteri di raggiungimento.
- **Task:** simboleggiato con un esagono, indica una particolare attività eseguibile da un determinato attore per raggiungere uno specifico obiettivo o risorsa.
- **Risorse:** vengono indicate con dei rettangoli e fanno riferimento ad elementi prodotti dalla realizzazione di un task o risorse necessarie al fine di poter eseguire un task.

Dopo aver rappresentato i principali componenti del sistema tramite i costrutti appena introdotti, è necessario esprimere le dipendenze tra questi componenti. Le dipendenze vengono rappresentate tramite degli archi contrassegnati con una D che evidenziano, in base al loro orientamento, il flusso di dipendenza (da non confondersi con il flusso delle informazioni che potrebbe anche essere inverso). Le dipendenze possono legare tra di loro soltanto gli attori attraverso 4 tipologie di legame, ovvero *goal dependency*, *task dependency*, *resource dependency* e *softgoal dependency*: la differenza tra questi legami sta nel fatto che le dipendenze tra i 2 attori possono essere caratterizzate da elementi diversi. La forza e l'utilità del linguaggio I^* risiedono nel fatto che permette di eseguire delle decomposizioni che permettono di rendere il diagramma estremamente dettagliato, chiaro e completo, in quanto è possibile mettere in evidenza tutte le relazioni di dipendenza esistenti. Le principali forme di decomposizione sono:

- **Goal decomposition:** l'obiettivo viene decomposto in tanti sotto-obiettivi, che possono essere legati da una relazione di AND (AND DECOMPOSITION) in cui devono essere soddisfatti tutti i sotto-obiettivi per concorrere al raggiungimento dell'obiettivo sovrastante, oppure da una relazione di OR (OR DECOMPOSITION) in cui è sufficiente raggiungere almeno uno dei sotto-obiettivi.
- **Means-end decomposition:** si verifica quando una determinata attività è un mezzo per raggiungere un determinato fine. Le attività sono da intendersi congiuntive, mentre se si vuole esprimere un legame disgiuntivo è necessario utilizzare prima il legame di goal decomposition visto precedentemente e, solo successivamente, si può applicare questa tipologia di decomposizione.

- **Softgoal decomposition:** si esprime il grado con cui vari elementi (obiettivi, risorse o altre attività) aiutano nel perseguimento di un softgoal. Ad esempio, il livello "make" indica che quell'elemento contribuisce in maniera consistente al raggiungimento del goal.
- **Task decomposition:** indica tutto ciò che è propedeutico alla realizzazione di quella particolare attività. Ad esempio, per poter compiere un task potrebbe essere necessario possedere una determinata risorsa o aver raggiunto uno specifico obiettivo.

*I** permette inoltre di andare a definire qual è l'ambito di competenza di un determinato attore, andando a identificare nello strategic rational model il boundary associato a quell'attore specifico. Questo particolare modello, in cui generalmente si trova una decomposizione raffinata di tutte le competenze dell'attore, unito ai flussi di dipendenze dei vari attori del sistema, permette di comprendere tutte quelle che sono le dinamiche di interazione all'interno del sistema. Tutta la sezione progettuale relativa alla raccolta, analisi e modellazione dei requisiti è stata, infatti, eseguita tramite il supporto di questo linguaggio di modellazione.

6.2 Triadi per le security policy

Le security policy associate agli asset identificati possono essere estrapolati dalle tiadi conosciute. Le triadi più importanti e conosciute sono tre:

1. CIA TRIADE:

- *Confidenzialità:* una certa informazione non può essere accessibile ad utenti non autorizzati
- *Integrità:* una certa informazione non può essere modificata da utenti non autorizzati
- *Disponibilità:* una certa informazione deve essere sempre disponibile quando richiesta da utenti autorizzati.

2. AAA TRIADE:

- *Autenticità:* una certa informazione è autentica e vera
- *Garanzia:* gli utenti si comportano come desiderato
- *Responsabilità:* è sempre possibile attribuire la responsabilità di una determinata azione all'interno del sistema.

3. TRIADE SAFETY/RELIABILITY/RESILIENCE:

- *Safety:* il sistema non reca danno a cose e persone
- *Affidabilità:* il sistema eroga un servizio come gli utenti si aspettano
- *Resilienza:* garantire una continuità di servizio anche in seguito ad errori e guasti.

6.3 Tecnologie utilizzate

In questo paragrafo verranno descritte brevemente per una maggiore comprensione tutte le tecnologie che sono state utilizzate e mostrate in questo progetto per la realizzazione del software.

6.3.1 Node.js

Node.js (<https://nodejs.org/it/>) è un framework JavaScript che permette di utilizzare V8, l'interprete JavaScript di Google. Questo consente agli sviluppatori di realizzare web application con JavaScript non più solo lato client, ma anche sfruttandolo come linguaggio di programmazione lato server. La caratteristica principale di Node.js risiede nella possibilità che offre di accedere alle risorse del sistema operativo in modalità event-driven e non sfruttando il classico modello basato su processi o thread concorrenti, utilizzato dai classici web server. Il modello event-driven, o “programmazione ad eventi”, si basa su un concetto piuttosto semplice: si lancia una azione quando accade qualcosa. Ogni azione quindi risulta asincrona a differenza dei pattern di programmazione più comune in cui una azione succede ad un'altra solo dopo che essa è stata completata. Ciò dovrebbe garantire maggiore efficienza delle applicazioni grazie ad un sistema di callback gestito a basso livello dal runtime.

6.3.2 IPFS

L'InterPlanetary File System, o IPFS (<https://ipfs.io/>) è un protocollo e una rete peer-to-peer per la memorizzazione e la condivisione dei dati in un file system distribuito. L'IPFS utilizza l'indirizzamento dei contenuti per identificare in modo univoco ogni file in uno spazio dei nomi globale che collega tutti i dispositivi di calcolo. IPFS permette agli utenti non solo di ricevere ma anche di ospitare contenuti. Il file system è costruito attorno ad un sistema decentralizzato di utenti-operatori che detengono una parte dei dati complessivi, creando un sistema resiliente di archiviazione e condivisione dei file. Ogni utente nella rete può servire un file in base al suo indirizzo di contenuto, e altri peer nella rete possono trovare e richiedere quel contenuto da qualsiasi nodo che lo abbia usando una tabella hash distribuita (DHT). I vantaggi evidenti che derivano dal modello di archiviazione distribuita di IPFS si applicano all'archiviazione dei dati che risulta così molto più efficiente e alla permanenza immutabile:

- Nessuna dipendenza dai server
- Riduzione di costo
- Storicità dei dati memorizzati sulla rete

6.3.3 MongoDB

MongoDB(<https://www.mongodb.com/it>) è un DBMS non relazionale, orientato ai documenti. Classificato come un database di tipo NoSQL, MongoDB si allontana dalla struttura tradizionale basata su tabelle dei database relazionali in favore di documenti in stile JSON con schema dinamico, rendendo l'integrazione di dati di alcuni tipi di applicazioni più facile e veloce. MongoDB è concepito in origine come database distribuito; l'alta disponibilità, la scalabilità orizzontale e la distribuzione geografica sono quindi native e facili da usare. Il modello del documento di MongoDB è facile da comprendere e utilizzare per gli sviluppatori, e offre allo stesso tempo tutte le funzionalità richieste per soddisfare i requisiti più complessi a qualsiasi scala.

6.3.4 Flutter

Flutter (<https://flutter.dev/web>) è un progetto open source totalmente gratuito per la creazione di app native di alta qualità su iOS e Android in tempi rapidi e con il supporto alle interfacce native. Flutter supporta un ambiente di sviluppo unificato per ambienti web e mobile, dove il codice è composto da una singola codebase. Le app e le interfacce native Flutter utilizzano la GPU e possono accedere ai servizi API. È ideato per lo sviluppo in velocità e lo stateful hot reload permette di modificare il codice e crearlo rapidamente. Include inoltre una vasta gamma di widget personalizzabili. Gran parte del suo sistema è scritto in Dart, un nuovo e moderno linguaggio orientato agli oggetti che definisce gestione, animazioni, framework e widget ed offre agli sviluppatori un grande controllo sul sistema stesso. L'intento di Dart è quello di risolvere i problemi di JavaScript offrendo al tempo stesso migliori prestazioni, la possibilità di sviluppare più facilmente strumenti utili alla gestione di progetti di grandi dimensioni e migliori funzionalità legate alla sicurezza.

6.3.5 Quorum

Quorum (<https://www.goquorum.com/>) è una piattaforma aziendale basata su blockchain. Si tratta di un fork del client pubblico di Ethereum con diversi miglioramenti a livello di protocollo per supportare le esigenze di business. Lo scopo principale del progetto Quorum è quello di sviluppare un client Ethereum aziendale che permetta alle aziende di beneficiare della tecnologia blockchain. Poiché Quorum è un progetto open-source, la base di codice della piattaforma è aperta, il che promuove la fiducia nella piattaforma. Quorum garantisce la riservatezza delle transazioni che è una caratteristica richiesta in molti settori come i servizi finanziari, la sanità, la legge e il governo. Assicura inoltre che la velocità e la scalabilità della rete siano adeguate per gestire i casi di utilizzo aziendale. Un'altra caratteristica fondamentale è che la blockchain è accessibile solo alle entità autorizzate. Rispetto alla tecnologia Ethereum, Quorum apporta i seguenti miglioramenti:

- Gestione dei permessi di rete e peer-to-peer
- Miglioramento della privacy delle transazioni e dei contratti
- Meccanismi di consenso basati sul voto
- Migliori prestazioni

6.3.6 Solidity

Solidity (<https://solidity.readthedocs.io/en>) è un linguaggio di programmazione orientato agli oggetti per la scrittura di Smart Contract. Viene utilizzato per l'implementazione di Smart Contract su varie piattaforme blockchain, in particolare Ethereum. Solidity è un linguaggio di programmazione progettato per sviluppare Smart Contract che girano sull'EVM. Con Solidity, gli sviluppatori sono in grado di scrivere applicazioni che implementano la logica di business auto-forzante incorporata nei Smart Contract, lasciando una registrazione senza ripudio e autorevole delle transazioni.

6.3.7 Truffle

Truffle(<https://www.trufflesuite.com/>) è un ambiente di sviluppo ed un framework di test che lavora sulla blockchain di Ethereum. La suite è progettata per gestire l'intero flusso di lavoro dei developer blockchain, dallo sviluppo dello smart contact fino all'applicazione front-end.