

UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA

SISTEMI COMPLESSI: MODELLI E SIMULAZIONE



Sviluppo di un Simulatore per Smart Grid basato su Sistemi Multiagente

Autori:

Alessandro Isceri - 879309 - a.isceri@campus.unimib.it

Mattia Ingrassia - 879204 - m.ingrassia3@campus.unimib.it

Settembre 2025

Indice

1	Introduzione	1
1.1	Considerazioni sull'Utilizzo dell'Energia Elettrica	1
1.2	Obiettivi del Progetto	1
2	Smart Grid Multiagente	2
2.1	Stato dell'Arte	2
2.2	MAS: Multi-Agent System	2
2.3	JADE - Java Agent DEvelopment Framework	2
3	Realismo del Simulatore	5
3.1	Simulazione degli Eventi Atmosferici	5
3.1.1	Meteo	5
3.1.2	Nuvolosità	6
3.1.3	Vento	6
3.2	Coordinate e Calcolo della Distanza	7
3.3	Trasporto di Energia nella Rete	8
3.3.1	Perdita di Energia durante la Trasmissione	8
3.3.2	Batterie	9
3.4	Impianti Fotovoltaici	10
3.5	Impianti Eolici	13
3.6	Impianti Idroelettrici	14
3.7	Impianti Diesel	14
3.8	Conversione da Watt a Wattora	14
4	Sviluppo del Simulatore	15
4.1	Sistema a Turni	15
4.2	Configurazione del Simulatore	15
4.3	Componenti del Sistema	16
4.4	Modellazione degli Agenti	17
4.4.1	Smart Building	17
4.4.2	Renewable Power Plant	18
4.4.3	Non-Renewable Power Plant	18
4.4.4	Grid	18
4.4.5	Load Manager	19
4.5	Ambiente di Simulazione	20
4.6	Interazione tra Agenti e Protocollo di Comunicazione	20
4.6.1	Discovery iniziale	21
4.6.2	Simulation Agent	22
4.6.3	Smart Building - Grid	25
4.6.4	Grid - Power Plant	27
4.6.5	Load Manager - Grid	28
4.6.6	Load Manager - Non Renewable Power Plant	29
4.6.7	Grid - Grid	30
4.7	Classi Util	31
4.8	Configurazione dell'Ambiente	32
4.8.1	Cables	32
4.8.2	DieselPowerPlants	33

4.8.3	Grids	34
4.8.4	HydroPowerPlants	35
4.8.5	LoadManagers	36
4.8.6	Owners	37
4.8.7	PhotovoltaicPowerPlant	37
4.8.8	SmartBuildings	38
4.8.9	WindPowerPlants	40
5	Scenario di Simulazione	41
5.1	L'isola di Tioman	41
5.1.1	Kampung Tekek	42
5.1.2	Kampung Juara	43
5.1.3	Kampung Salang	44
5.1.4	Kampung Paya	44
5.1.5	Kampung Bunut	45
5.1.6	Kampung Air Batang	45
5.1.7	Kampung Mukut	46
5.2	Configurazione dello Scenario	46
5.2.1	Impianti di Produzione Elettrica	46
5.2.2	Modellazione dei Consumatori	47
5.2.3	Catene di Markov	48
5.2.4	Configurazione del Simulatore	49
5.3	Risultati Ottenuti	50
6	Conclusioni	53
6.1	Considerazioni Finali	53
6.2	Sviluppi Futuri	53
	Bibliografia	54

Elenco delle figure

1	Diagramma di sequenza generato dallo sniffer	3
2	Paradigma asincrono di gestione dei messaggi di JADE	4
3	Panoramica dei principali Behaviours utilizzabili in JADE	4
4	Rappresentazione dei Weather Codes	5
5	Scala okta per la rappresentazione della copertura delle nuvole	6
6	Rappresentazione degli angoli di Azimut e Zenit	10
7	Architettura degli agenti utilizzabili nel simulatore	17
8	Diagramma di sequenza che riporta uno scambio di messaggi generico che avviene durante un turno	24
9	Topografia dell'isola di Tioman	42
10	Dettaglio su Kampung Tekek	43
11	Dettaglio su Kampung Juara	43
12	Dettaglio su Kampung Salang	44
13	Dettaglio su Kampung Paya	44
14	Dettaglio su Kampung Bunut	45
15	Dettaglio su Kampung Air Batang	45

16	Dettaglio su Kampung Mukut	46
17	Riferimento al carico medio dei consumi nell'arco di una giornata	48
18	Carico medio dei consumi nell'arco della giornata simulata	48
19	Markov Chain per la simulazione del meteo e relativa copertura nuvolare	49
20	Markov Chain per la simulazione dell'andamento della velocità del vento	49
21	Confronto della produzione di energia tra lo studio di riferimento e il simulatore	50
22	Variazione del meteo nell'arco della giornata simulata	51
23	Variazione della velocità del vento nell'arco della giornata simulata	51

Elenco delle tabelle

1	Classificazione della velocità del vento secondo la scala di Beaufort	7
2	Valori di albedo per diverse superfici	13
3	Struttura di un messaggio di coordinate discovery	21
4	Struttura di un messaggio di non-renewable discovery	21
5	Struttura di un messaggio di cable discovery	22
6	Struttura di un messaggio di inizio simulazione	22
7	Struttura di un messaggio di interruzione della simulazione	23
8	Struttura di un messaggio di ripresa della simulazione	23
9	Struttura di un messaggio di inizio turno	23
10	Struttura di un messaggio di fine turno	24
11	Struttura di un messaggio di richiesta di energia da parte dello Smart Building .	25
12	Struttura di un messaggio di risposta da parte della Grid	25
13	Struttura di un messaggio di rilascio di energia da parte dello Smart Building . .	26
14	Struttura di un messaggio di rilascio di energia da parte di uno Smart Building in Blackout	26
15	Struttura di un messaggio di Blackout da parte di uno Smart Building	27
16	Struttura di un messaggio di ripristino di uno Smart Building	27
17	Struttura di un messaggio di ripristino non possibile	27
18	Struttura di un messaggio di produzione dei Power Plant	28
19	Struttura di un messaggio da Grid a Load Manager	28
20	Struttura di un messaggio contenente le informazioni sulla distribuzione	29
21	Struttura di un messaggio contenente le richieste di energia alle Non-Renewable Power Plant	30
22	Struttura di un messaggio di instradamento tra Grid	30
23	Panoramica dei dati salvati durante la simulazione	31
24	Elementi della rete elettrica	41
25	Caratteristiche dell'impianto diesel modellato	46
26	Caratteristiche della centrale idroelettrica modellata	46
27	Caratteristiche dell'impianto fotovoltaico modellato	47
28	Caratteristiche della centrale eolica modellata	47
29	Caratteristiche della centrale eolica modellata	47
30	Parametri con cui è stata eseguita la simulazione sullo scenario di Tioman . . .	50

1 Introduzione

1.1 Considerazioni sull'Utilizzo dell'Energia Elettrica

É evidente come le esigenze energetiche di ogni individuo siano cambiate col passare del tempo. A pari passo con la progressione tecnologica, e con una digitalizzazione ormai presente in quasi ogni campo, il fabbisogno elettrico necessario per soddisfare ogni richiesta aumenta di anno in anno.

Per moltissimo tempo la produzione di energia elettrica è stata legata all'utilizzo di combustibili fossili, mentre in tempi più recenti si è deciso di spingere per una diminuzione del loro utilizzo, sia per motivi di inquinamento, sia perché si tratta di una risorsa non rinnovabile, destinata prima o poi a esaurirsi.

Con questo cambio di posizione, la presenza di fonti rinnovabili su tutto il territorio globale è aumentata, rendendo impianti fotovoltaici, idroelettrici ed eolici dei punti chiave per la soddisfazione del fabbisogno energetico in tutto il mondo.

Sebbene questi impianti abbiano un impatto minimo sull'ambiente, essi sono fortemente dipendenti dai fattori atmosferici e dalla zona in cui sono installati.

Per questo motivo, è fondamentale riuscire ad orchestrare in maniera intelligente la coesistenza di impianti rinnovabili e non rinnovabili, usando questi ultimi solo nei casi in cui le fonti rinnovabili non bastassero in determinati momenti per soddisfare la richiesta energetica.

É importante inoltre limitare la quantità di energia elettrica prodotta, in modo che sia sufficiente per adempiere alle richieste della rete senza però incorrere in sprechi derivanti da una eccessiva produzione.

Una gestione ottimale che minimizzi gli sprechi sarà fondamentale per gestire il crescere della domanda di energia elettrica globale e fare in modo che tutti possano avere accesso senza problemi all'elettricità.

1.2 Obiettivi del Progetto

L'obiettivo del progetto è creare un simulatore pseudo-realistico di una rete elettrica, utilizzando un'architettura multiagente, che offra diverse funzionalità, tra cui:

- Monitorare i consumi e gli sprechi di una rete esistente.
- Monitorare l'andamento della produzione energetica di una rete.
- Testare eventuali modifiche nella struttura di una rete esistente (cambiare cavi, inserire/rimuovere degli elementi, ecc...).
- Modellare con precisione ogni elemento della rete, sia per quanto riguarda gli edifici che per quanto riguarda le griglie di distribuzione e gli impianti energetici.
- Testare nuovi algoritmi di distribuzione di energia elettrica.
- Simulare la creazione di una nuova rete da zero.

2 Smart Grid Multiagente

2.1 Stato dell'Arte

L'aumentare delle dimensioni delle reti di distribuzione elettrica ha portato a un cambio di necessità rispetto agli approcci tradizionali applicati fino a oggi. Studi recenti di esperti del settore hanno portato alla valutazione di un cambiamento nella gestione dei generatori elettrici, promuovendo un approccio distribuito dove diverse entità collaborano tra di loro per produrre e distribuire energia piuttosto che avere un grande generatore centralizzato che produca e distribuisca in autonomia l'energia necessaria per soddisfare le richieste.

Infatti, si stima che in futuro le griglie di distribuzione seguiranno sempre di più un approccio distribuito, dato che questo metodo presenta diversi punti di forza; ad esempio, esso è caratterizzato da una migliore resistenza ai guasti, siccome la produzione energetica non dipende da un singolo nodo ma ognuno può contribuire alla produzione e compensare le mancanze di altri nodi. Inoltre, anche gli edifici smart, che sono sempre più frequenti, possono partecipare attivamente alla produzione di energia elettrica se dotati di un impianto fotovoltaico, nel caso avessero dei surplus di energia. Le smart homes possono inoltre prevedere il proprio consumo energetico nel prossimo futuro ed effettuare una richiesta di energia abbastanza precisa in modo tale da evitare sprechi [1].

Il dominio di riferimento si presta bene ad una modellazione ad agenti dove diverse entità autonome comunicano e collaborano per raggiungere un obiettivo comune, come descritto in [2].

2.2 MAS: Multi-Agent System

I sistemi presi in considerazione, chiamati sistemi MAS (Multi-Agent System), sono formati da diverse entità, chiamate agenti, che lavorano in autonomia ma possono collaborare per raggiungere i propri obiettivi o dei traguardi comuni.

Tendenzialmente i sistemi MAS sono composti da pochi agenti dotati di un buon livello di autonomia, piuttosto che da molti agenti con un comportamento semplice. Questo tipo di sistema viene solitamente utilizzato per affrontare problemi che sono fisicamente o logicamente distribuiti, e quindi si adatta molto bene al dominio applicativo preso in considerazione.

La collaborazione tra gli agenti può avvenire in diverse forme, tra le più comuni troviamo l'orchestrazione, dove un agente viene prefissato come direttore e si occupa di dirigere la comunicazione, inviando indicazioni agli agenti interessati su come deve avvenire la conversazione. Un'altra tecnica largamente utilizzata è la coordinazione in cui non esiste un direttore che guida la comunicazione, ma gli agenti sono implicitamente organizzati.

Tendenzialmente la trasmissione di informazioni tra gli agenti avviene tramite lo scambio di messaggi, anche se esistono altre modalità.

Esistono diversi framework che permettono la modellazione di sistemi MAS, tra cui ZEUS, AgentBuilder, JADE e MADkit, ognuno con i propri pregi e difetti, i quali vengono presentati in [1]. Per la realizzazione del simulatore è stato utilizzato JADE.

2.3 JADE - Java Agent DEvelopment Framework

JADE è un framework sviluppato in Java da Telecom Italia SpA [3], che permette di creare ed eseguire sistemi multiagente, offrendo diversi strumenti utili soprattutto in fase di testing; questi tool permettono di creare agenti "dummy" per inviare messaggi manualmente e l'inserimento di un agente sniffer, che rende possibile il monitoraggio dei messaggi scambiati tra gli agenti

del sistema e la creazione di diagrammi di sequenza a runtime. Entrambe le funzionalità sono utilizzabili tramite una semplice interfaccia grafica.

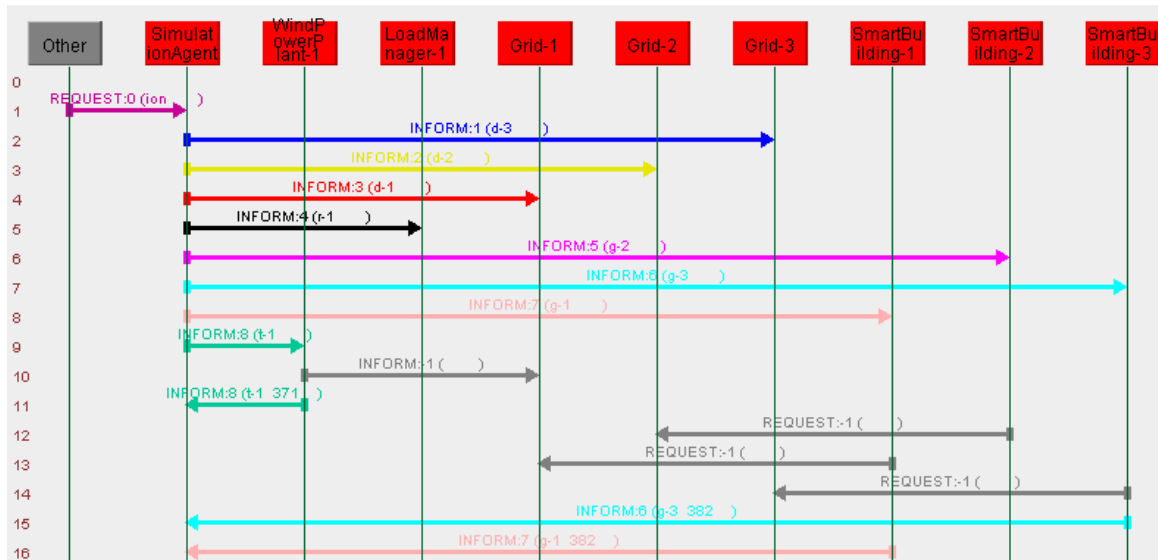


Figura 1: Diagramma di sequenza generato dallo sniffer

JADE è largamente utilizzato in letteratura e segue lo standard FIPA (Foundation of Intelligent Physical Agents) per la gestione dei messaggi [4]; questo standard indica alcune proprietà che i messaggi devono necessariamente avere.

Per ogni messaggio, ci deve essere un mittente, che ovviamente corrisponde all'agente che ha inviato il messaggio, uno o più agenti destinatari e una performative, detta anche intenzione comunicativa, che indica qual è l'intento del messaggio; il tipo di performative varia a seconda del messaggio e i valori più utilizzati sono:

- **INFORM**: lo scopo del messaggio è quello di informare un agente.
- **REQUEST**: nel messaggio è presente una richiesta.
- **AGREE**: il messaggio viene inviato con l'intento di accettare una richiesta.
- **REFUSE**: il messaggio viene inviato con l'intento di rifiutare una richiesta.

In aggiunta alla performative e al contenuto effettivo del messaggio, le comunicazioni possono contenere ulteriori informazioni, tra cui l'id della conversazione, che ritorna utile per la gestione delle conversazioni "parallele" tra due o più agenti, oltre alla possibilità di specificare delle ontologie condivise.

Molti componenti della comunicazione FIPA derivano da studi umanistici che approfondiscono l'interazione umana; ad esempio, l'utilizzo di una performative all'interno dei messaggi è un'idea che deriva dallo studio di sovrastrutture note nella comunicazione tra persone, ciò rende ogni messaggio più comprensibile.

Per permettere agli agenti di comunicare, JADE utilizza l'Asynchronous Message Passing; infatti, ogni agente possiede una coda di messaggi ricevuti che viene aggiornata ogni qualvolta l'agente riceve o legge un messaggio.

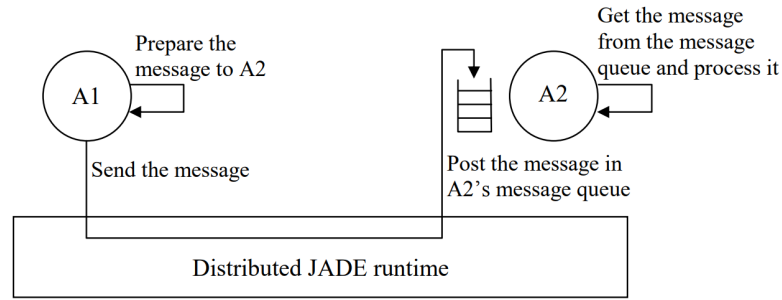


Figura 2: Paradigma asincrono di gestione dei messaggi di JADE

All'interno del framework, ogni agente sviluppato deve estendere la classe Agent di JADE e deve essere caratterizzato da dei comportamenti; tali comportamenti si possono specificare estendendo una sottoclasse di Behaviour, che varia a seconda del comportamento desiderato, e devono essere poi assegnati all'agente.

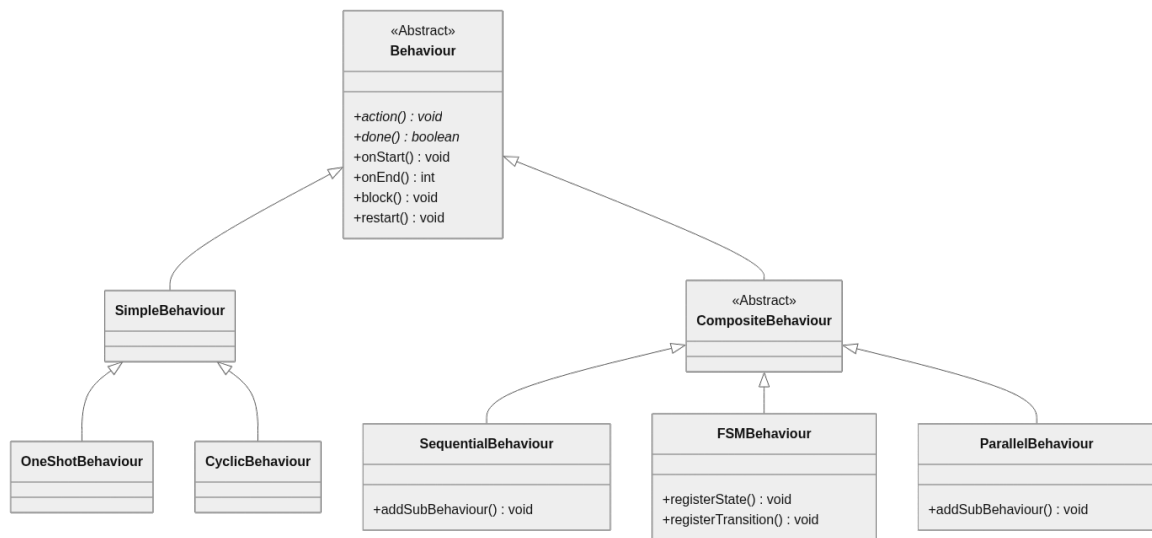


Figura 3: Panoramica dei principali Behaviours utilizzabili in JADE

Tutti i behaviours sono estensioni della classe astratta Behaviour che contiene i metodi che vengono ereditati da ogni sottoclasse. Il metodo **action()** contiene la logica principale di ogni behaviour, con tutte le operazioni da eseguire, mentre il metodo **done()** ritorna un valore booleano per indicare se il behaviour è terminato o meno. I principali behaviours utilizzati sono:

- **OneShotBehaviour**: questi behaviour vengono eseguiti una sola volta per poi terminare.
- **CyclicBehaviour**: questa tipologia di behaviour non termina mai e verrà rieseguito molteplici volte durante il ciclo di vita dell'agente.
- **SequentialBehaviour**: questa tipologia di behaviour racchiude diversi sub-behaviour e quando viene eseguito procede ad eseguire in maniera sequenziale i behaviour che racchiude, uno dopo l'altro.

3 Realismo del Simulatore

Affinché il simulatore fosse il più realistico possibile, sono state utilizzate e documentate le scelte implementative dei vari elementi per fare sì che assumessero dei valori verosimili alla realtà.

3.1 Simulazione degli Eventi Atmosferici

Siccome molte fonti di energia rinnovabile dipendono da diversi fattori atmosferici, come il meteo o la velocità del vento, si è deciso di realizzare un modello stocastico basato su catene di Markov per simulare i cambiamenti atmosferici.

In particolare, sono state create due Markov Chain, una per simulare l'andamento del meteo e una per la velocità del vento.

Questi modelli vengono creati utilizzando i dati relativi agli anni precedenti al periodo di simulazione, ricavati tramite le API di <https://open-meteo.com/en/docs>.

3.1.1 Meteo

Per quanto riguarda le previsioni meteorologiche, le informazioni ricavate dalle API sono rappresentate con dei Weather Codes. Ogni Weather Code è un numero intero che rappresenta la condizione meteorologica in un dato momento, come è possibile vedere nella figura 4.



Figura 4: Rappresentazione dei Weather Codes
Crediti: Leftium

Partendo da questi valori, viene effettuata una semplificazione degli stati raggruppando i codici meteorologici in delle macro-categorie:

- **Sunny:** uno stato che indica la presenza di un cielo sereno, corrispondente ai valori 0 e 1.
- **Cloudy:** uno stato che indica la presenza di nuvole, comprende i valori 2, 3, 45 e 48.
- **Rainy:** uno stato che indica la presenza di pioggia, include i valori 51, 53, 55, 56, 57, 61, 63, 65, 66, 67, 80, 81, 82, 95, 96 e 99.
- **Snowy:** uno stato che indica la presenza di neve, corrispondente ai valori 71, 73, 75, 77, 85 e 86.

Gli stati sopra riportati corrispondono agli stati della catena di Markov.

Dopo aver eseguito il raggruppamento in stati, la computazione della matrice di transizione della Markov Chain è triviale; infatti, per ricavare il valore da inserire nella cella (i, j) della matrice basta contare, a partire dai dati, quante volte il meteo è passato dallo stato i allo stato j sul numero totale di volte in cui il meteo è uscito dallo stato i .

3.1.2 Nuvolosità

Dato che per il calcolo dell'energia prodotta dai pannelli fotovoltaici è fondamentale conoscere anche la copertura nuvolosa, è stato deciso di associare ad ogni stato della catena di Markov appena descritta un valore di copertura nuvolosa secondo la scala okta, mostrata nella figura 5.

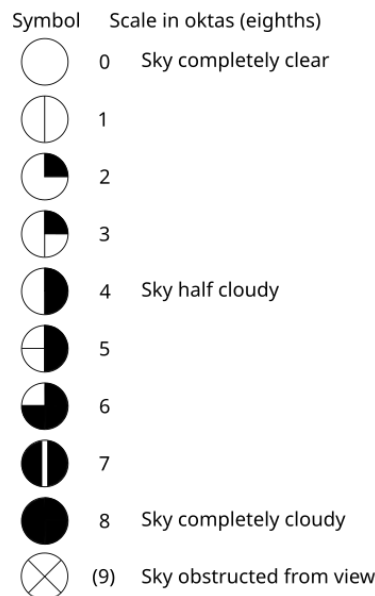


Figura 5: Scala okta per la rappresentazione della copertura delle nuvole
Crediti: Frasmacon

Per farlo, sono stati recuperati dalle API anche i dati che indicano la copertura del cielo espressa in percentuale.

Questi valori sono stati convertiti in ottavi e ad ogni stato della catena di Markov creata per il meteo è stato associato il valore medio della copertura nuvolosa per quello stato.

3.1.3 Vento

Analogamente a quanto fatto per il modello di predizione degli eventi meteorologici, il simulatore è in grado di creare la catena di Markov che permette di simulare cambiamenti nella velocità del vento, fondamentale per il calcolo dell'energia prodotta dagli impianti eolici. Sempre utilizzando le stesse API vengono recuperati i dati riguardanti la velocità del vento, in km/h .

In questo caso, i valori che indicano la velocità del vento sono continui e, di conseguenza, è stato deciso di effettuare una procedura di discretizzazione utilizzando la scala di Beaufort.

La scala di Beaufort è un sistema empirico di misurazione della forza del vento, composto da 13 stati che rappresentano diverse velocità del vento; ogni stato della scala di Beaufort corrisponde a uno stato della catena di Markov.

Forza	Stato	Velocità del vento [km/h]
0	Calm	< 1
1	Light Air	< 6
2	Light Breeze	< 12
3	Gentle Breeze	< 19
4	Moderate Breeze	< 29
5	Fresh Breeze	< 39
6	Strong Breeze	< 50
7	Near Gale	< 62
8	Gale	< 75
9	Strong Gale	< 89
10	Storm	< 103
11	Violent Storm	< 118
12	Hurricane	> 118

Tabella 1: Classificazione della velocità del vento secondo la scala di Beaufort

Il calcolo della matrice di transizione avviene analogamente a quanto fatto per la prima catena di Markov.

Dato che ad ogni stato della scala di Beaufort corrisponde un intervallo di valori, è stato deciso, in fase di simulazione, di impostare come velocità del vento un numero casuale proveniente dall'intervallo di riferimento.

3.2 Coordinate e Calcolo della Distanza

Ogni agente, tra le diverse informazioni, contiene anche una coppia di coordinate che rappresentino la sua posizione (latitudine e longitudine, in gradi decimali). Ciò permette di implementare scenari altamente realistici per quanto riguarda anche la posizione geografica degli agenti. Le coordinate sono estremamente importanti per il calcolo dell'energia che viene persa durante la trasmissione; infatti, l'energia persa dipende, tra le altre cose, anche dalla lunghezza del cavo, che coincide con la distanza tra due punti descritti da tali coordinate.

Di seguito vengono riportate le formule necessarie per calcolare la distanza tra due punti sul nostro pianeta espressi tramite coordinate.

Innanzitutto, occorre convertire la latitudine e la longitudine dai gradi al corrispettivo valore in radianti; per farlo, è sufficiente applicare la seguente formula:

$$\theta_{\text{rad}} = \theta_{\text{deg}} \cdot \frac{\pi}{180} \quad (1)$$

In cui:

- θ_{deg} è la latitudine/longitudine in gradi decimali.
- θ_{rad} è la latitudine/longitudine in radianti.

Si procede calcolando la differenza tra le latitudini (ϕ) e le longitudini (λ) (in radianti):

$$\Delta_{\phi} = \phi_1 - \phi_2 \quad (2)$$

$$\Delta_{\lambda} = \lambda_1 - \lambda_2 \quad (3)$$

Successivamente, viene utilizzata la formula di Haversine per calcolare la distanza minima tra due punti sulla superficie terrestre.

$$a = \sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right) \quad (4)$$

$$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{1-a}) \quad (5)$$

Infine, per ottenere la distanza in metri tra i due punti, è sufficiente effettuare il prodotto tra il raggio del pianeta Terra R , pari a $6371 \cdot 10^3$, e il valore appena calcolato.

$$d = R \cdot c \quad (6)$$

Ovviamente, tale lunghezza è solo un'approssimazione che rappresenta la distanza in linea d'aria tra due agenti; per forza di cose, è inverosimile che corrisponda alla distanza reale, ma si avvicina abbastanza ad essa per simulare in maniera coerente i vari scenari applicativi.

3.3 Trasporto di Energia nella Rete

Il simulatore tiene conto delle perdite di energia che avvengono durante il trasporto e delle limitazioni che nascono nel trasferimento di essa in alcune situazioni.

3.3.1 Perdita di Energia durante la Trasmissione

Per determinare la potenza persa durante la trasmissione non è sufficiente calcolare solamente la lunghezza del cavo, ma servono altre informazioni importanti.

Infatti, la formula che permette di calcolare la potenza persa (in watt) è la seguente: [5, pp. 30-34]

$$P_{\text{lost}} = R \cdot I^2 \quad (7)$$

Dove:

- R rappresenta la resistenza del cavo, in Ω .
- I rappresenta la corrente elettrica, in A .

Per calcolare la resistenza R del cavo viene utilizzata la seguente formula: [5, pp. 129-132]

$$R = \rho * \frac{L}{A} \quad (8)$$

In cui:

- ρ rappresenta la resistività del materiale, espressa in $\Omega \cdot m$.
- L rappresenta la lunghezza del cavo, in m .
- A rappresenta la sezione del cavo, espressa in m^2 .

I due materiali più usati per la realizzazione di cavi di questo tipo sono il rame e l'alluminio, che hanno una resistività rispettivamente pari a $1.68 \cdot 10^{-8}$ e $2.82 \cdot 10^{-8}$. Infine, la seguente formula permette di calcolare il valore della corrente I .

$$I = \frac{P_{\text{produced}}}{V} \quad (9)$$

In questa formula, P_{produced} rappresenta la potenza prodotta in watt, mentre V rappresenta il voltaggio del cavo in volt, quest'ultimo può cambiare a seconda del contesto di utilizzo; infatti, i cavi che lavorano con energia ad alta tensione vengono utilizzati per il trasporto su lunghe distanze, per garantire una perdita ridotta, mentre i cavi che trasportano energia a bassa tensione sono principalmente usati per la distribuzione di energia nelle case [5, pp. 17-20]. Come si può notare dalle formule, aumentando la tensione, la corrente diminuisce. Di conseguenza, anche le perdite diminuiscono; infatti, la potenza persa è proporzionale al quadrato della corrente e, pertanto, un incremento della tensione porta a una riduzione significativa delle perdite energetiche.

Infine, per calcolare l'efficienza di una linea è sufficiente utilizzare la seguente formula:

$$\eta = 1 - \frac{P_{\text{lost}}}{P_{\text{produced}}} \quad (10)$$

Ovviamente, se η è pari a 0, significa che tutta l'energia inviata sulla linea viene persa; viceversa, se η vale 1, tutta l'energia inviata raggiunge la destinazione.

Talvolta può risultare utile calcolare la potenza da trasmettere affinché una certa quantità di energia arrivi a destinazione, tenendo conto delle perdite dovute alla trasmissione. Per farlo, è sufficiente utilizzare la seguente formula:

$$P_{\text{needed}} = \frac{V^2 \pm \sqrt{V^4 - 4 \cdot R \cdot V^2 \cdot P_{\text{requested}}}}{2 \cdot R} \quad (11)$$

Naturalmente, questa formula produrrà due soluzioni, di cui sicuramente almeno una sarà positiva; nel caso fossero entrambe positive, verrà preso in considerazione il valore minimo tra le due. Altrimenti, bisogna considerare l'unica soluzione positiva.

3.3.2 Batterie

Per modellare nel modo più realistico possibile le batterie, utilizzate sia negli edifici per uso domestico sia come impianti di accumulo di energia sulla rete, è necessario calcolare quanta energia esse possano ricevere o rilasciare in un determinato lasso di tempo. Infatti, ogni batteria ha un valore diverso di corrente di carica/scarica.

Per calcolare quanta energia al massimo una batteria può rilasciare/ricevere in un intervallo temporale è sufficiente usare la seguente formula:

$$P_{\Delta t} = V \cdot I_{\text{discharge}} \cdot \eta \cdot \Delta t \quad (12)$$

Dove:

- V rappresenta il voltaggio della batteria in volt.
- $I_{\text{discharge}}$ è la corrente massima rilasciabile/ricevibile dalla batteria, in ampere.
- η è un fattore di efficienza compreso tra 0 ed 1.

- Δt rappresenta l'intervallo di tempo espresso in ore. Ad esempio, 15 minuti corrispondono a 0.25 ore.

Di conseguenza, quando nel simulatore si cercherà di caricare o scaricare una batteria, l'energia ricevibile/rilasciabile in un intervallo di tempo deve essere al massimo pari al valore calcolato con la formula 12.

3.4 Impianti Fotovoltaici

Gli impianti fotovoltaici, presenti sia sulle case che come centrali elettriche, sono stati modellati attraverso l'utilizzo delle formule presentate in questa sezione.

Dal simulatore è possibile accedere all'orario in cui sorge e tramonta il sole, in questo modo si può controllare l'ora simulata e verificare se ci si trova nella fase notturna della simulazione, dove il sole ovviamente non può raggiungere i pannelli.

L'energia prodotta viene calcolata tramite la seguente formula:

$$P_{photovoltaic} = E_{POA} \cdot A \cdot \eta \quad (13)$$

In questa formula, A rappresenta l'area totale dei pannelli fotovoltaici dell'impianto in m^2 , η rappresenta un parametro di efficienza che varia a seconda del tipo di pannello, mentre E_{poa} rappresenta l'energia che può generare l'insieme dei pannelli (Plane Of Array, POA) [6]. Il suo valore viene ricavato sommando i seguenti parametri:

$$E_{POA} = E_{beam} + E_{ground} + E_{diffuse} \quad (14)$$

Dove:

- E_{beam} rappresenta l'energia ricavata dalla quantità di luce solare diretta che colpisce la superficie del pannello.
- E_{ground} indica l'energia ricavata dalla quantità di luce riflessa dal terreno su cui è posato il pannello.
- $E_{diffuse}$ corrisponde all'energia ricavata dalla luce che viene diffusa nell'atmosfera durante l'irraggiamento.

Prima di poter compiere questo calcolo, tuttavia, è necessario conoscere diverse informazioni. Inizialmente, bisogna calcolare l'angolo zenitale θ_s del pannello, ovvero l'angolo che si crea tra la direzione dei raggi solari e la direzione perpendicolare alla superficie dell'orizzonte.

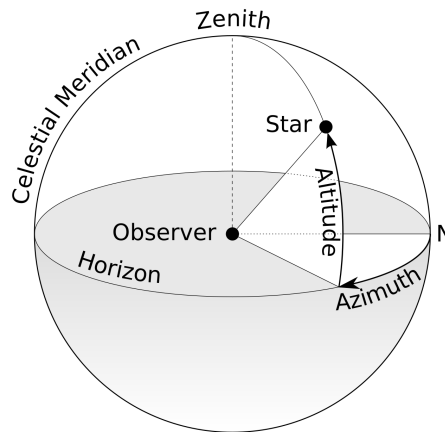


Figura 6: Rappresentazione degli angoli di Azimut e Zenit
Crediti: TWC Carlson

Secondariamente occorre calcolare l'ora solare effettiva del luogo di interesse; per farlo si utilizza la differenza tra la sua longitudine e il meridiano standard associato al fuso orario, considerando il giorno attuale per tenere traccia delle perturbazioni nate dalla rotazione della Terra attorno al Sole durante l'anno [7].

$$\text{SolarTime} = \text{LocalTime} + (\text{StandardMeridian} - \text{LocalMeridian}) \cdot 4' + \text{EoT} \quad (15)$$

Dove EoT è l'Equation of Time, necessaria per correggere la differenza tra l'ora solare effettiva e l'ora solare media, calcolata come segue:

$$\text{EoT}[\text{min}] = 9.87 \sin(2x) - 7.53 \cos(x) - 1.5 \sin(x) \quad (16)$$

Con:

$$x = \frac{360^\circ}{365} (\text{DoY} - 81) \quad (17)$$

Dove DoY (Day of the Year) rappresenta il giorno dell'anno. Grazie a queste informazioni è possibile calcolare l'angolo orario ω_s in gradi, che dipende dall'ora solare locale.

$$\omega_s = ((\text{SolarTime}[\text{h}] - 12) * 15) \quad (18)$$

In seguito, viene calcolato l'angolo di declinazione, rappresentato come δ_s , che rappresenta l'angolo che si forma tra l'equatore della Terra e una linea immaginaria tracciata dal centro della Terra al centro del Sole.

$$\delta_s = 23.45^\circ \cdot \sin(x) \quad (19)$$

In cui x è la medesima dell'equazione 17.

Una volta ricavati questi valori, è possibile calcolare $\cos(\theta_s)$

$$\cos(\theta_s) = \cos(\phi) \cdot \cos(\delta_s) \cdot \cos(\omega_s) + \sin(\phi) \cdot \sin(\delta_s) \quad (20)$$

Da cui si può facilmente ottenere l'angolo zenitale θ_s calcolando l'arcocoseno:

$$\theta_s = \arccos(\cos(\theta_s)) \quad (21)$$

Questi calcoli erano necessari per arrivare a calcolare il coseno dell'angolo di Azimut (ϕ_s), il quale viene calcolato come segue:

$$\cos(\phi_s) = \frac{\sin(\delta_s) \cdot \cos(\phi) - \cos(\omega_s) \cdot \cos(\delta_s) \cdot \sin(\phi)}{\sin(\theta_s)} \quad (22)$$

Per poter così calcolare l'angolo di incidenza AOI, che rappresenta l'incidenza con cui i raggi colpiscono il pannello [6]:

$$\text{AOI} = \arccos(\cos(\theta_s) \cdot \cos(\theta_t) + \sin(\theta_s) \cdot \sin(\theta_t) \cdot \cos(\phi_s - \phi_{s,array})) \quad (23)$$

Dove θ_t è l'angolo di inclinazione (Tilt) del pannello.

Ora, è possibile calcolare la Global Horizontal Irradiation, GHI, ovvero l'irradiazione solare che colpisce una superficie orizzontale, utilizzando i modelli di Kasten-Czeplak [8, pp. 67–73]

$$\text{GHI} = \text{GHI}_{\text{clear}} \cdot \left(1 - 0.75 \cdot \left(\frac{\text{CloudCover}}{8} \right)^{3.4} \right) \quad (24)$$

CloudCover rappresenta la copertura del cielo creata dalle nuvole espressa sulla base della scala okta (illustrata nella figura 5), $\text{GHI}_{\text{clear}}$ rappresenta il valore di GHI che viene assunto quando il cielo è completamente sereno. Questo valore si può calcolare utilizzando il modello di Berger–Duffie (BD) [7]:

$$\text{GHI}_{\text{clear}} = I_0 \cdot 0.7 \cdot \cos(\theta_s) \quad (25)$$

Nella formula, I_0 corrisponde all'indice di radiazione extraterrestre, ovvero la radiazione che raggiunge la parte esterna dell'atmosfera terrestre, che viene calcolata come segue:

$$I_0 = 1367.7 \cdot \left(1 + 0.033 \cdot \cos\left(\frac{2\pi}{365} \cdot \text{DoY}\right) \right) \quad (26)$$

Dove 1367.7 rappresenta la costante solare globale (GSC), la quale misura la quantità di energia ricevuta da una data area a un'unità astronomica di distanza dal Sole.

Grazie a queste informazioni è inoltre possibile calcolare il clearness index (K_t) [9]:

$$K_t = \frac{\text{GHI}}{I_0 \cdot \max(0.065, \cos(\theta_s))} \quad (27)$$

Adesso, è possibile calcolare la Diffuse Horizontal Irradiance (DHI) utilizzando la formula di correlazione di Erbs [8, pp. 65–79].

$$\text{DHI} = \begin{cases} 0.165 & 0.8 < K_t \leq 1 \\ 0.951 - 0.16 \cdot K_t + 4.388 \cdot (K_t)^2 - 16.64 \cdot (K_t)^3 + 12.34 \cdot (K_t)^4 & 0.22 \leq K_t \leq 0.8 \\ 1 - 0.09 \cdot K_t & 0 \leq K_t < 0.22 \end{cases} \quad (28)$$

A questo punto, si può calcolare la Direct Normal Irradiation (DNI), ovvero la parte di irradiazione solare che raggiunge direttamente la superficie del pannello.

$$\text{DNI} = \frac{\text{GHI} - \text{DHI}}{\cos(\theta_s)} \quad (29)$$

Adesso, sono disponibili tutte le informazioni per calcolare il risultato dell'equazione 14:

$$E_{\text{beam}} = \text{DNI} \cdot \cos(\text{AOI}) \quad (30)$$

$$E_{\text{ground}} = \text{GHI} \cdot \text{albedo} \cdot \frac{(1 - \cos(\theta_t))}{2} \quad (31)$$

L'albedo è la frazione di GHI che viene riflessa dal terreno, idealmente assume un valore molto vicino a 0 se la superficie è molto scura e un valore tendente a 1 se la superficie è metallica o richiama un bianco acceso. Di seguito vengono riportati alcuni esempi con i relativi valori di albedo [6]

Superficie	Albedo
Ambiente urbano	0.14 - 0.22
Erba	0.15 - 0.25
Erba fresca	0.26
Neve fresca	0.82
Neve bagnata	0.55 - 0.75
Asfalto asciutto	0.09 - 0.15
Asfalto bagnato	0.18
Calcestruzzo	0.25 - 0.35
Piastrelle rosse (tegole)	0.33
Alluminio	0.85
Rame	0.74
Acciaio zincato nuovo	0.35
Acciaio zincato molto sporco	0.08

Tabella 2: Valori di albedo per diverse superfici

Infine, si può calcolare l'ultimo valore utilizzando il modello di diffusione isotropico [6]:

$$E_{\text{diffuse}} = \text{DHI} \cdot \left(\frac{1 + \cos(\theta_t)}{2} \right) \quad (32)$$

3.5 Impianti Eolici

Gli impianti eolici modellati utilizzano le informazioni sul vento ottenute dal simulatore per calcolare la quantità di energia prodotta. Ogni turbina ha dei limiti di velocità, che variano da impianto a impianto, forniti al simulatore in fase di configurazione. Se la velocità del vento non rientra in questa fascia operativa, la turbina viene spenta dato che una velocità troppo bassa non sarebbe sufficiente per produrre energia, e viceversa una velocità troppo alta rischierebbe di danneggiare l'impianto.

Per calcolare la potenza prodotta dalle centrali eoliche (in W) è necessario applicare la seguente formula [10, p. 6]:

$$P_{\text{wind}} = 0.5 \cdot \rho \cdot V^3 \cdot A \cdot C_p \quad (33)$$

Dove:

- ρ rappresenta la densità dell'aria, pari a 1.225 kg/m^3 .
- V rappresenta la velocità del vento, in m/s .
- A rappresenta la superficie spazzata dal rotore in m^2 , calcolabile come $\pi \cdot \text{raggio del rotore}^2$.
- C_p rappresenta il coefficiente di potenza e può assumere solo valori inferiori a 0.593 (Limite di Betz).
- 0.5 rappresenta l'efficienza degli impianti eolici.

Questo valore va in seguito moltiplicato per il numero di pale eoliche presenti nell'impianto, in quanto la formula permette di calcolare la potenza generata da una singola turbina.

Si noti che, dopo aver raggiunto la potenza nominale, l'energia prodotta non aumenta ulteriormente all'aumentare della velocità del vento. Per questo motivo, viene effettuato un aggiustamento se la velocità del vento supera la velocità nominale rimanendo sotto ai limiti della turbina. La velocità nominale viene descritta come segue:

$$V_{\text{nom}} = \frac{V_{\text{max}} - V_{\text{min}}}{2} \quad (34)$$

In particolare, in questo caso la velocità V diventa pari a V_{nom} ai fini del calcolo della potenza generata.

3.6 Impianti Idroelettrici

Per quanto riguarda le centrali idroelettriche, la formula utilizzata per calcolare l'energia prodotta, presentata in [11], è la seguente:

$$P_{\text{hydro}} = \eta \cdot \rho \cdot Q \cdot g \cdot H \quad (35)$$

In cui:

- ρ è la densità dell'acqua che passa dal generatore, in kg/m^3 .
- Q è il volume di acqua che passa attraverso la turbina, in m^3/s .
- g è l'accelerazione dovuta alla gravità, in m/s^2 .
- H è l'altezza del salto compiuto dal corso d'acqua per arrivare alla turbina, in m .
- η è un fattore di efficienza, compreso tra 0 e 1, che cambia da generatore a generatore.

Questa formula permette di modellare le centrali idroelettriche a filo d'acqua, ovvero le centrali che sfruttano il flusso naturale di un corso d'acqua, senza dover creare bacini di accumulo.

3.7 Impianti Diesel

Il generatore diesel ideato per questo simulatore consiste in una versione semplificata, dove l'energia prodotta è ottenuta calcolando il prodotto tra la potenza del motore e un fattore di efficienza:

$$P_{\text{diesel}} = P_{\text{engine}} \cdot \eta \quad (36)$$

3.8 Conversione da Watt a Wattora

Dato che il simulatore lavora con intervalli di tempo di lunghezza fissa, è necessario convertire i valori di energia da watt a wattora, adattandoli agli intervalli di tempo simulati per ottenere i valori corretti. Per fare ciò viene utilizzata la seguente formula:

$$P [Wh] = P [W] \cdot \Delta t \quad (37)$$

Dove Δt indica la durata dell'intervallo di tempo espresso in ore. Ad esempio, 15 minuti corrispondono a 0.25 ore.

4 Sviluppo del Simulatore

4.1 Sistema a Turni

Per l'implementazione del sistema è stato deciso di adottare una logica a turni; la durata del turno può essere impostata dal file `app.config`. I turni permettono di: gestire comodamente la simulazione, calcolare accuratamente l'energia persa e scandire il tempo in maniera molto precisa.

Il sistema a turni permette di evitare asincronie temporali nella simulazione, passando al turno successivo solo quando tutti gli agenti hanno svolto i loro compiti, utilizzando un meccanismo di sincronizzazione su condizione a barriera. Senza questa scelta implementativa, dato che ogni agente ha compiti diversi durante ogni turno, nasceva il rischio che alcuni agenti si trovassero in un istante temporale diverso rispetto agli altri.

La logica di simulazione e i turni sono gestiti da un apposito agente chiamato `SimulationAgent` che verrà illustrato in seguito.

Inoltre, si presuppone che i consumi e le produzioni che avvengono in un turno rimangano costanti durante tutta la sua durata.

4.2 Configurazione del Simulatore

Il simulatore creato accetta diversi parametri per configurare e personalizzare la singola simulazione, i quali vanno specificati nel file `app.config`. I parametri da inserire sono i seguenti:

- **turn_duration**: una stringa formattata come `hh:mm` che rappresenta la durata temporale di un turno. Ad esempio, il valore `00:30` significa che ogni turno corrisponde a 30 minuti di simulazione.
- **interval_between_turns**: un numero intero che rappresenta il numero di millisecondi di attesa che intercorrono tra la fine di un turno e l'inizio del turno successivo.
- **save_interval**: una stringa formattata come `hh:mm` che indica ogni quanto esportare i dati generati dal simulatore.
- **simulation_start_date**: una stringa contenente la data in cui inizia la simulazione, strutturata come `yyyy-mm-dd`.
- **latitude**: latitudine del luogo della simulazione in gradi, espressa come numero reale.
- **longitude**: longitudine del luogo della simulazione in gradi, espressa come numero reale.
- **weather_turn_duration**: una stringa formattata come `hh:mm` che rappresenta l'intervallo di tempo che trascorre da un evento meteorologico all'altro. Tale valore deve essere un multiplo di **turn_duration**. Ad esempio, un valore pari a `01:00` significherebbe che ogni due turni il meteo potrebbe cambiare se il parametro di **turn_duration** è impostato a `00:30`.
- **scenario_name**: il nome dello scenario che verrà utilizzato, deve corrispondere al nome della relativa cartella contenente le informazioni dei vari agenti, che verrà descritta in seguito.

- **price_volatility**: la percentuale di cambiamento che può avere il prezzo dell'elettricità durante la simulazione, solitamente compreso nell'intervallo $[0, 1]$.
- **price_trend**: l'andamento a lungo termine del prezzo dell'elettricità. Un valore positivo indica una tendenza a salire e viceversa per un valore negativo.
- **random_seed**: un seed utilizzato per la riproducibilità delle simulazioni, permette di generare sempre gli stessi numeri casuali.

4.3 Componenti del Sistema

Per realizzare il simulatore sono stati inizialmente individuati tutti i componenti utili per la realizzazione del sistema.

Analizzando alcuni studi effettuati su scenari applicativi simili a quello descritto in questo progetto, come [12] e [2], sono stati individuati i componenti più importanti di un simulatore di questo tipo; vengono riportati di seguito gli agenti modellati:

- **Smart Building**, un agente che rappresenta un edificio intelligente ed autonomo, è in grado di produrre energia se dotato di pannelli fotovoltaici, di immagazzinare energia se dotato di batteria e può richiedere energia alla Grid o rilasciare in rete eventuali eccessi. Ogni edificio possiede anche una priorità che può essere alta, media o bassa.
- **Grid**, un agente che rappresenta una griglia di distribuzione, è in grado di: ricevere energia dai Power Plant ad essa connesse, raccogliere le richieste di energia da parte degli edifici sotto la sua area di competenza e condividere l'energia con le Grid a cui è collegata. La priorità di una Grid è pari alla priorità massima delle richieste che ha ricevuto dagli edifici. Alcune Grid potrebbero avere un impianto di accumulo per immagazzinare l'energia in eccesso e distribuirla in un successivo momento, in caso di necessità.
- **Power Plant**, un agente che rappresenta una centrale elettrica, può essere di tipo rinnovabile o non; la centrale, se accesa, produce l'energia e la affida alla Grid a cui è connessa. Esempi di Power Plant possono essere impianti fotovoltaici, eolici, idroelettrici e diesel.
- **Load Manager**, un agente che rappresenta un'entità astratta, che ha lo scopo di guidare le Grid nell'instradamento intra-grid dell'energia, per assicurarsi di distribuirla nel miglior modo possibile. Per come è realizzato il simulatore ad oggi è possibile specificare un solo Load Manager.

Oltre agli agenti sopra riportati, ne sono stati sviluppati degli altri con lo scopo di rendere la simulazione più personalizzabile.

- **Simulation Agent**, un agente astratto che ha il compito di gestire la simulazione. Questo agente deve adempiere a vari compiti, tra cui l'inizializzazione dell'ambiente di simulazione, dove vengono creati tutti gli altri agenti. Inoltre, si occupa di gestire l'organizzazione dei turni, sia dal punto di vista logico che temporale, e di simulare i cambiamenti meteorologici utilizzando le Markov Chain descritte in 3.1. Permette inoltre all'utente di interrompere e far ripartire la simulazione quando lo desidera.
- **Owner**, un agente che rappresenta il proprietario di uno o più edifici; può ricevere messaggi da un utente esterno (chi utilizza il simulatore) per modificare le richieste energetiche degli edifici di sua proprietà, in modo tale da simulare cambiamenti durante la simulazione.

4.4 Modellazione degli Agenti

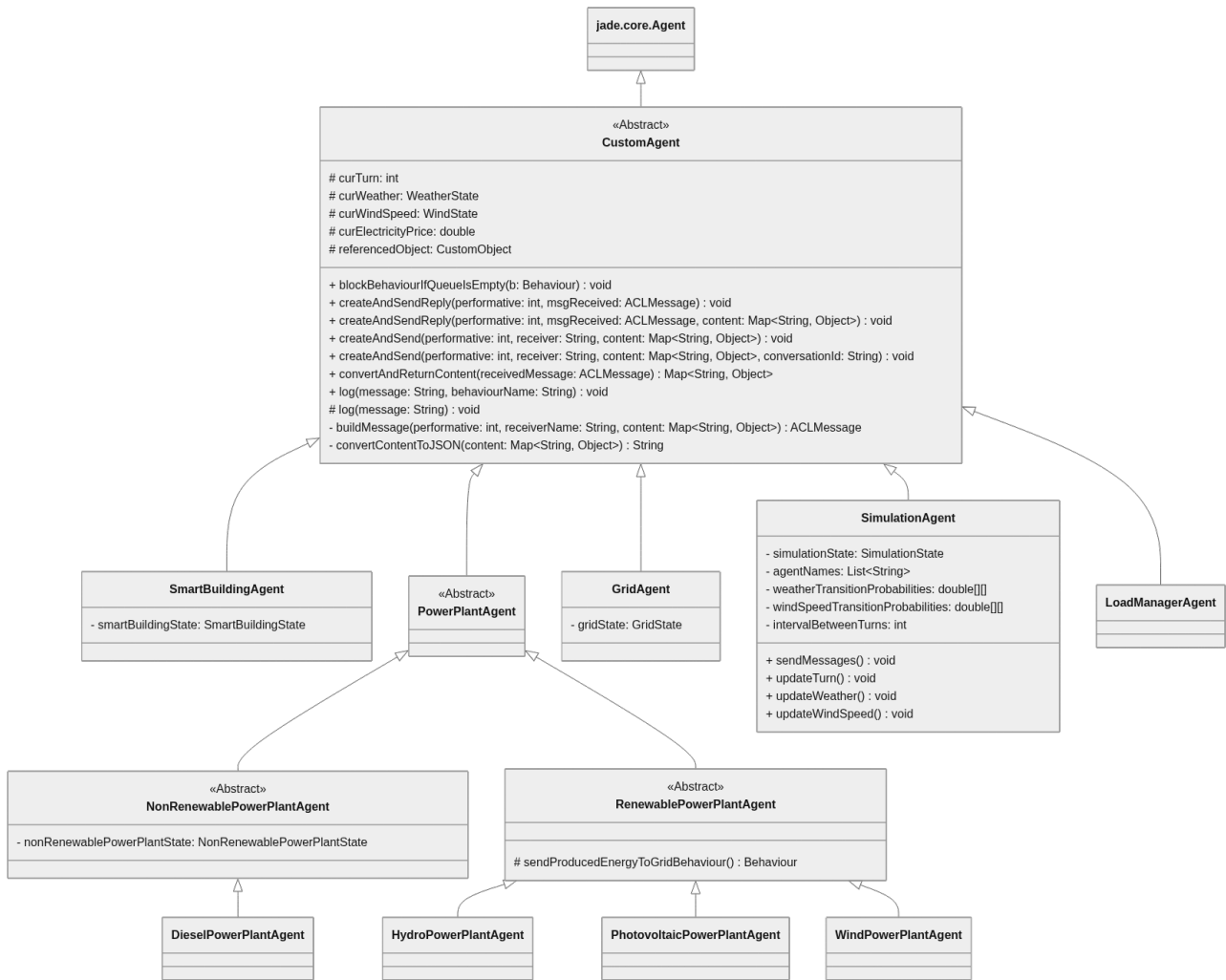


Figura 7: Architettura degli agenti utilizzabili nel simulatore

4.4.1 Smart Building

Ogni Smart Building è un agente di tipo isteretico, ovvero un agente che possiede uno stato interno; il suo stato può assumere tre valori: Blackout, Losing Energy e Gaining Energy. All'inizio di ogni turno l'edificio calcola il consumo che dovrà sostenere e l'eventuale produzione di energia qualora fosse dotato di pannelli fotovoltaici. Inoltre, calcola il consumo previsto per il prossimo turno in base alla Routine salvata in memoria.

Ogni edificio è caratterizzato da un insieme di Appliances descritti da un nome e dal relativo consumo. Gli orari di utilizzo di tali Appliances possono essere specificati tramite una Task. Ogni edificio è infatti dotato di una Routine, ovvero un insieme di Task che deve seguire per simulare il consumo energetico dell'edificio in questione.

Se l'edificio si trova nello stato Gaining Energy, significa che la sua produzione interna (derivante dall'impianto fotovoltaico e dall'energia presente nella sua batteria) è abbastanza da poter sostenere autonomamente i consumi di questo turno e, di conseguenza, può anche ricaricare la propria batteria ed eventualmente rilasciare l'energia in eccesso nella Grid.

Se invece si trova nello stato Losing Energy, significa che la produzione interna non è abbastanza per soddisfare i consumi durante questo turno e, di conseguenza, l'edificio manda una richiesta

di energia alla Grid. Quest'ultima potrà decidere di accettarla o rifiutarla, rispondendo in base alle proprie disponibilità. Ogni messaggio di questo tipo conterrà, oltre alla quantità di energia richiesta, anche la priorità dell'edificio.

Infine, se l'edificio ha richiesto un certo quantitativo di energia e la Grid non può soddisfare questa richiesta, allora l'edificio andrà nello stato Blackout. In questo stato l'edificio spegnerà tutti i propri Appliances in attesa di un rilascio di energia da parte della Grid. Qualora l'edificio avesse a disposizione dei pannelli fotovoltaici e una batteria, cercherà di ricaricarla finché non raggiunge una quantità di energia sufficiente per ripartire in autonomia. Altrimenti, se l'edificio è dotato di pannelli ma non di una batteria, l'energia prodotta verrà rilasciata direttamente nella rete.

Questi agenti possono essere utilizzati anche per rappresentare scenari più articolati, dove modellare interi quartieri con centinaia di case diventerebbe impraticabile. Infatti, assegnando agli Appliances dei consumi maggiori, che simulerebbero i consumi di più edifici, e tarando i relativi cavi con dei valori realistici, è possibile utilizzare questo tipo di agente per rappresentare un quartiere o un agglomerato di abitazioni.

4.4.2 Renewable Power Plant

Le centrali elettriche rinnovabili sono agenti tropistici estremamente semplici, che non possiedono uno stato interno. Il loro compito è quello di calcolare quanta energia produrranno in questo turno, in base alle condizioni meteorologiche, e inviare l'energia prodotta alla Grid a cui sono collegati.

4.4.3 Non-Renewable Power Plant

Le centrali elettriche non rinnovabili sono agenti di tipo isteretico, che possono trovarsi in due stati: acceso o spento.

Alla fine di ciascun turno, ogni centrale non rinnovabile riceve un messaggio da parte del Load Manager che indica se deve accendersi per contribuire alla produzione energetica e, in tal caso, quanta energia deve produrre per il prossimo turno.

4.4.4 Grid

Le Grid sono degli agenti isteretici, possono assumere due stati: Receive e Send, a seconda della quantità di energia richiesta e/o ricevuta.

All'inizio del turno ogni Grid riceve le richieste ed emissioni di energia e la previsione del consumo del turno successivo da parte degli edifici sotto la propria area di competenza. Successivamente, ottengono l'energia inviata dai Power Plant rinnovabili e non, e a seconda dei valori ricevuti cambiano il proprio stato. Se l'energia richiesta è superiore a quella ricevuta, la Grid entrerà nello stato Receive, altrimenti nello stato Send. A questo punto, ogni Grid invia un messaggio al Load Manager contenente:

- il proprio stato;
- la quantità di energia di cui ha bisogno o che può trasmettere ad altre Grid a seconda dello stato in cui si trova;
- un livello di priorità equivalente al livello massimo di priorità delle richieste di energia ricevute;
- le informazioni sulla propria batteria, se presente;

- la previsione sul consumo del prossimo turno;
- la quantità di energia ricevuta da fonti rinnovabili.

Dopo aver inviato le proprie informazioni, attende le istruzioni di distribuzione dal Load Manager. Una volta ricevute le istruzioni di instradamento dal Load Manager, ogni Grid procederà a seguire tali indicazioni, inoltrando l'energia alle rispettive Grid destinatarie. Infine, la Grid procederà con la trasmissione dell'energia a sua disposizione ai Building che ne necessitano e, qualora ci fosse dell'energia in eccesso, essa verrà utilizzata per caricare il proprio sistema di accumulo.

4.4.5 Load Manager

Il Load Manager è un agente "knowledge-based". Esso percepisce l'ambiente tramite i messaggi che riceve dalle Grid. Il suo compito è quello di stabilire come le diverse Grid debbano scambiarsi la corrente tra loro, in modo tale da ottimizzare la ripartizione dell'energia minimizzando le perdite e soddisfacendo il maggior numero di richieste possibili.

Il Load Manager conosce la struttura della rete elettrica che connette le Grid e la memorizza sotto forma di grafo durante l'inizializzazione; in particolare, ogni Grid rappresenta un nodo a cui in ogni turno viene assegnato uno stato, un livello di priorità, una quantità di energia e le informazioni sulla propria batteria come descritto in 4.4.4. Ogni arco che connette due nodi corrisponde a un cavo esistente tra due Grid; ciascun arco è etichettato da un costo che dipende dalla sua resistività, voltaggio e lunghezza; più il costo è elevato, meno è conveniente trasmettere energia attraverso quell'arco.

Le istruzioni di distribuzione vengono determinate attraverso tre fasi, eseguite sequenzialmente.

1. Distribuzione dell'energia fornita dalle Grid: in un primo momento, il Load Manager identifica i nodi produttori, caratterizzati dallo stato Send, e nodi richiedenti, caratterizzati dallo stato Receive. L'algoritmo opera seguendo un ordine di priorità: inizialmente, vengono presi in considerazione i nodi richiedenti con priorità massima, cercando di soddisfare il maggior numero di richieste distribuendo l'energia dei nodi produttori. Una volta che tutti i nodi aventi priorità massima sono stati soddisfatti, si ripete la medesima operazione considerando la priorità direttamente inferiore, finché non vengono soddisfatte tutte le richieste o fino a quando terminano i produttori. L'energia viene distribuita cercando il percorso di costo minimo che connette il consumatore e un produttore, calcolato con Dijkstra. Una volta individuato il cammino ottimale, il Load Manager calcola quanta energia deve essere inviata, comprensiva della possibile perdita che avviene durante la trasmissione nel percorso.
2. Distribuzione dell'energia delle batterie: questa fase viene effettuata solo se ci sono ancora dei nodi richiedenti. Il Load Manager cercherà di attingere dall'energia immagazzinata nelle batterie sulla rete per poter soddisfare il maggior numero di richieste procedendo in maniera simile a quanto descritto nella fase 1, con l'unica differenza che in questo caso i nodi produttori sono i nodi che possiedono una batteria utilizzabile.
3. Distribuzione dell'energia extra: infine, se tutte le richieste delle Grid sono state soddisfatte ed è presente dell'energia in eccesso, questa corrente verrà instradata verso le Grid dotate di un sistema di accumulo, per evitare sprechi.

Una volta determinate le istruzioni di distribuzione, esse vengono inoltrate alle Grid in modo tale che possano condividere l'energia secondo tali indicazioni.

Infine, il Load Manager determina quali centrali non rinnovabili è necessario attivare nel prossimo turno per soddisfare le richieste previste.

Per fare ciò, inizialmente analizza la previsione dei consumi delle Grid per il turno successivo e verifica lo stato delle batterie. Se il livello di carica di una batteria risulta inferiore al 25%, calcolerà la quantità di potenza richiesta per iniziare una fase di ricarica con l'obiettivo di raggiungere la soglia del 75%.

Sulla base di queste informazioni, della quantità di energia rinnovabile prodotta in questo turno e dell'energia in surplus, il Load Manager determina quali centrali non rinnovabili sarà necessario attivare nel turno successivo.

Per calcolare quanta energia sarà necessario produrre con le centrali non rinnovabili il prossimo turno, viene utilizzata la seguente formula:

$$P = C - 0.7 \cdot (P_{\text{surplus}} + P_{\text{renewable}}) + C_{\text{battery}} \quad (38)$$

Dove:

- C rappresenta il consumo previsto per il prossimo turno, in Wh .
- C_{Battery} rappresenta l'energia necessaria per caricare le batterie, in Wh .

A seconda della quantità di energia richiesta, verrà acceso il numero minimo di centrali non rinnovabili in funzione della produzione di energia che possono fornire durante un turno per soddisfare tale richiesta.

4.5 Ambiente di Simulazione

È importante definire le caratteristiche dell'ambiente in cui sono situati gli agenti per modellare al meglio il simulatore. Nello specifico, le caratteristiche dell'ambiente sviluppato in questo progetto sono le seguenti:

- **Inaccessibile:** gli agenti non possono accedere direttamente ad ogni informazione presente nell'ambiente.
- **Non deterministico:** non tutte le azioni degli agenti hanno un esito pre-determinato.
- **Sequenziale:** le azioni degli agenti non dipendono solamente da ciò che percepiscono in un determinato momento, ma anche da ciò che è successo in precedenza o dal loro stato.
- **Dinamico:** l'ambiente non cambia solamente attraverso le azioni degli agenti, ma dipende anche da fattori esterni come le condizioni meteorologiche.
- **Discreto:** ogni agente ha un numero finito di stati e azioni che può compiere.

4.6 Interazione tra Agenti e Protocollo di Comunicazione

Ogni agente può comunicare direttamente con gli altri agenti che sono fisicamente collegati ad esso, ad eccezione delle due figure astratte rappresentate da Simulation Agent e Load Manager. È stato scelto un modello di interazione diretto, in particolare con conoscenza a priori, in quanto ogni agente conosce già dal principio i nomi degli agenti con cui deve comunicare senza la necessità di interrogare un agente di mezzo.

È stata scelta questa modalità di interazione in quanto il sistema ad agenti utilizzato per creare il simulatore è un sistema chiuso, dato che non prevede che nuovi agenti vengano aggiunti durante l'esecuzione.

Inoltre, lo scambio di informazioni avviene attraverso messaggi conformi allo standard FIPA, descritto in precedenza, dove il contenuto corrisponde a un oggetto JSON.

4.6.1 Discovery iniziale

Inizialmente gli agenti conoscono solo i nomi degli altri agenti con cui devono comunicare, che vengono forniti nello scenario di riferimento. Durante la fase di setup, prima dell'effettivo inizio della simulazione, avviene uno scambio di messaggi di discovery per conoscere le coordinate degli agenti confinanti. Durante questa fase, organizzata mediante l'uso della tecnica di coordinazione, ogni agente manda le proprie coordinate ai suoi vicini. Queste informazioni sono fondamentali perché verranno poi utilizzate per calcolare la distanza effettiva tra i vari agenti ed effettuare i calcoli necessari per il trasporto corretto dell'energia.

Campo	Valore
Sender	Agent-1
Receiver	Agent-2
Conversation ID	-
Performative	INFORM
Content	{ "operation": "discovery", "latitude": 2.8073871646, "longitude": 104.1411802 }

Tabella 3: Struttura di un messaggio di coordinate discovery

Al termine di questa fase, gli agenti Grid e Non-Renewable Power Plant inviano al Load Manager alcune informazioni aggiuntive. In particolare, i Power Plant inviano al Load Manager la massima quantità di energia che possono fornire durante un turno.

Campo	Valore
Sender	NonRenewablePowerPlant-3
Receiver	LoadManager-1
Conversation ID	-
Performative	INFORM
Content	{ "maxTurnProduction": 37500.0, "on": true }

Tabella 4: Struttura di un messaggio di non-renewable discovery

Invece, la Grid comunica i nomi delle Grid ad essa collegate ed i rispettivi cavi, in modo tale che il Load Manager possa costruire il grafo su cui eseguire gli algoritmi per organizzare l'instradamento dell'energia.

Campo	Valore
Sender	Grid-1
Receiver	LoadManager-1
Conversation ID	cable-discovery
Performative	INFORM
Content	<pre> { "cableCosts": [{ "cableSection": 2.5E-6, "resistivity": 1.68E-8, "voltage": 20000.0, "length": 5271.5607, "cableResistance": 35.42488, "from": "Grid-1", "to": "Grid-2", "cableType": "copperMedium2.5" }, // ...] }</pre>

Tabella 5: Struttura di un messaggio di cable discovery

4.6.2 Simulation Agent

Questo agente è un'entità astratta che ha il solo compito di gestire e orchestrare il corretto andamento temporale dei vari agenti.

Per far partire la simulazione al primo avvio è necessario inviare un messaggio attraverso un Dummy Agent.

Campo	Valore
Sender	DummyAgent
Receiver	SimulationAgent
Conversation ID	start-simulation
Performative	REQUEST
Content	-

Tabella 6: Struttura di un messaggio di inizio simulazione

É inoltre possibile fermare e far ripartire la simulazione; per farlo è necessario inviare i seguenti messaggi:

Campo	Valore
Sender	DummyAgent
Receiver	SimulationAgent
Conversation ID	stop-simulation
Performative	REQUEST
Content	-

Tabella 7: Struttura di un messaggio di interruzione della simulazione

Campo	Valore
Sender	DummyAgent
Receiver	SimulationAgent
Conversation ID	resume-simulation
Performative	REQUEST
Content	-

Tabella 8: Struttura di un messaggio di ripresa della simulazione

Una volta iniziata la simulazione, all’inizio di ogni turno Simulation Agent invia ad ogni agente un messaggio per comunicarne l’effettivo inizio; nel messaggio sono anche presenti le informazioni sul meteo, sulla velocità del vento e sul prezzo dell’energia elettrica corrente.

Di seguito viene riportato il contenuto di un messaggio di inizio turno:

Campo	Valore
Sender	SimulationAgent
Receiver	Agent-1
Conversation ID	turn-Agent-1
Performative	INFORM
Content	{ "currentTurn": 0, "currentWindSpeed": 0, "currentWeather": 0, "currentElectricityPrice": 0.049 }

Tabella 9: Struttura di un messaggio di inizio turno

Una volta mandati tutti i messaggi, attende che ogni singolo agente comunichi la fine del proprio turno e, quando riceve tutti i messaggi di fine turno, aggiorna le impostazioni della simulazione e invia a tutti gli agenti il messaggio di inizio del turno successivo.

Campo	Valore
Sender	Agent-1
Receiver	SimulationAgent
Conversation ID	turn-Agent-1
Performative	INFORM
Content	-

Tabella 10: Struttura di un messaggio di fine turno

Inoltre, è possibile rappresentare lo scambio dei messaggi tra i vari agenti con un diagramma di sequenza.

Di seguito viene riportato l'esempio di un turno generico di esecuzione.

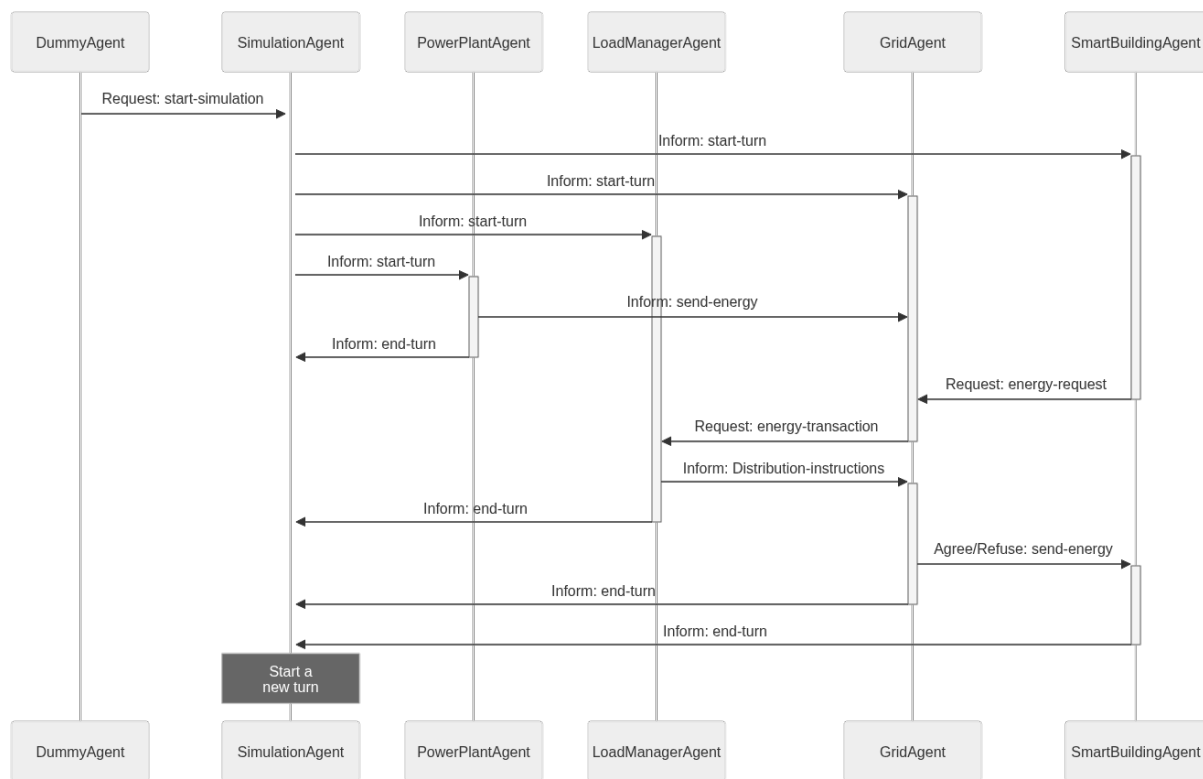


Figura 8: Diagramma di sequenza che riporta uno scambio di messaggi generico che avviene durante un turno

4.6.3 Smart Building - Grid

La comunicazione tra Smart Building e Grid può essere composta da più messaggi. Se lo Smart Building si trova nello stato di Losing Energy, invierà alla Grid una richiesta di energia. Un esempio di questo tipo di messaggio viene riportato di seguito:

Campo	Valore
Sender	SmartBuilding-2
Receiver	Grid-1
Conversation ID	-
Performative	REQUEST
Content	{ "blackout": false, "energyTransaction": { "type": "EnergyTransactionWithoutBattery", "priority": "LOW", "energyTransactionValue": 761.6666, "nodeName": "SmartBuilding-2", "transactionType": "RECEIVE" }, "nextTurnExpectedConsumption": 869.5366 }

Tabella 11: Struttura di un messaggio di richiesta di energia da parte dello Smart Building

La Grid in risposta potrà generare un messaggio di **AGREE** per indicare il fatto che è in grado di soddisfare la richiesta o un messaggio di **REFUSE** in caso contrario.

Campo	Valore
Sender	Grid-1
Receiver	SmartBuilding-2
Conversation ID	-
Performative	AGREE / REFUSE
Content	{ "operation": "consume", "requestedEnergy": 761.6666 }

Tabella 12: Struttura di un messaggio di risposta da parte della Grid

Se invece l'edificio si trova nello stato Gaining Energy e ha caricato la propria batteria, può rilasciare l'energia in eccesso; in questo caso, non si aspetterà nessuna risposta da parte della Grid.

Campo	Valore
Sender	SmartBuilding-2
Receiver	Grid-1
Conversation ID	-
Performative	REQUEST
Content	{ "blackout": false, "energyTransaction": { "type": "EnergyTransactionWithoutBattery", "priority": "LOW", "energyTransactionValue": 0.0, "nodeName": "SmartBuilding-2", "transactionType": "SEND" }, "nextTurnExpectedConsumption": 0.0 }

Tabella 13: Struttura di un messaggio di rilascio di energia da parte dello Smart Building

Se l'edificio è nello stato di Blackout, ha un sistema di produzione di energia e non possiede una batteria, rilascerà l'energia prodotta in rete.

Campo	Valore
Sender	SmartBuilding-2
Receiver	Grid-1
Conversation ID	-
Performative	REQUEST
Content	{ "blackout": true, "energyTransaction": { "type": "EnergyTransactionWithoutBattery", "priority": "MEDIUM", "energyTransactionValue": 0.0, "nodeName": "SmartBuilding-2", "transactionType": "SEND" }, "nextTurnExpectedConsumption": 950.0 }

Tabella 14: Struttura di un messaggio di rilascio di energia da parte di uno Smart Building in Blackout

Infine, se l'agente Smart Building si trova nello stato Blackout, invierà un solo messaggio alla Grid informandola sul suo stato per il turno corrente. Questo messaggio è necessario siccome gli edifici possono ripristinarsi autonomamente e la Grid deve sapere se hanno ancora bisogno di energia o meno.

Campo	Valore
Sender	SmartBuilding-2
Receiver	Grid-1
Conversation ID	-
Performative	INFORM
Content	{ "blackout": false, "nextTurnExpectedConsumption": 542.5677 }

Tabella 15: Struttura di un messaggio di Blackout da parte di uno Smart Building

Infine, se un edificio è in Blackout e la Grid ha abbastanza energia per ripristinarlo, essa invierà un messaggio di questo tipo:

Campo	Valore
Sender	Grid-1
Receiver	SmartBuilding-1
Conversation ID	restore-SmartBuilding-1
Performative	INFORM
Content	{ "givenEnergy": 456.7772 }

Tabella 16: Struttura di un messaggio di ripristino di uno Smart Building

Altrimenti, se la Grid non può ripristinare un edificio, invierà il seguente messaggio:

Campo	Valore
Sender	Grid-1
Receiver	SmartBuilding-1
Conversation ID	restore-SmartBuilding-1
Performative	INFORM
Content	{ "givenEnergy": -1.0 }

Tabella 17: Struttura di un messaggio di ripristino non possibile

4.6.4 Grid - Power Plant

La comunicazione tra l'agente Grid e il Power Plant è abbastanza semplice: la centrale invia un messaggio contenente l'energia prodotta (in *Wh*) per il turno in corso.

Campo	Valore
Sender	WindPowerPlant-1
Receiver	Grid-3
Conversation ID	-
Performative	INFORM
Content	{ "givenEnergy": 100000.0 }

Tabella 18: Struttura di un messaggio di produzione dei Power Plant

4.6.5 Load Manager - Grid

La comunicazione tra Grid e Load Manager avviene in due passaggi: in un primo momento, la Grid invia le sue informazioni al Load Manager, impostando un messaggio con la seguente struttura:

Campo	Valore
Sender	Grid-3
Receiver	LoadManager-1
Conversation ID	-
Performative	REQUEST
Content	{ "currentTurnEnergyProduction": 0.0, "energyTransaction": { "type": "EnergyTransactionWithBattery", "priority": "HIGH", "energyTransactionValue": 50184.5661, "nodeName": "Grid-3", "transactionType": "SEND", "battery": { "voltage": 400.0, "maxCapacityInWattHour": 4320000.0, "maxCapacityInAmpHour": 10800.0, "storedEnergy": 3456000.0, "dischargeCurrent": 350.0, "efficiency": 0.9, "stateOfCharge": 0.8 } }, "nextTurnExpectedConsumption": 3808.3333 }

Tabella 19: Struttura di un messaggio da Grid a Load Manager

Il Load Manager, dopo aver determinato il miglior modo di distribuire l'energia, invia delle Distribution Instructions ad ogni Grid, con un messaggio di questo tipo:

Campo	Valore
Sender	LoadManager-1
Receiver	Grid-3
Conversation ID	-
Performative	INFORM
Content	<pre>{ "activeNonRenewablePowerPlants": [{ "name": "DieselPowerPlant-3", "maxTurnProduction": 7500.0, "lastTurnProduction": 7500.0, "on": true }, // ...], "numberOfMessagesToReceive": 0, "distributionInstructions": [{ "nodesPath": ["Grid-3", "Grid-2"], "energyToDistribute": 1363.7003 }, // ...] }</pre>

Tabella 20: Struttura di un messaggio contenente le informazioni sulla distribuzione

Il messaggio contiene i seguenti parametri:

- **numberOfMessagesToReceive**: il numero di messaggi che la Grid deve aspettarsi di ricevere dalle Grid vicine.
- **distributionInstructions**: un array che contiene degli oggetti formati dall'attributo **energyToDistribute**, che indica l'energia che la Grid deve mandare al prossimo nodo sul **nodesPath**; il parametro **nodesPath** contiene il percorso, espresso come sequenza di nodi, che l'energia deve percorrere. Il nodo che riceverà il messaggio sarà il primo della lista dei **nodespath**, e dovrà inviare l'energia specificata al suo successore.
- **activeNonRenewablePowerPlants**, un array che contiene le informazioni aggiornate dei Power Plant non rinnovabili.

4.6.6 Load Manager - Non Renewable Power Plant

Il Load Manager deve anche comunicare ad ogni centrale elettrica di tipo non rinnovabile se deve accendersi per il prossimo turno e, in caso affermativo, quanta energia dovrebbe cercare di fornire al sistema.

Per fare ciò utilizza il seguente messaggio:

Campo	Valore
Sender	LoadManager-1
Receiver	NonRenewablePowerPlant-1
Conversation ID	-
Performative	INFORM
Content	{ "requiredEnergy": 298906.3630, "on": true }

Tabella 21: Struttura di un messaggio contenente le richieste di energia alle Non-Renewable Power Plant

Una volta ricevuto questo messaggio, la centrale diesel aggiornerà il suo stato interno di conseguenza, senza rispondere al messaggio del Load Manager.

4.6.7 Grid - Grid

La comunicazione intra-grid avviene tramite l'utilizzo della tecnica di orchestrazione. In questo caso, l'orchestratore è il Load Manager che, come spiegato in precedenza, fornisce a ogni Grid le istruzioni di cui ha bisogno per comunicare con i propri vicini.

In questa fase ciascuna Grid si occupa semplicemente di leggere le direttive inviate dal Load Manager; ogni direttiva contiene la quantità di energia da distribuire e il percorso sul grafo che deve percorrere. La Grid si occuperà di controllare la lunghezza del percorso; se tale lunghezza è pari a uno, allora quella Grid sarà il destinatario e di conseguenza riceverà la quantità di energia presente nel messaggio, comprensiva di perdite. Altrimenti, se la lista contiene almeno due elementi, procederà rimuovendo il primo elemento dalla lista e inoltrerà il messaggio al nodo successivo, aggiornando la quantità di energia calcolando la perdita che si crea durante la trasmissione. Ogni Grid conosce anche il numero di messaggi che deve ricevere e attende la ricezione di ogni singola comunicazione.

Campo	Valore
Sender	Grid-4
Receiver	Grid-3
Conversation ID	-
Performative	INFORM
Content	{ "distributionInstructions": { "nodesPath": ["Grid-3", "Grid-1"], "energyToDistribute": 2399.9488 } }

Tabella 22: Struttura di un messaggio di instradamento tra Grid

4.7 Classi Util

Per facilitare il lavoro degli agenti e per mantenere un ordine logico nella gestione del codice, sono state create delle classi di utility che offrono diverse funzionalità. Ogni classe espone dei metodi statici che possono essere chiamati da ciascun agente senza dover istanziare nuovi oggetti per utilizzare i metodi.

Prima di tutto è stata creata una classe chiamata `TimeUtil.java` che contiene tutta la logica riguardante la gestione del tempo. La classe si occupa di effettuare la conversione tra tempo effettivo e numero di turni quando necessario, migliorando di molto l'adattabilità del sistema alle durate variabili dei turni di simulazione.

Per quanto riguarda la gestione del meteo e delle condizioni atmosferiche del simulatore, è stata realizzata la classe `WeatherUtil.java`, che ha il compito, in fase di inizializzazione, di ricavare le informazioni sul meteo riguardanti la zona in cui avviene la simulazione e creare i modelli stocastici descritti in 3.1, per poi aggiornare il meteo durante l'esecuzione utilizzando i suddetti modelli.

La classe `JsonUtil.java` è utilizzata durante la fase di setup degli agenti. Questa classe espone un metodo che permette a ogni agente di istanziare correttamente l'oggetto del modello ad esso associato, contenuto in un file JSON.

È inoltre presente la classe `MessageUtil.json` che contiene una serie di variabili statiche spesso utilizzate come chiavi degli oggetti JSON trasmessi durante la comunicazione tramite messaggi; queste variabili risultano particolarmente utili per evitare problemi dovuti a errori di battitura o simili.

`EnergyUtil.java` è invece una classe di utilità utilizzata per tutto ciò che concerne le informazioni sui cavi e la loro gestione in fase di inizializzazione; contiene anche le basi logiche e le informazioni per il monitoraggio dei costi dell'elettricità in quanto ne ricava il prezzo medio in base alla nazione in cui avviene la simulazione. Tuttavia, è stata implementata solo una bozza della gestione dei prezzi in quanto non rappresenta un punto centrale per lo sviluppo del progetto, sebbene si presti bene per una futura espansione.

Infine, la classe `EnergyMonitorUtil.java` è stata creata per monitorare i consumi e le produzioni di energia; essa espone dei metodi statici che permettono agli agenti di salvare dei resoconti sui loro consumi e le loro produzioni di energia; esporta i dati che ha memorizzato a intervalli regolari a seconda del parametro di configurazione dell'esecuzione corrente. I dati esportati sono i seguenti:

Nome	Unità di misura
BatteryProduction	<i>W</i>
BuildingProduction	<i>W</i>
DieselProduction	<i>W</i>
HydroProduction	<i>W</i>
PhotovoltaicProduction	<i>W</i>
WindProduction	<i>W</i>
TotalDemand	<i>W</i>
WindSpeed	<i>km/h</i>
WeatherState	{0, 1, 2, 3}
StoredBattery	<i>W</i>

Tabella 23: Panoramica dei dati salvati durante la simulazione

Dove il valore salvato in `WeatherState` corrisponde al rispettivo stato della catena di Markov; in particolare, il numero rappresenta rispettivamente, in ordine crescente, gli stati Sunny, Cloudy, Rainy e Snowy.

4.8 Configurazione dell'Ambiente

Per far sì che il simulatore funzioni correttamente, è necessaria la presenza di una cartella nel percorso `scenarios/{scenario_name}/` contenente i seguenti file:

- `cables.json`.
- `dieselPowerPlants.json`.
- `grids.json`.
- `hydroPowerPlants.json`.
- `loadManagers.json`.
- `owners.json`.
- `photovoltaicPowerPlants.json`.
- `smartBuildings.json`.
- `windPowerPlants.json`.

Di seguito, viene spiegato come devono essere impostati i singoli file JSON.

4.8.1 Cables

In questo file deve essere presente un oggetto con due chiavi, ovvero `"types"` e `"links"`.

Il valore associato alla chiave `"types"` è un oggetto che contiene una serie di coppie chiave-valore; ogni chiave corrisponde al tipo di cavo e il valore è un oggetto contenente le sue caratteristiche, rappresentate da:

- **cableSection**: sezione del cavo in m^2 .
- **resistivity**: resistività del materiale scelto in $\Omega \cdot m$.
- **voltage**: voltaggio del cavo in V .

Un esempio di oggetto assegnato alla chiave `"types"` viene riportato di seguito:

```
"types": {
  "copperHigh2.5": {
    "cableSection": 2.5e-6,
    "resistivity": 1.68e-8,
    "voltage": 11000.0
  },
  "copperLow2.5": {
    "cableSection": 2.5e-6,
    "resistivity": 1.68e-8,
    "voltage": 230.0
  }
}
```

Listing 1: Struttura degli oggetti `types` contenuti in `cables.json`

Per quanto riguarda invece la chiave **"links"**, ad essa andrà associata una lista contenente degli oggetti, i quali saranno descritti dai seguenti attributi:

- **from**: il nome dell'agente da cui inizia il cavo.
- **to**: il nome dell'agente in cui finisce il cavo.
- **cableType**: il tipo di cavo, deve corrispondere con una chiave dell'oggetto assegnato a **"types"**.

Si noti che i cavi sono bidirezionali; di conseguenza, non è necessario duplicarli invertendo il campo **"from"** e il campo **"to"**.

Un esempio di lista assegnata alla chiave **"link"** viene riportato di seguito:

```
"links": [  
  {  
    "from": "SmartBuilding-1",  
    "to": "Grid-1",  
    "cableType": "copperLow2.5"  
  },  
  {  
    "from": "PhotovoltaicPowerPlant-1",  
    "to": "Grid-1",  
    "cableType": "copperHigh2.5"  
  }  
]
```

Listing 2: Struttura degli oggetti **links** contenuti in **cables.json**

Inoltre, è importante ricordare che i collegamenti possono esistere solamente tra queste coppie di agenti:

- Smart Building - Grid
- Grid - Power Plant
- Grid - Grid

Quindi, ad esempio, non è possibile collegare direttamente un edificio a una centrale elettrica.

4.8.2 DieselPowerPlants

Questo file contiene le informazioni sui Diesel Power Plant presenti nel sistema, rappresentati sotto forma di oggetti.

Ogni chiave presente nel file deve iniziare con la stringa **"DieselPowerPlant-"**, dopo il trattino è possibile inserire un nome o un numero a piacere per poter distinguere gli agenti. Ogni nome o numero identificativo deve essere univoco nella propria categoria.

Ad ogni chiave si deve associare un oggetto contenente i seguenti attributi:

- **gridName**: una stringa contenente il nome della Grid a cui l'impianto è connesso.
- **loadManagerName**: una stringa che rappresenta il nome del Load Manager del sistema.

- **enginePower**: un numero decimale contenente la potenza del motore espressa in W .
- **efficiency**: un numero compreso tra 0 ed 1 che rappresenta l'efficienza del sistema.
- **coordinates**: un oggetto contenente due chiavi: "latitude" e "longitude" a cui viene associato un numero reale che rappresenta rispettivamente la latitudine e la longitudine in gradi decimali.

Un esempio di file JSON viene riportato di seguito:

```
{
  "DieselPowerPlant-1": {
    "gridName": "Grid-2",
    "loadManagerName": "LoadManager-1",
    "enginePower": 2500000.0,
    "efficiency": 0.9,
    "coordinates": {
      "latitude": 2.82431943118895,
      "longitude": 104.16497639061139
    }
  }
}
```

Listing 3: Struttura di dieselPowerPlant.json

4.8.3 Grids

Questo file contiene le informazioni sulle Grid presenti nel sistema, rappresentate sotto forma di oggetti.

Anche in questo caso, ciascuna chiave dovrà iniziare con "Grid-" e dovrà essere univoca nel sistema.

L'oggetto associato a una chiave dovrà contenere i seguenti attributi:

- **smartBuildingNames**: un array di stringhe; indica i nomi degli edifici ad essa collegate.
- **loadManagerName**: una stringa che rappresenta il nome del Load Manager del sistema.
- **gridNames**: un array di stringhe; indica i nomi delle Grid ad essa collegate.
- **nonRenewablePowerPlantNames**: un array di stringhe; indica i nomi delle centrali non rinnovabili ad essa collegate.
- **renewablePowerPlantNames**: un array di stringhe; indica i nomi delle centrali rinnovabili ad essa collegate.
- **coordinates**: un oggetto contenente due chiavi: "latitude" e "longitude" a cui viene associato un numero reale che rappresenta rispettivamente la latitudine e la longitudine in gradi decimali.
- **battery**: un parametro facoltativo che contiene un oggetto che viene descritto in seguito.

Gli oggetti di tipo Battery devono contenere almeno 5 attributi:

- **voltage**: un numero che indica il voltaggio della batteria in Volt (V).
- **maxCapacityInWattHour**: l'energia massima che la batteria può contenere in Wh .
- **maxCapacityInAmpHour**: l'energia massima che la batteria può contenere in Ah .
- **storedEnergy**: l'energia attualmente contenuta nella batteria in Wh .
- **dischargeCurrent**: un numero che indica la corrente massima di scarica della batteria, espressa in Ampere (A).
- **efficiency**: un numero compreso tra 0 ed 1 che indica l'efficienza della batteria.
- **stateOfCharge**: un numero compreso tra 0 ed 1 che indica la percentuale di carica iniziale della batteria.

Basta che sia definito almeno un parametro tra "maxCapacityInWattHour" e "maxCapacityInAmpHour"; se entrambi i parametri vengono definiti, devono ovviamente essere coerenti; lo stesso vale per gli attributi "stateOfCharge" e storedEnergy.

Un oggetto di esempio viene riportato di seguito:

```
{
  "Grid-1":{
    "smartBuildingNames": ["SmartBuilding-1"],
    "loadManagerName": "LoadManager-1",
    "gridNames": ["Grid-2"],
    "nonRenewablePowerPlantNames": [ ],
    "renewablePowerPlantNames": ["PhotovoltaicPowerPlant-1",
      "HydroPowerPlant-1", "WindPowerPlant-1"],
    "coordinates": {
      "latitude": 2.823847438209252,
      "longitude": 104.16354506562476
    }
  }
}
```

Listing 4: Struttura di grids.json

4.8.4 HydroPowerPlants

Questo file contiene le informazioni sulle centrali idroelettriche presenti nel sistema, rappresentate sotto forma di oggetti.

Anche in questo caso, ogni chiave deve iniziare con la stringa "HydroPowerPlant-" e deve necessariamente essere univoca.

Ad ogni chiave bisogna associare un oggetto definito dai seguenti campi:

- **gridName**: una stringa contenente il nome della Grid a cui l'impianto è connesso.
- **efficiency**: un numero compreso tra 0 ed 1 che rappresenta l'efficienza del sistema.

- **flowRate**: un numero che indica il volume di acqua medio che passa attraverso la turbina, in m^3/s . Questo valore corrisponde alla portata, esso verrà moltiplicato per un numero aleatorio compreso tra $[0, 85, 1.15]$ per simulare un andamento non completamente costante della produzione.
- **headHeight**: un numero che rappresenta l'altezza del salto compiuto dal corso d'acqua per arrivare alla turbina, in m .
- **coordinates**: un oggetto contenente due chiavi: "latitude" e "longitude" a cui viene associato un numero reale che rappresenta rispettivamente la latitudine e la longitudine in gradi decimali.

Di seguito viene riportato un esempio del contenuto del file JSON:

```
{
  "HydroPowerPlant-1": {
    "gridName": "Grid-1",
    "efficiency": 0.7,
    "flowRate": 0.25,
    "headHeight": 141.0,
    "coordinates": {
      "latitude": 2.777288542973285,
      "longitude": 104.20477531556426
    }
  }
}
```

Listing 5: Struttura di `hydroPowerPlants.json`

4.8.5 LoadManagers

Questo file JSON può contenere un solo oggetto, dato che al momento non è possibile inserire più di un Load Manager nel simulatore.

La chiave deve iniziare con la stringa "LoadManager-" e il valore ad essa associato sarà un oggetto descritto dai seguenti attributi:

- **gridNames**: un array di stringhe che contiene tutte le Grid a cui il Load Manager è collegato a livello logico.
- **nonRenewablePowerPlantNames**: un array di stringhe che contiene tutti i nomi dei Power Plant non rinnovabili presenti nel sistema.

Viene riportato un esempio di contenuto del file JSON:

```
{
  "LoadManager-1": {
    "gridNames": ["Grid-1", "Grid-2", "Grid-3"],
    "nonRenewablePowerPlantNames": ["DieselPowerPlant-1"]
  }
}
```

Listing 6: Struttura di `loadManagers.json`

4.8.6 Owners

Ancora una volta, ogni chiave deve iniziare con la stringa "Owner-" e deve essere univoca. Ad ogni chiave viene associato un oggetto con un solo parametro: "smartBuildingNames" a cui viene associata una lista di stringhe che rappresentano gli edifici di proprietà dell'Owner.

```
{
  "Owner-1": {
    "smartBuildingNames": ["SmartBuilding-1"]
  }
}
```

Listing 7: Struttura di owners.json

4.8.7 PhotovoltaicPowerPlant

Anche in questo caso, ogni chiave deve iniziare con la stringa "PhotovoltaicPowerPlant-" e deve necessariamente essere univoca. L'oggetto associato a una chiave deve contenere le informazioni descritte nel dettaglio in 3.4, riportate di seguito:

- **gridName**: una stringa contenente il nome della Grid a cui l'impianto è connesso.
- **efficiency**: un valore di efficienza che cambia a seconda del tipo di pannello utilizzato.
- **area**: estensione totale in m^2 dei pannelli.
- **azimuthAngleArray**: angolo Azimut del sistema fotovoltaico in gradi.
- **tiltAngle**: angolo di inclinazione del pannello rispetto alla superficie, sempre in gradi.
- **albedo**: valore di albedo della superficie.
- **coordinates**: un oggetto contenente due chiavi: "latitude" e "longitude" a cui viene associato un numero reale che rappresenta rispettivamente la latitudine e la longitudine in gradi decimali.

```
{
  "PhotovoltaicPowerPlant-1": {
    "gridName": "Grid-1",
    "efficiency": 0.1,
    "area": 5000.0,
    "azimuthAngleArray": 0.0,
    "tiltAngle": 30.0,
    "albedo": 0.33,
    "coordinates": {
      "latitude": 2.8249031004478966,
      "longitude": 104.16163910828905
    }
  }
}
```

Listing 8: Struttura di photovoltaicPowerPlant.json

4.8.8 SmartBuildings

Per l'ennesima volta, ogni chiave deve iniziare con la stringa "SmartBuilding-" e deve necessariamente essere univoca.

Ad ogni chiave bisogna associare un oggetto con i seguenti attributi:

- **priority**: una stringa che può assumere solamente i valori "LOW", "MEDIUM", "HIGH" e che indica il valore di importanza della struttura.
- **appliances**: un parametro che contiene un array di oggetti di tipo Appliance che verranno descritti in seguito.
- **buildingPhotovoltaicSystem**: un oggetto che contiene gli stessi valori presenti in PhotovoltaicPowerPlant.
- **gridName**: una stringa contenente il nome della Grid a cui l'agente è connesso.
- **battery**: un parametro facoltativo che contiene una batteria analoga a quella presente in Grids.
- **routine**: un parametro che contiene un array di oggetti di tipo Task che vengono descritti in seguito.
- **coordinates**: un oggetto contenente due chiavi: "latitude" e "longitude" a cui viene associato un numero reale che rappresenta rispettivamente la latitudine e la longitudine in gradi decimali.

Gli oggetti di tipo Appliance devono avere 4 attributi:

- **name**: una stringa che indica il nome del dispositivo.
- **alwaysOn**: un booleano che indica se il dispositivo è sempre acceso o meno.
- **on**: un booleano che indica se il dispositivo è acceso in questo momento.
- **hourlyConsumption**: un numero che indica il consumo orario dell'elettrodomestico in *Wh*.

Gli oggetti di tipo Task devono contenere 3 attributi:

- **applianceName**: una stringa che contiene il nome di un elettrodomestico, questo nome deve corrispondere necessariamente al nome di un oggetto contenuto nell'array **appliances**, descritto in precedenza.
- **startTime**: una stringa nel formato **hh:mm** che indica l'orario in cui l'elettrodomestico deve accendersi.
- **endTime**: una stringa contenente un orario nel formato **hh:mm** che indica quando l'elettrodomestico si deve spegnere.

```

{
  "SmartBuilding-1": {
    "priority": "LOW",
    "appliances": [
      {
        "name": "Television-1",
        "alwaysOn": false,
        "on": false,
        "hourlyConsumption": 200.0
      }
    ],
    "buildingPhotovoltaicSystem": {
      "efficiency": 0.1,
      "area": 40.0,
      "coordinates": {
        "latitude": 2.8250926770202165,
        "longitude": 104.16364967701783
      },
      "azimuthAngleArray": 0.0,
      "albedo": 0.33,
      "tiltAngle": 30.0
    },
    "gridName": "Grid-1",
    "battery": {
      "voltage": 25.6,
      "maxCapacityInWattHour": 5120,
      "storedEnergy": 0.0,
      "dischargeCurrent": 200.0,
      "efficiency": 0.9,
      "stateOfCharge": 0.0
    },
    "routine": {
      "tasks": [
        {
          "applianceName": "Television-1",
          "startTime": "00:30",
          "endTime": "02:00"
        }
      ]
    },
    "coordinates": {
      "latitude": 2.8250926770207165,
      "longitude": 104.16364967701783
    }
  }
}

```

Listing 9: Struttura di smartBuildings.json

4.8.9 WindPowerPlants

Per l'ultima volta, ogni chiave deve iniziare con la stringa "WindPowerPlant-" e deve necessariamente essere univoca.

Ogni oggetto di tipo WindPowerPlant deve contenere le seguenti informazioni:

- **gridName**: una stringa contenente il nome della Grid a cui l'impianto è connesso.
- **minWindSpeed**: minima velocità operativa per poter produrre energia, in m/s .
- **maxWindSpeed**: massima velocità operativa per poter produrre energia, in m/s .
- **rotorDiameter**: lunghezza del diametro del rotore in m .
- **numberOfTurbines**: numero di turbine presenti nell'impianto.
- **coordinates**: un oggetto contenente due chiavi: "latitude" e "longitude" a cui viene associato un numero reale che rappresenta rispettivamente la latitudine e la longitudine in gradi decimali.

```
{
  "WindPowerPlant-1": {
    "gridName": "Grid-1",
    "minWindSpeed": 2.5,
    "maxWindSpeed": 11.0,
    "rotorDiameter": 120.0,
    "numberOfTurbines": 2,
    "coordinates": {
      "latitude": 2.8078769856987176,
      "longitude": 104.1325987382965
    }
  }
}
```

Listing 10: Struttura di windPowerPlant.json

5 Scenario di Simulazione

Per testare il simulatore è stato necessario riprodurre uno scenario che contenesse gli elementi descritti in precedenza, in modo tale da poter avere un punto di riferimento e confronto per valutarne la correttezza.

Tra gli studi rilevanti per questo argomento è stato individuato l'articolo [12], che contiene un caso di studio adatto al contesto di sviluppo.

5.1 L'isola di Tioman

Nell'articolo viene presentato come scenario di riferimento l'isola di Tioman, chiamata anche Pulau Tioman. Questa piccola isola della Malesia, situata nel Mar Cinese Meridionale, è sempre stata fortemente dipendente dai combustibili fossili per la produzione di energia elettrica, a causa della sua posizione geografica isolata rispetto alle penisole principali. Tuttavia, negli ultimi anni, sono state intraprese iniziative per l'installazione di fonti rinnovabili di energia elettrica, per promuovere la produzione di energia pulita e per ammortizzare costi e consumi derivati dal generatore principale dell'isola, che opera tramite la combustione del diesel.

Al momento, sull'isola sono installati:

- Un generatore a Diesel, che al momento è la fonte principale di energia elettrica.
- Un impianto fotovoltaico.
- Una centrale idroelettrica.

Inoltre, si sta valutando la possibile installazione di un parco eolico offshore nella costa occidentale dell'isola; per questo motivo sono stati presi in considerazione i dati di alcuni studi per valutare l'impatto della loro installazione.

In seguito viene riportata una topografia dell'isola con la relativa legenda per mostrare la distribuzione sul territorio delle varie entità.







Icona	Significato
	Power Plant Diesel
	Power Plant Eolico
	Power Plant Idroelettrico
	Power Plant Fotovoltaico
	Grid Elettrica
	Centro Abitato / Smart Building

Tabella 24: Elementi della rete elettrica

Sono state analizzate le informazioni dei principali generatori di energia e dei villaggi più importanti dell'isola.

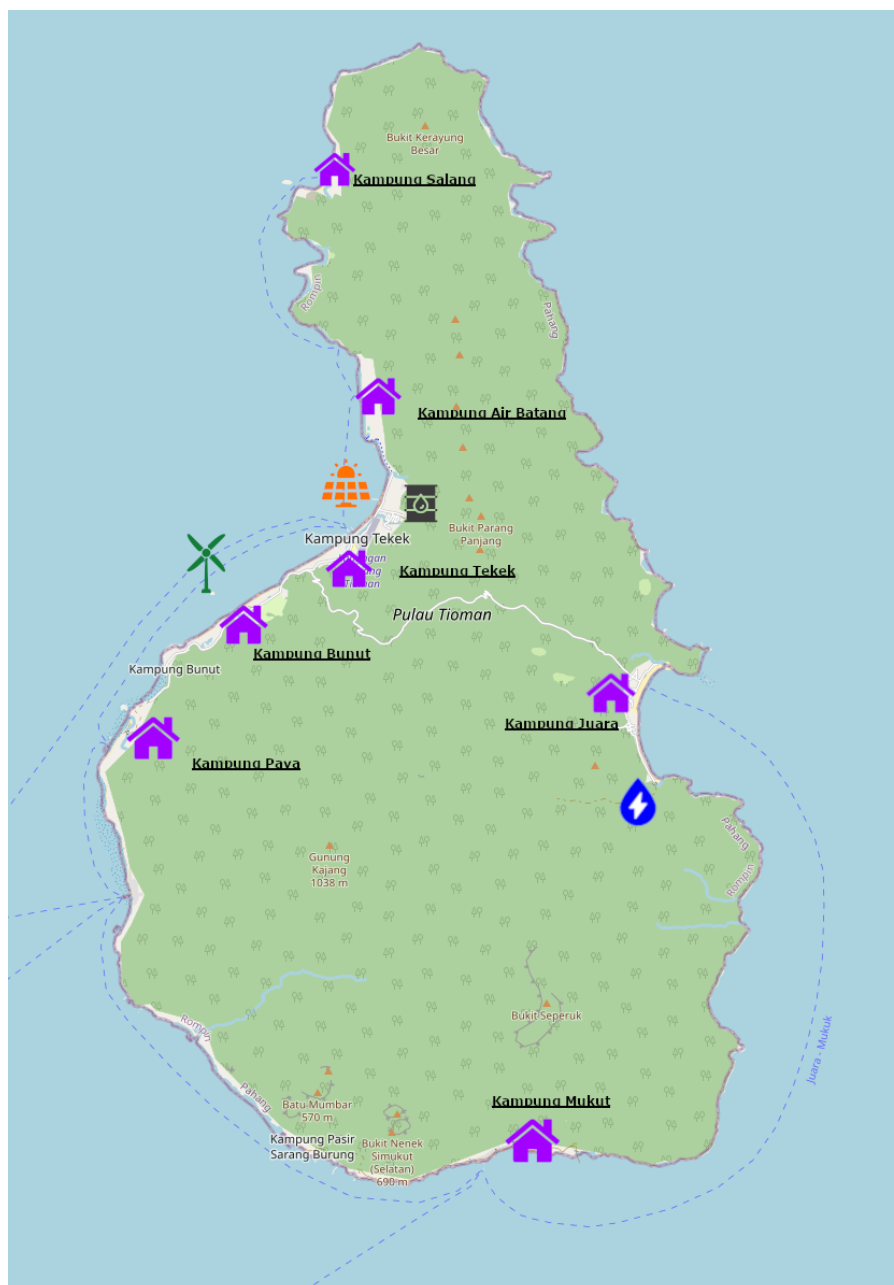


Figura 9: Topografia dell'isola di Tioman

Di seguito, viene riportata una visione nel dettaglio dei villaggi principali, chiamati kampung nella lingua locale, semplificati ai fini della simulazione.

5.1.1 Kampung Tekek

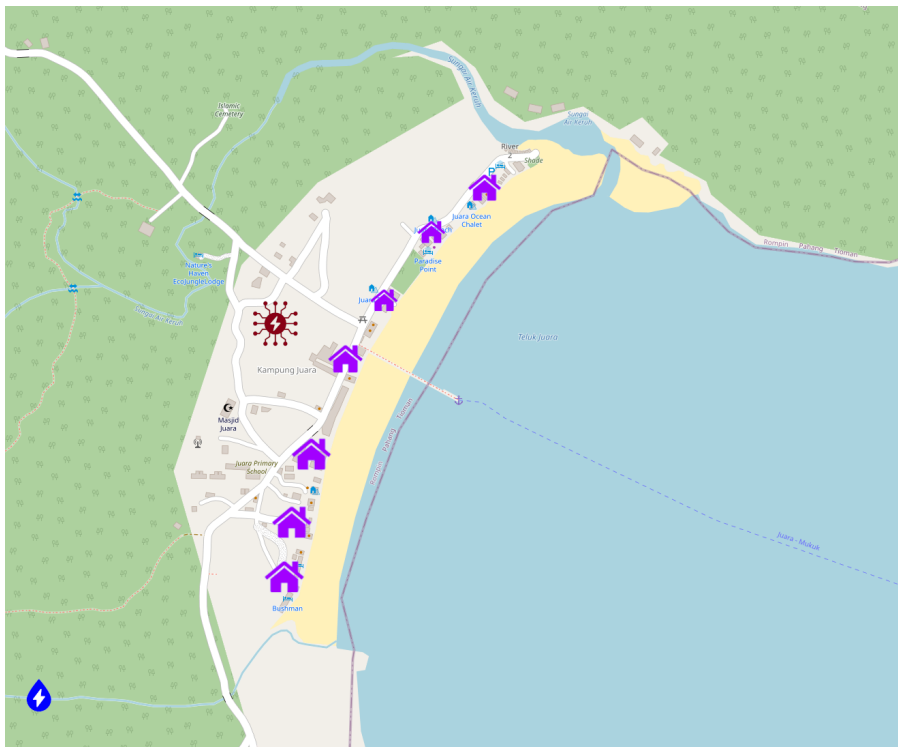
Kampung Tekek è uno dei due principali villaggi dell'isola, contiene un aeroporto, un ospedale e diverse abitazioni. Dal punto di vista energetico, in questo villaggio sono presenti sia l'impianto fotovoltaico che il generatore diesel, che lo rendono il villaggio più importante per la produzione elettrica dell'isola.

Possiede inoltre un impianto di accumulo utile per immagazzinare l'energia in eccesso, per poi distribuirla in caso di necessità.



5.1.2 Kampung Juara

Kampung Juara è il secondo villaggio in ordine di importanza, contiene diverse attività e abitazioni; nei pressi di questo villaggio è presente la centrale idroelettrica dell'isola, che sfrutta un corso d'acqua che la attraversa.



5.1.3 Kampung Salang

Kampung Salang è un piccolo villaggio situato nella parte settentrionale dell'isola ed è formato da poche abitazioni.

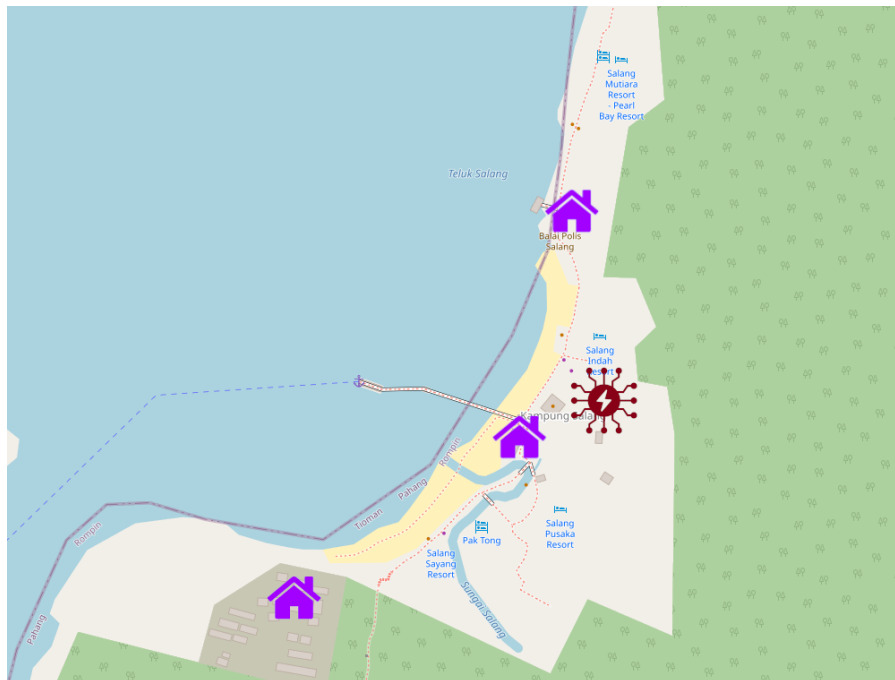


Figura 12: Dettaglio su Kampung Salang

5.1.4 Kampung Paya

Kampung Paya è un villaggio situato nella parte occidentale dell'isola, contiene poche abitazioni e qualche piccolo negozio.

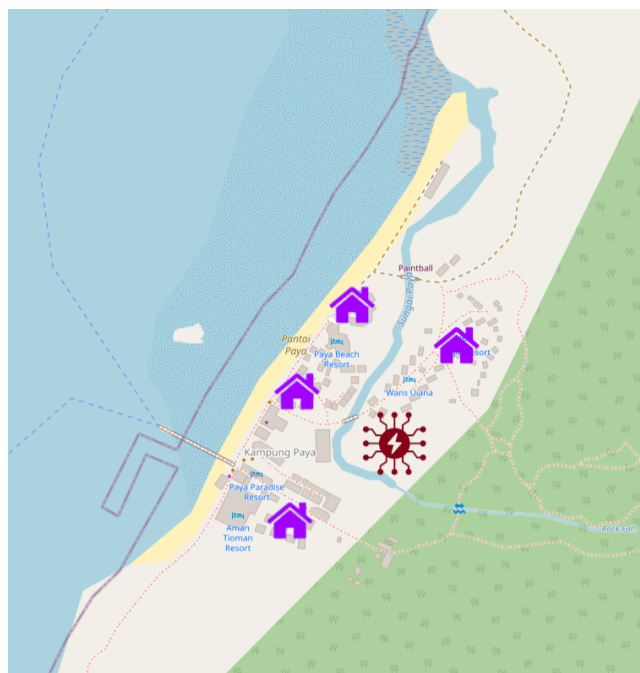


Figura 13: Dettaglio su Kampung Paya

5.1.5 Kampung Bunut

Kampung Bunut è un piccolo villaggio situato nella parte settentrionale dell'isola, contiene delle abitazioni, un campo da golf e alcune attività. Nei pressi di questo villaggio si è valutata l'installazione di un parco eolico offshore.

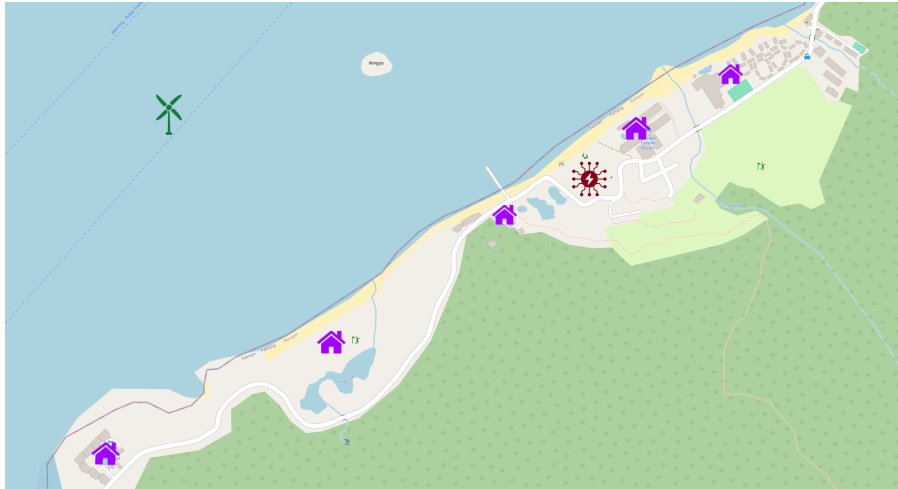


Figura 14: Dettaglio su Kampung Bunut

5.1.6 Kampung Air Batang

Kampung Air Batang è un villaggio situato nella parte centro-settentrionale dell'isola, formato da un esiguo numero di abitazioni ed esercizi commerciali.

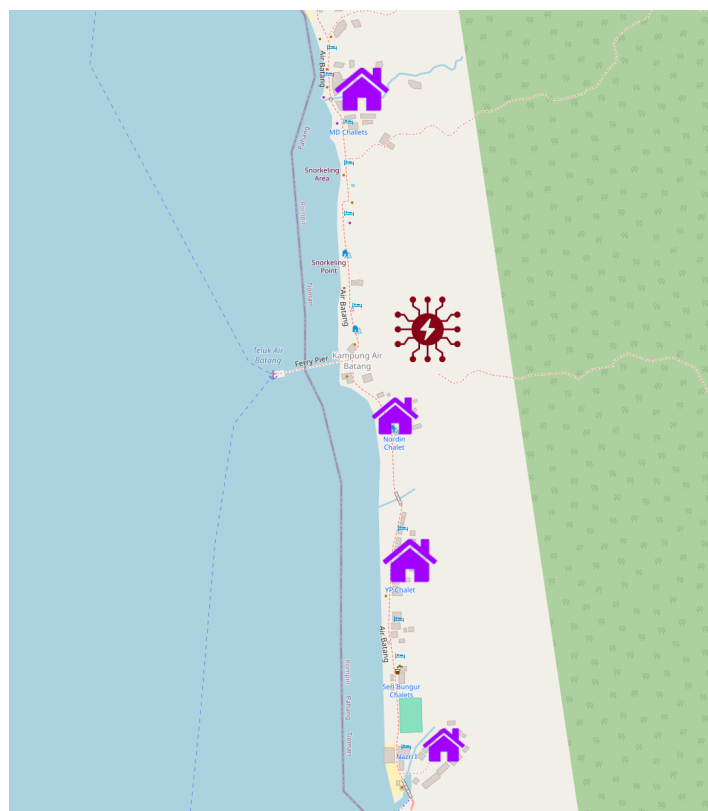


Figura 15: Dettaglio su Kampung Air Batang

5.1.7 Kampung Mukut

Kampung Mukut è un piccolo villaggio situato nella parte meridionale dell'isola, contiene poche abitazioni e un paio di imprese locali.

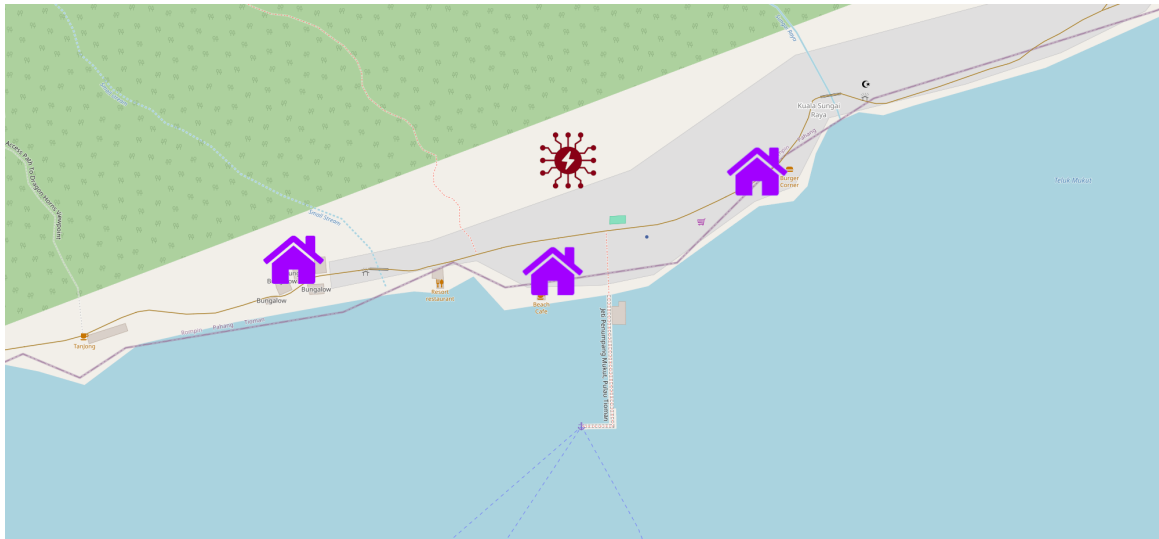


Figura 16: Dettaglio su Kampung Mukut

5.2 Configurazione dello Scenario

Per simulare lo scenario dell'isola di Tioman, sono state raccolte le informazioni relative agli impianti di produzione di energia e alle abitudini di consumo elettrico del territorio.

5.2.1 Impianti di Produzione Elettrica

Innanzitutto, per quanto riguarda le centrali non rinnovabili, è stato modellato un impianto che presenta le seguenti caratteristiche:

Parametro	Valore
Potenza Motore	2,500,000
Efficienza	0.90

Tabella 25: Caratteristiche dell’impianto diesel modellato

Per quanto concerne invece l'impianto idrico, esso è stato modellato utilizzando le seguenti informazioni:

Parametro	Valore
Efficienza	0.70
Portata	0.25
Altezza del salto	141.0

Tabella 26: Caratteristiche della centrale idroelettrica modellata

Successivamente è stato modellato l'impianto fotovoltaico con le seguenti impostazioni:

Parametro	Valore
Efficienza	0.10
Area	5000
Angolo di azimut dei pannelli	0.0
Angolo di inclinazione dei pannelli	30.0
Albedo	0.33

Tabella 27: Caratteristiche dell'impianto fotovoltaico modellato

In seguito, per modellare il Power Plant eolico sono stati utilizzati i seguenti parametri:

Parametro	Valore
Velocità minima operativa	2.5
Velocità massima operativa	11.0
Diametro rotore	70
Numero di turbine	2

Tabella 28: Caratteristiche della centrale eolica modellata

Infine, i cavi presenti sull'isola sono stati modellati utilizzando i seguenti valori:

Tipo di cavo	Sezione del cavo	Resistività	Voltaggio
Alta tensione	$2.5 \cdot 10^{-6}$	$1.68 \cdot 10^{-8}$	33000
Media Tensione	$2.5 \cdot 10^{-6}$	$1.68 \cdot 10^{-8}$	15000

Tabella 29: Caratteristiche della centrale eolica modellata

Sono stati impiegati cavi ad alta tensione per connettere le centrali elettriche alle Grid e per collegare tra loro le diverse Grid.

Invece, per quanto riguarda la distribuzione di energia tra Grid e agglomerati di edifici, sono stati utilizzati cavi a media tensione.

Se fossero state modellate le singole abitazioni, sarebbe stato necessario utilizzare i cavi a bassa tensione.

5.2.2 Modellazione dei Consumatori

Siccome era irrealizzabile la modellazione casa per casa di tutta l'isola, sia per mancanza delle singole informazioni, sia per l'elevato numero di edifici da inserire manualmente nel sistema, sono stati modellati i consumatori come agglomerati di edifici per simulare la domanda energetica dell'isola nell'arco di una giornata nel periodo non monsonico.

I consumi utilizzati durante la simulazione sono stati ricavati dallo studio [12].

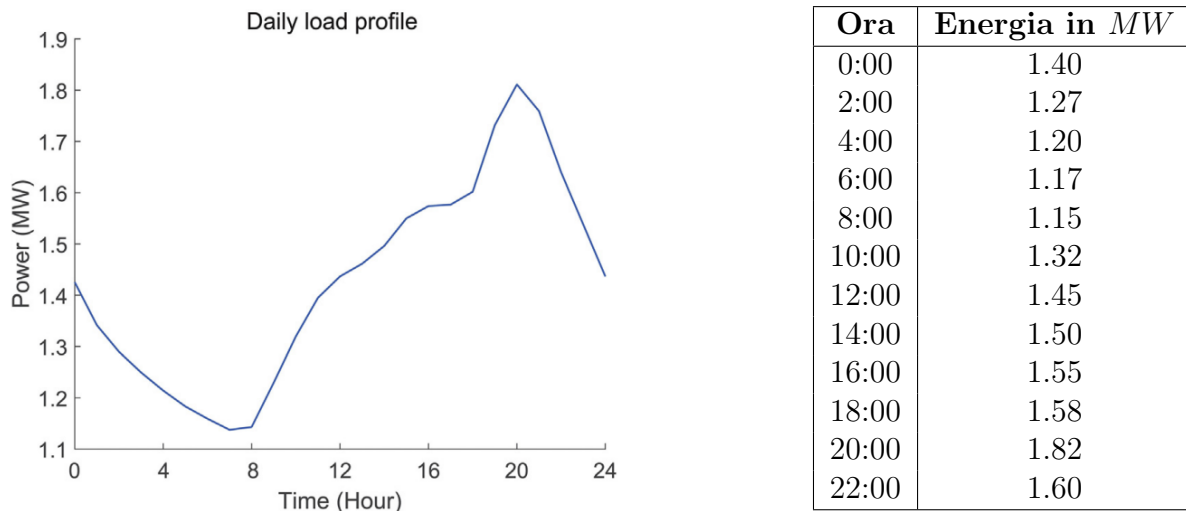


Figura 17: Riferimento al carico medio dei consumi nell’arco di una giornata

Di seguito viene riportata la domanda energetica simulata nello scenario modellato:

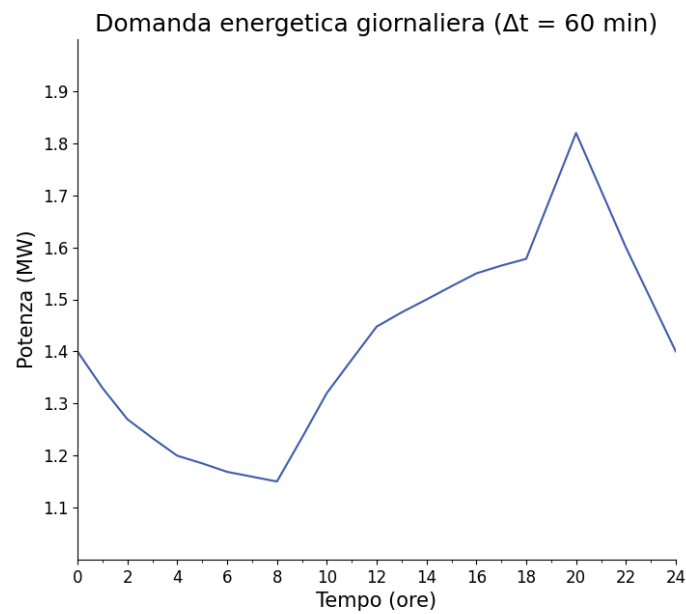


Figura 18: Carico medio dei consumi nell’arco della giornata simulata

5.2.3 Catene di Markov

Di seguito vengono riportate le Markov Chain ricavate a partire dai dati ottenuti tramite le API, come descritto in 3.1.

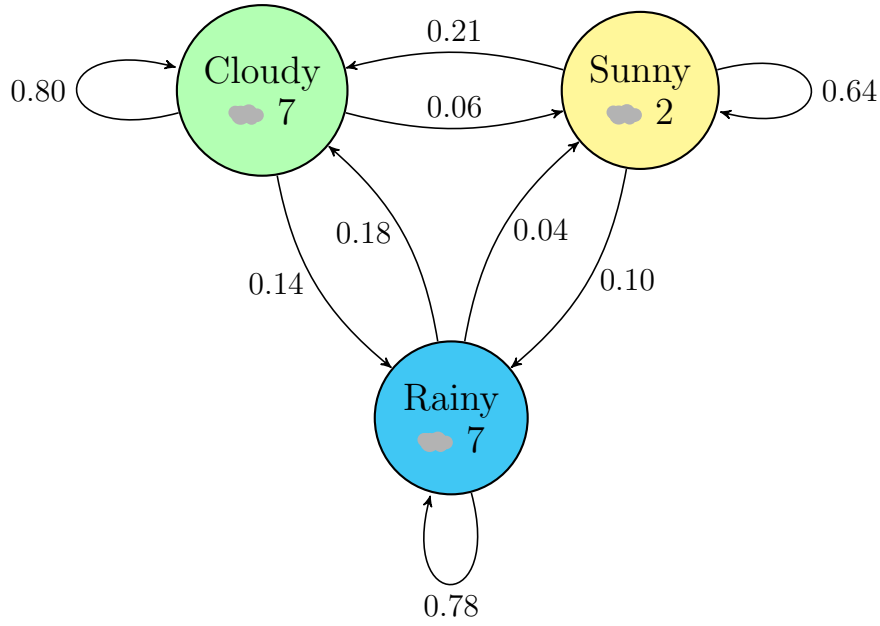


Figura 19: Markov Chain per la simulazione del meteo e relativa copertura nuvolare

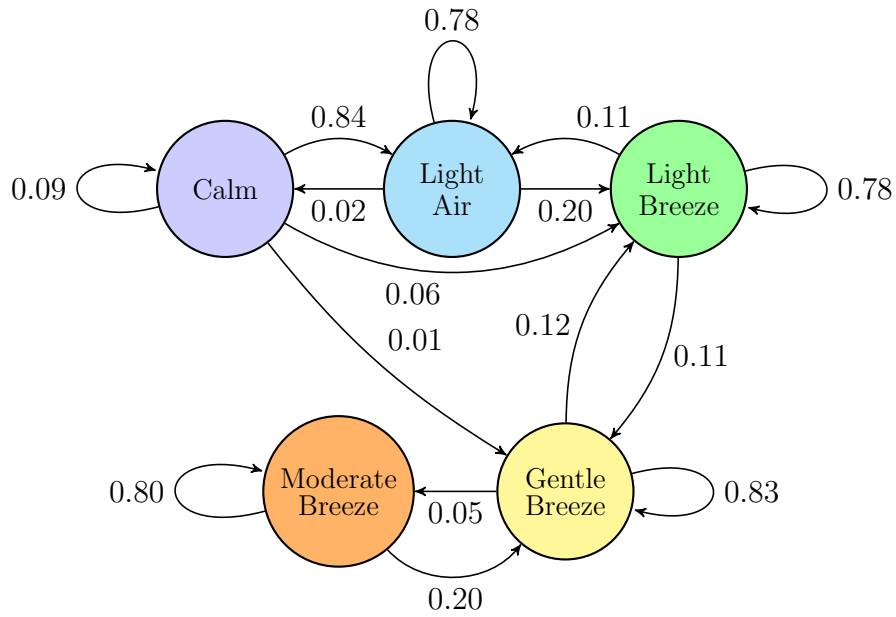


Figura 20: Markov Chain per la simulazione dell'andamento della velocità del vento

5.2.4 Configurazione del Simulatore

Per quanto riguarda la configurazione del simulatore, sono state utilizzate le seguenti impostazioni:

Parametro	Valore
turn_duration	00:30
interval_between_turns	100
save_interval	24:00
simulation_start_date	2025-04-01
latitude	2.7925619
longitude	104.1694260
weather_turn_duration	01:00
scenario_name	Tioman
price_volatility	0.05
price_trend	0.0002
random_seed	42

Tabella 30: Parametri con cui è stata eseguita la simulazione sullo scenario di Tioman

È stata scelta una `turn_duration` abbastanza bassa per provare a simulare cosa accade ogni mezz'ora, in modo tale da poter produrre dei grafici quanto più accurati possibile. Inoltre, è stato scelto un valore pari a 24 ore per il parametro `save_interval` in modo tale da poter esportare i dati sui consumi e le produzioni giornaliere, con lo scopo di confrontarli con i valori presenti nell'articolo di riferimento.

5.3 Risultati Ottenuti

In questa sezione vengono riportati i grafici prodotti durante la simulazione dello scenario appena presentato, i quali sono confrontati con quelli ricavati dall'articolo di riferimento. In particolare, sono state misurate le produzioni, i consumi e le variazioni delle condizioni meteorologiche.

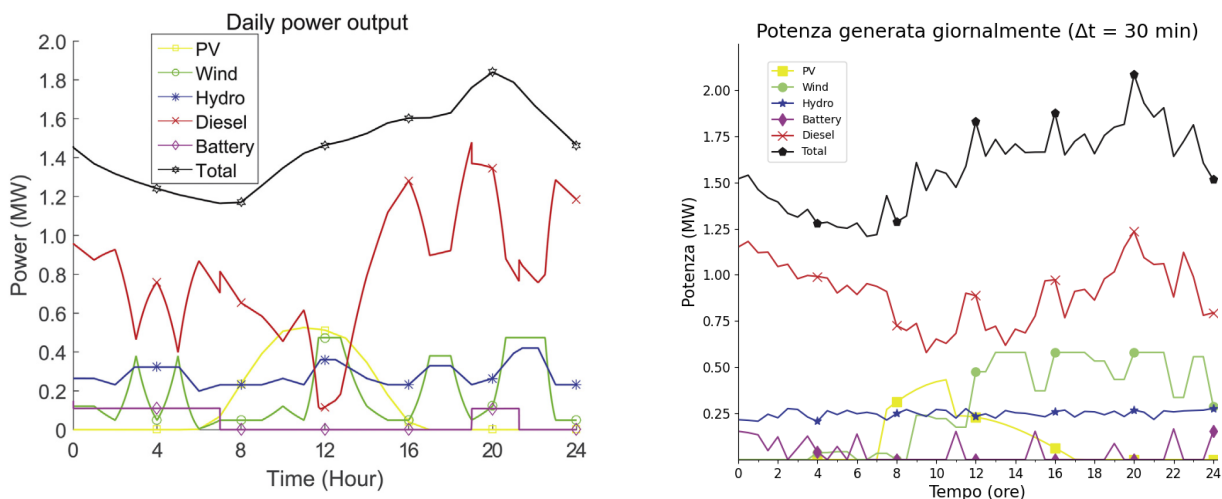


Figura 21: Confronto della produzione di energia tra lo studio di riferimento e il simulatore.

A sinistra, il grafico prodotto nell'articolo di riferimento, a destra, il grafico prodotto dal simulatore sviluppato in questo progetto

Come si può notare dal confronto tra i grafici mostrati nella figura 21, le fonti di energia rinnovabile sono state modellate in maniera abbastanza verosimile; ad esempio, si nota che la centrale fotovoltaica raggiunge il picco della propria produzione verso le 12, producendo una potenza istantanea di circa $500kW$.

La curva della produzione di energia idroelettrica risulta molto simile a quella mostrata nell'articolo, con un andamento che si assesta regolarmente attorno ai valori di produzione medi rispetto alla portata indicata per il corso d'acqua che attraversa l'isola.

Per quanto riguarda invece l'energia eolica, si può notare che anch'essa assume valori simili al grafico di riferimento con dei punti di massimo di circa $500kW$.

Di seguito vengono riportate le variazioni del meteo e della velocità del vento misurate durante la giornata simulata.

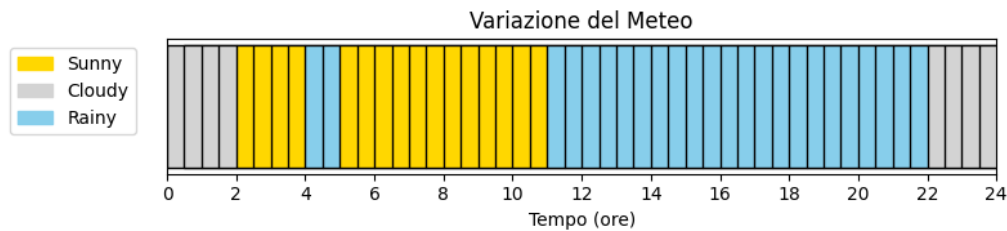


Figura 22: Variazione del meteo nell'arco della giornata simulata

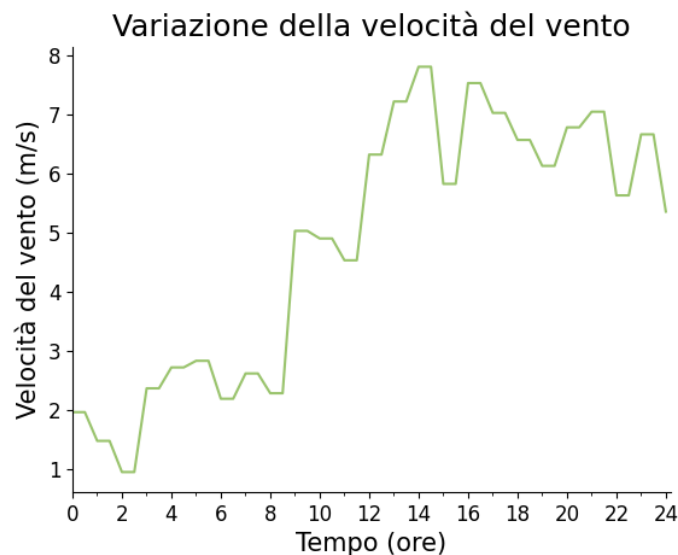


Figura 23: Variazione della velocità del vento nell'arco della giornata simulata

Come si può notare dalla figura 22, verso le 12:00 l'ambiente ha simulato un cambiamento meteorologico rilevante dove, passando da uno stato soleggiato a uno stato piovoso, la produzione di energia fotovoltaica è diminuita considerevolmente.

Contemporaneamente, anche il vento ha subito dei cambiamenti importanti, passando da una

velocità di circa $3m/s$ nelle ore mattutine a una velocità che si aggira tra i 7 e gli 8 m/s , portando a un incremento notevole della produzione di energia eolica.

Anche l'utilizzo della batteria avviene in fasce di orario coerenti con la natura del sistema; si nota infatti che essa tende a rilasciare energia più frequentemente durante la notte.

Inoltre, per quanto riguarda la produzione di energia non rinnovabile si può notare che, in entrambi i grafici, essa segue l'andamento della richiesta e della produzione di energia rinnovabile. Infatti, nel grafico di destra si vede che la produzione della centrale a diesel è abbastanza costante siccome le fonti di energia rinnovabile sono relativamente stabili (ad esempio, quando cala la produzione solare, aumenta quella eolica).

Tuttavia, quando la produzione rinnovabile aumenta, quella non rinnovabile diminuisce di conseguenza, alleggerendo il carico del generatore a diesel.

Infine, si può notare che la produzione totale di energia è molto simile tra i due grafici e segue la richiesta totale. Questa è un'ulteriore conferma del realismo del simulatore sviluppato ed inoltre dimostra che la coordinazione e la gestione dei singoli agenti sono efficaci.

Ovviamente è impossibile avere una corrispondenza perfetta tra i dati ottenuti dai due studi, visto che il simulatore sviluppato dipende fortemente dalle condizioni meteo, le quali vengono simulate in maniera aleatoria.

6 Conclusioni

6.1 Considerazioni Finali

Il progetto ha presentato un approccio distribuito multiagente in un ambito dove ha sempre regnato la centralizzazione, presentando una soluzione valida, resistente ai guasti e scalabile come alternativa ai metodi tradizionali.

Nonostante questo tipo di approccio sia più complesso da realizzare rispetto a quelli classici, esso si sposa molto bene con la natura decentralizzata del problema. Infatti, la programmazione multiagente richiede che le singole entità abbiano un certo grado di autonomia e sappiano comunicare attraverso un protocollo studiato ad-hoc, ma rende possibile il raggiungimento di ottimi risultati e può essere considerata una prospettiva interessante per il futuro.

JADE si è prestato bene allo sviluppo del simulatore e, sebbene la soluzione proposta sia ben lontana dalla perfezione, si ritiene essere un buon punto di partenza per ricerche future.

6.2 Sviluppi Futuri

Le opzioni di espansione del simulatore sono molteplici, di seguito ne vengono riportate alcune.

Un possibile sviluppo futuro consiste nell'implementazione di un sistema di tracciamento dei consumi di carburante delle centrali a diesel. Si potrebbe infatti, a partire dalla produzione elettrica di un turno, quantificare i consumi di diesel e tenerne traccia per calcolarne i costi e le emissioni.

Secondariamente, sempre per quanto riguarda il tracciamento dei costi, è già presente una prima implementazione per la simulazione della variazione dei costi dell'energia elettrica. Sarebbe interessante valutare un'implementazione che calcola i costi anche in base all'impianto da cui proviene tale energia.

Inoltre, si potrebbe provare ad estendere l'agente Smart Building per fare in modo che utilizzi dei modelli predittivi che valutino le proprie abitudini di consumo e l'andamento dei prezzi dell'energia elettrica. In questo modo, sarebbe possibile capire qual è il momento più opportuno per accendere i propri elettrodomestici in base al costo dell'energia, risparmiando e alleggerendo il carico complessivo della rete elettrica.

Un'altra possibile estensione del simulatore potrebbe consistere nell'aggiunta di altri agenti, come ad esempio le auto elettriche; l'idea potrebbe essere quella di fare in modo che le auto possano spostarsi nell'ambiente, connettersi agli agenti esistenti per ricaricarsi o fornire energia alle Grid in caso di necessità.

Infine, un'ulteriore modifica potrebbe consistere nell'inserimento nel sistema di Load Manager multipli. Infatti, ad oggi è possibile specificare un solo Load Manager ma, con questa estensione, si potrebbe rendere l'algoritmo di distribuzione più efficace per grandi reti di distribuzione; ciò richiederebbe tuttavia un'ulteriore fase nel protocollo di comunicazione dove i Load Manager devono accordarsi per organizzare una distribuzione ottimale dell'energia.

Bibliografia

- [1] G. H. Merabet, M. Essaaïdi, H. Talei, M. R. Abid, N. Khalil, M. Madkour e D. Benhaddou, «Applications of Multi-Agent Systems in Smart Grids: A survey,» in *2014 International Conference on Multimedia Computing and Systems (ICMCS)*, 2014, pp. 1088–1094. DOI: 10.1109/ICMCS.2014.6911384 (cit. a p. 2).
- [2] S. Karnouskos e T. N. d. Holanda, «Simulation of a Smart Grid City with Software Agents,» in *2009 Third UKSim European Symposium on Computer Modeling and Simulation*, 2009, pp. 424–429. DOI: 10.1109/EMS.2009.53 (cit. alle pp. 2, 16).
- [3] Telecom Italia Lab, *JADE*, 2025. indirizzo: <https://jade.tilab.com/> (cit. a p. 2).
- [4] Foundation for Intelligent Physical Agents, *FIPA Standards and Specifications*, 2023. indirizzo: <http://www.fipa.org> (cit. a p. 3).
- [5] P. Schavemaker e L. van der Sluis, «Electrical Power System Essentials,» in 2nd. Wiley, 2008 (cit. alle pp. 8, 9).
- [6] Sandia National Laboratories, *PV Performance Modeling Collaborative (PVPMC)*. indirizzo: <https://pvpmc.sandia.gov/modeling-guide/1-weather-design-inputs/> (cit. alle pp. 10–13).
- [7] J. S. Stein, C. W. Hansen e M. J. Reno, «Global Horizontal Irradiance Clear Sky Models: Implementation and Analysis,» Sandia National Laboratories, rapp. tecn. SAND2012–2389, mar. 2012. DOI: 10.2172/1039404. indirizzo: <https://www.osti.gov/servlets/purl/1039404> (cit. alle pp. 11, 12).
- [8] D. R. Myers, «Solar Radiation: Practical Modeling for Renewable Energy Applications,» in 1st. CRC Press, 2013. DOI: 10.1201/b13898. indirizzo: <https://doi.org/10.1201/b13898> (cit. alle pp. 11, 12).
- [9] B. Marion, «Model for Deriving the Direct Normal and Diffuse Horizontal Irradiance from the Global Tilted Irradiance,» *Solar Energy*, vol. 122, nov. 2015. DOI: 10.1016/j.solener.2015.10.024. indirizzo: <https://www.osti.gov/servlets/purl/1227196> (cit. a p. 12).
- [10] T. Burton, D. Sharpe, N. Jenkins e E. Bossanyi, *Wind Energy Handbook*. Chichester, UK: John Wiley & Sons, Ltd, 2001, ISBN: 0-471-48997-2. indirizzo: https://library.uniteddiversity.coop/Energy/Wind/Wind_Energy_Handbook.pdf (cit. a p. 13).
- [11] J. Twidell e T. Weir, «Renewable Energy Resources,» in 3rd. Routledge, 2015, pp. 208–212. DOI: 10.4324/9781315766416. indirizzo: <https://doi.org/10.4324/9781315766416> (cit. a p. 14).
- [12] M. R. Basir Khan, R. Jidin e J. Pasupuleti, «Multi-agent based distributed control architecture for microgrid energy management and optimization,» *Energy Conversion and Management*, vol. 112, pp. 288–307, 2016, ISSN: 0196-8904. DOI: <https://doi.org/10.1016/j.enconman.2016.01.011>. indirizzo: <https://www.sciencedirect.com/science/article/pii/S0196890416000273> (cit. alle pp. 16, 41, 47).