

BUSINESS PROCESS MODELING

P43: MOSTRA

3 novembre 2023

Andrea Cardia

Università di Pisa

Indice

1	Introduzione	1
2	Diagrammi BPMN	1
2.1	Stile di progettazione	1
2.2	Scelta del software	2
2.3	Scenario 1	2
2.4	Scenario 2	4
2.5	Variante dello scenario 2	5
3	Reti di Petri	5
3.1	Nozioni fondamentali sulle reti di Petri	5
3.2	Workflow Nets	6
3.3	Traduzione dei diagrammi in reti di Petri	7
4	Analisi delle reti	7
5	Conclusioni	8
Appendice A: Diagramma scenario 1		9
Appendice B: Diagramma scenario 2		9
Appendice C: Diagramma variante scenario 2		9

1 Introduzione

In questo documento si progetteranno dei diagrammi BPMN (Business Processes Modeling Notation) per descrivere il processo attraverso il quale una maestra si coordina con la biglietteria di una mostra e con la biglietteria di un'agenzia di noleggio autobus per organizzare una gita per la sua classe. I diagrammi saranno due: nel primo scenario la maestra contatta la biglietteria della mostra e la biglietteria degli autobus per richiedere informazioni, dopodichè decide se annullare la gita o se prenotare i biglietti; in fase di prenotazione la maestra deve opportunamente trovare un accordo tra la data dell'esposizione e la data di partenza (e quella di ritorno) degli autobus per garantire la disponibilità di posti e del mezzo di trasporto; fino a una settimana prima della visita è possibile modificare il numero di biglietti o la data un numero impreciso di volte, fino a quando la visita viene effettuata. Nel secondo scenario la gita dovrà essere approvata da parte del consiglio di classe. In aggiunta a questi due scenari verrà proposta una semplice variante dello scenario 2 in cui, a seguito della disapprovazione dell'itinerario proposto da parte del consiglio di classe, la maestra può decidere se proporre una nuova gita (sempre contattando le stesse biglietterie) o se desistere e annullarla definitivamente. I processi progettati con la Business Process Modeling Notation verranno opportunamente tradotti in reti di Petri per verificarne la bontà di progettazione; quest'ultima sarà valutata in termini di una metrica chiamata soundness che verrà spiegata nel Capitolo 3.2. Verranno infine effettuate delle analisi sulle reti di Petri derivate dai diagrammi BPMN sulla base della teoria sottostante a questi oggetti matematici.¹

2 Diagrammi BPMN

La progettazione di un diagramma BPMN prevede dapprima l'identificazione degli attori coinvolti nel processo; nel nostro caso è possibile identificare in questi attori la maestra, la biglietteria degli autobus, e la biglietteria della mostra. Per ciascun attore si riserva uno spazio rettangolare nel diagramma chiamato pool al cui interno viene svolto il processo del singolo attore. Le pools rappresentanti i diversi attori sono dotate di un unico inizio e di un'unica fine, e, mentre gli elementi di cui sono composte (gli eventi e le attività) sono connessi tra di loro da archi continui, gli elementi di pools diverse sono connessi con archi tratteggiati di modo da realizzare un meccanismo di collaborazione anche detto "coreografia".

2.1 Stile di progettazione

Nel progettare i diagrammi si sono utilizzate un paio di prescrizioni di stile:

- Vista la forte intercambiabilità tra eventi e attività, si è scelto di utilizzare principalmente queste ultime per rappresentare, sia le situazioni prive di marcatori², sia quelle concernenti l'invio e la ricezione di messaggi, mentre si è riservato agli eventi ruoli particolarmente speciali come il vincolo di attesa temporale o il vincolo di cattura di un segnale dall'esterno.
- Per questioni di leggibilità ma soprattutto di coerenza logica, si sono incolonnati allo stesso livello i connettori logici delle diverse pools che trovavano corrispondenza reciproca; quindi, per esempio, laddove era presente una decisione da prendere da parte della maestra, e dunque vi era presente uno XOR SPLIT, ed in corrispondenza di tale decisione le biglietterie erano in attesa (dunque era presente un Event-based Exclusive Gateway - EBG da qui in avanti), si sono posti i connettori logici delle due pools allo stesso livello.
- Sempre per migliorare la leggibilità dei diagrammi, è stato scelto di assegnare uno colore diverso a ciascuna pool diversa da quella centrale (della maestra), e gli elementi di quest'ultima sono stati colorati del colore dell'elemento con cui erano collegati (e lasciati bianchi qualora non fossero collegati a nessun elemento esterno). In altre parole, le attività della pool della maestra collegate alla pool della biglietteria della mostra sono state colorate di azzurro, mentre quelle collegate alla pool della biglietteria degli autobus sono state colorate di arancione.

¹Verranno richiamati i concetti fondamentali nel §3.1

²I marcatori sono dei simboli che nella notazione BPMN servono a fornire informazioni aggiuntive sull'elemento di cui fanno parte, o a modificare il comportamento di un elemento all'interno del diagramma di processo.

2.2 Scelta del software

In merito alla scelta del software per la progettazione dei diagrammi, se ne sono considerati di diversi: **Visual Paradigm Online** è un ottimo software che offre di progettare comodamente diagrammi BPMN utilizzando un browser web senza la necessità di scaricare l'applicazione. Oltre a questo software si sono esplorate altre risorse come **Camunda** o **yEd**. Quest'ultimo è uno strumento più generico che offre la possibilità di progettare molte altre tipologie di diagramma, mentre **Camunda** è invece più specifico per la modellazione BPMN. **yEd** può essere utilizzato per creare qualsiasi tipo di diagramma, mentre **Camunda** è progettato specificamente per la modellazione dei processi aziendali. Ciò significa che **yEd** offre una gamma più ampia di funzionalità, ma può essere più difficile da usare. Pertanto, per un motivo di usabilità, è stato scelto **Camunda**.

2.3 Scenario 1

Durante la prima fase del processo la maestra contatta la biglietteria della mostra e la biglietteria degli autobus per richiedere informazioni; concettualmente queste operazioni possono essere pensate come eseguite in modo parallelo o in modo sequenziale equivalentemente; tuttavia, quando i diagrammi verranno tradotti in reti di Petri e si dovranno effettuare analisi quantitative come verificare la vitalità della rete, sarà preferibile ai fini della complessità computazionale avere il minor numero possibile di operazioni in parallelo. Per questo motivo si è scelto di progettare il succedersi di queste attività in modo sequenziale. In Figura 1 è mostrata la prima parte del diagramma. Ogni pool è dotata di un evento iniziale e le attività delle diverse pools si scambiano messaggi tra di loro tramite frecce tratteggiate. Le attività che inviano messaggi (per esempio le richieste della maestra alle diverse biglietterie) sono contrassegnate da un simbolo a forma di lettera di colore nero, mentre le attività che ricevono messaggi (come ad esempio la ricezione da parte delle biglietterie dei messaggi inviati dalla maestra) presentano lo stesso simbolo ma di colore bianco.

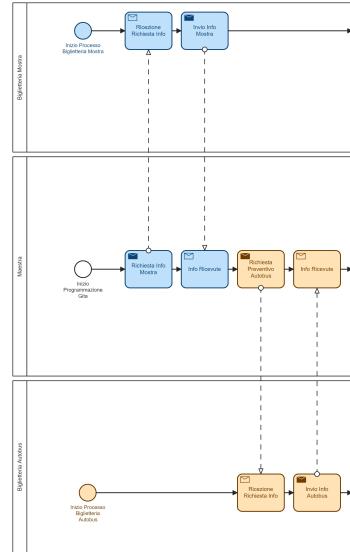


Figura 1: Richiesta informazioni da parte della maestra

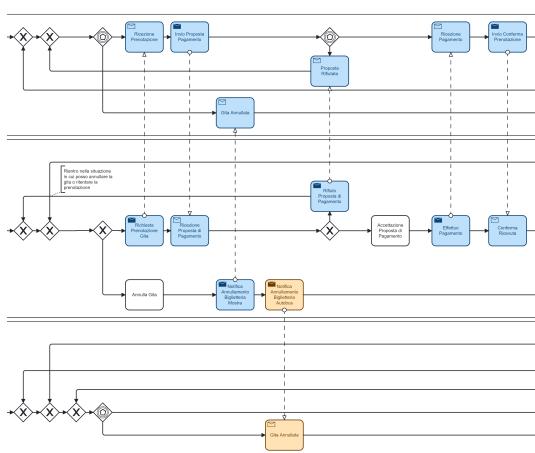


Figura 2: Prenotazione dei biglietti della mostra

dere la maestra. Nel caso in cui la maestra decida di annullare la gita, e dunque seguire il ramo inferiore dello snodo, si procederà al notificare opportunamente le due biglietterie (sempre in modo sequenziale per la stessa argomentazione che abbiamo usato in precedenza sul fatto di preferire la progettazione dei processi in maniera sequenziale piuttosto che in parallelo) e a quel punto tutti i processi termineranno; altrimenti, come si può vedere dalla predominanza del colore azzurro, si entra in una fase di dialogo tra

la maestra e la biglietteria della mostra (mentre la biglietteria degli autobus, ricordiamo, rimane in attesa). Qui la maestra prova a prenotare una data tra quelle proposte inizialmente per la mostra, la biglietteria della mostra propone un'opzione di pagamento relativa alla data scelta dalla maestra, e la maestra si trova davanti alla decisione di accettare o meno la proposta di pagamento della biglietteria della mostra. Ancora una volta questa decisione è rappresentata da uno XOR SPLIT in corrispondenza del quale c'è un EBG nella pool della biglietteria della mostra che rappresenta il fatto che quest'ultima deve attendere la conferma o il rifiuto della proposta di pagamento. Se la proposta di pagamento viene rifiutata, seguendo il ramo superiore dello snodo si ritorna al punto in cui la maestra può annullare la gita o proporre una nuova prenotazione; altrimenti il pagamento viene effettuato e di seguito confermato (per semplicità non si è tenuto in considerazione l'eventualità in cui il pagamento può andare storto ad esempio per problemi di solvibilità della maestra o problemi tecnici del sistema bancario).

A questo punto, come è possibile osservare in Figura 3, dopo aver comunicato con la biglietteria della mostra e aver prenotato i biglietti, la maestra può procedere alla verifica di disponibilità di posti per un (o più di uno eventualmente) autobus per portare la classe a vedere l'esposizione multimediale. Ricordando che la biglietteria degli autobus fino a questo momento si trovava in uno stato di attesa (in un EBG) è naturale (anzi d'obbligo) riprendere il suo flusso nella pool con un messaggio di ricezione (con un simbolo a forma di lettera di colore bianco); per la precisione è la maestra a richiedere la disponibilità degli autobus. Dopo aver ricevuto informazioni circa la disponibilità di posti per una determinata data (evidentemente compatibile con la data della mostra), la maestra può proporre una nuova data (e quindi ritornare indietro) attraverso un ciclo (aperto da uno XOR SPLIT e chiuso indietro da uno XOR JOIN), oppure procedere alla prenotazione. Anche in questo caso la biglietteria degli autobus propone una modalità di pagamento che la maestra può decidere se accettare o rifiutare. Come prima questa scelta è rappresentata da un altro ciclo aperto da uno XOR SPLIT e chiuso (indietro) da uno XOR JOIN nella pool della maestra e da un ciclo aperto da un EBG e chiuso da uno XOR JOIN nella pool della biglietteria degli autobus. In sostanza gli XOR JOIN che ci sono all'inizio di Figura 3 chiudono i due cicli del processo della maestra per proporre una nuova data a seguito della mancata corrispondenza con le date dei biglietti della mostra (già acquistati) e per effettuare un nuovo pagamento. Il processo continua quando la proposta di pagamento viene accettata. Sembra che la maestra debba obbligatoriamente accettare una proposta offerta ma sarà possibile effettuare il pagamento e annullarlo in seguito (non però dopo una settimana prima della data della gita).

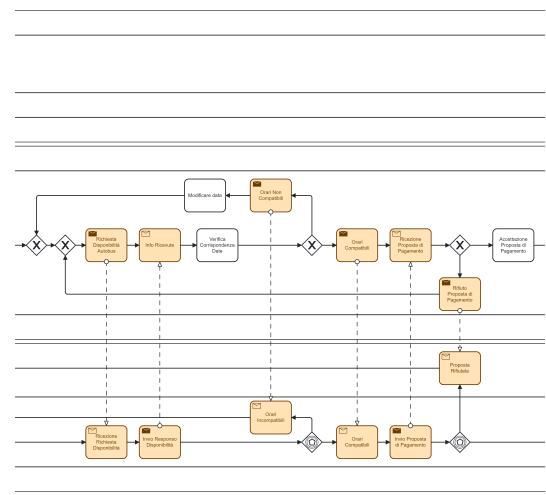


Figura 3: Prenotazione dei biglietti dell'autobus

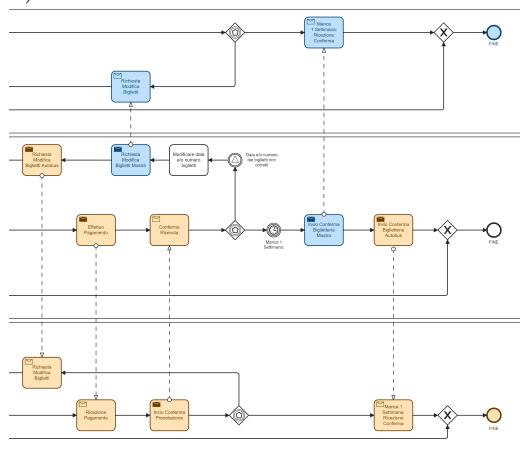


Figura 4: Fase finale del processo

Nell'ultima fase del processo, quella rappresentata in Figura 4, la maestra effettua il pagamento per la prenotazione dei biglietti dell'autobus, la biglietteria conferma l'avvenuto pagamento (anche qui per semplicità non si è tenuto conto di eventuali errori interni alla fase di pagamento), e tutti e tre i sottoprocessi si trovano davanti ad una condizione di attesa. I biglietti sono stati acquistati ma, fintanto che non manca una settimana alla data della gita, è possibile modificare la data e il numero dei biglietti. Dunque può avvenire che trascorra il tempo e che nulla vada storto, nel qual caso il processo prenderà il ramo di destra dello snodo nella pool della maestra comunicando ad entrambe le biglietterie la conferma della prenotazione e concludendo l'intero processo, o altrimenti, a seguito

della ricezione di un messaggio dall'esterno, la maestra provvederà a effettuare nuovamente la scelta di annullare la gita o pattuire dei nuovi orari tra le biglietterie riportando il flusso del processo allo snodo a cui si faceva riferimento all'inizio di Figura 2.

2.4 Scenario 2

Nello scenario il processo descritto in precedenza deve essere modificato semplicemente aggiungendo un vincolo: la gita deve essere approvata dal consiglio di classe prima di essere svolta. Al fine di modellare questa modifica è naturale inserire un nuovo attore nel processo, e dunque una nuova pool, quella del consiglio di classe, in cui le attività che comunicano con l'esterno e quelle che comunicano con essa saranno colorate di un altro colore. Per questo nuovo attore si è scelto il color violetto. Inoltre è anche naturale che la funzione svolta da tale nuovo attore si espandi unicamente all'inizio del processo; il consiglio di classe infatti, una volta approvata (o disapprovata) la gita non giocherà più nessun ruolo nel processo. Per questo motivo analizzeremo solamente la parte iniziale del processo dello scenario 2, visto che dopo il diagramma risulterà identico a quello dello scenario 1.

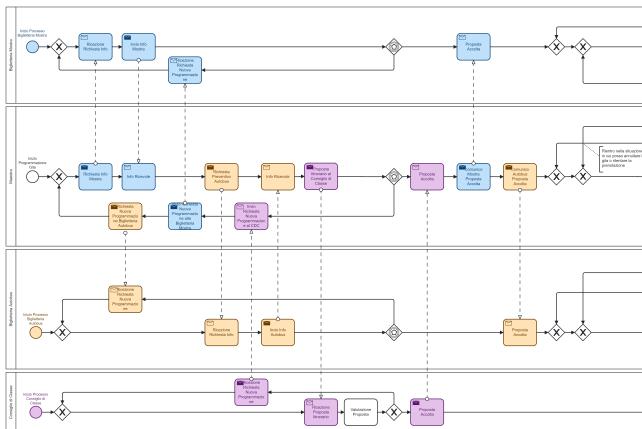


Figura 5: Adattamento del diagramma di scenario 1 alle specifiche di scenario 2

In Figura 5 è mostrato la parte del processo di scenario 1 modificata per adattarla alle specifiche dello scenario 2. Il processo parte inizializzando tutti e quattro i sottoprocessi. A questo punto l'unico che può andare avanti è quello della maestra, la quale chiede informazioni circa gli orari dell'esposizione alla biglietteria della mostra e gli orari degli autobus alla corrispondente biglietteria. Fino a questo momento il processo è identico a quello dello scenario 1. A questo punto, mentre le biglietterie rimangono in attesa di un messaggio, la maestra contatta il consiglio di classe per proporre un itinerario sulla base delle informazioni ricevute. Il flusso si sposta nella pool del consiglio di classe che deve decidere se approvare o meno l'itinerario proposto dalla maestra. Questa decisione è

rappresentata da uno **XOR SPLIT** nella pool del consiglio di classe. Qualora la proposta venga accolta dal consiglio il flusso può procedere verso una situazione in cui la maestra riprende la comunicazione con le biglietterie, il sottoprocesso del consiglio di classe termina e l'intero processo si ritrova nello snodo iniziale di Figura 2 in cui è possibile annullare la gita o proseguire con le prenotazioni. Qualora invece la proposta non venga accolta dal consiglio, questo può procedere a comunicare la sua disapprovazione alla maestra che a sua volta può procedere ad una nuova proposta di itinerario. In sostanza con questa modellazione la maestra continuerà a proporre nuovi itinerario finché il consiglio di classe non ne approverà uno. Questo modello non rispecchia a pieno ciò che potrebbe accadere nel mondo reale, e per questo motivo si è deciso di sviluppare una variante che tenga conto della possibilità da parte della maestra di annullare la gita a seguito di una disapprovazione del consiglio.

2.5 Variante dello scenario 2

In Figura 6 è mostrato il diagramma dello scenario 2 in cui si da alla maestra la possibilità di annullare la gita a seguito del rifiuto del consiglio alla richiesta di approvazione della gita. Una volta disapprovata la gita, il consiglio comunica la sua decisione alla maestra che, dopo averla a sua volta comunicata alle due biglietterie (che attenderanno una nuova programmazione o un eventuale annullamento della gita), può decidere se annullare la gita (comunicando tale decisione al consiglio di classe e alle biglietterie) facendo terminare tutti i sottoprocessi, o se proporre un nuovo itinerario al consiglio dopo aver richiesto nuove informazioni alle biglietterie.

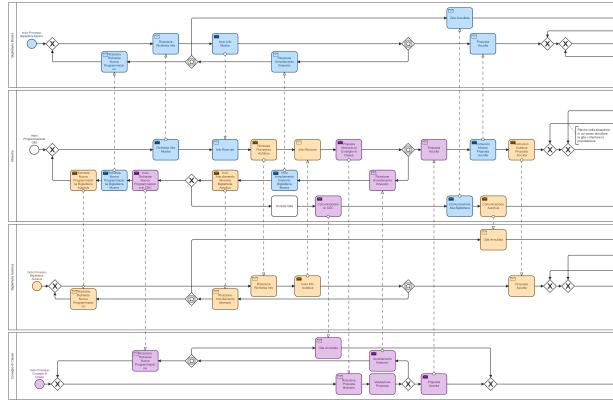


Figura 6: Variante dello scenario 2

3 Reti di Petri

Arrivati a questo punto è necessario analizzare i processi fin qui costruiti. Nel farlo ci serviremo delle reti di Petri, grafi orientati bipartiti (in cui i nodi possono essere piazze e transizioni, ed in cui ogni piazza può essere connessa a zero o più transizioni ed ogni transizione può essere connessa ad una o più piazze) che circa 60 anni fa sono stati inventati e approfonditamente studiati da Carl Adam Petri. Queste reti garantiscono che i processi aziendali da cui vengono opportunamente tradotte abbiano determinate caratteristiche e godano di importanti proprietà.

3.1 Nozioni fondamentali sulle reti di Petri

Nelle reti di Petri, le piazze si differenziano dalle transizioni per il fatto di poter ospitare un numero arbitrario $n \in \mathbb{N}$ di tokens al proprio interno. Denoteremo l'insieme delle piazze della rete con P , quello delle transizioni con T e l'insieme degli archi tra piazze e transizioni e tra transizioni e piazze con F . In questo modo una rete di Petri è definibile dalla tupla (P, T, F) . Le transizioni permettono di muovere i tokens tra le piazze all'interno della rete. Detto $(p, t) \in F$ l'arco che congiunge $p \in P$ a $t \in T$, una transizione t può essere attivata se nel suo preset $\bullet t \equiv \{p \in P | (p, t) \in F\}$ è presente almeno un token in ogni piazza. A seguito dell'attivazione ("firing" o anche "scatto") di $t \in T$ un token viene consumato da ogni piazza di $\bullet t$ ed un token viene creato in ogni piazza del postset $t\bullet \equiv \{p \in P | (t, p) \in F\}$. Una rete (P, T, F) dotata di uno o più tokens è detto sistema, ed è denotabile dalla tupla (P, T, F, M_0) dove M_0 è detta marcatura iniziale della rete. Una marcatura è un vettore con $|P|$ elementi in cui ciascun elemento corrisponde ad una piazza e rappresenta il numero di tokens in quella piazza. Una marcatura M' si dice raggiungibile da un'altra marcatura M se esiste una successione di transizioni $\sigma = t_1 t_2 t_3 \dots$ t.c. attivando σ si passa dalla marcatura M ad M' , e si scrive $M \xrightarrow{\sigma} M'$. Se in generale la sequenza $\sigma \in T^*$ è attivabile scriviamo $M \xrightarrow{\sigma}$. Si definisce inoltre $[M] \equiv \{M_1, M_2, \dots | \exists \sigma_i \text{ t.c. } M \xrightarrow{\sigma_i} M_i, i = 1, 2, \dots\}$, vale a dire l'insieme delle marcature raggiungibili da M . Dato $p \in P$ si indica con $M(p)$ il numero di tokens nella piazza p alla marcatura M . Enunciamo adesso alcune importanti proprietà delle reti di Petri:

- Un sistema (P, T, F, M_0) è vivo (per transizioni) se $\forall t \in T, \forall M \in [M_0], \exists M' \in [M]$ t.c. $M' \xrightarrow{t}$
- Un sistema (P, T, F, M_0) è vivo per piazze se $\forall p \in P, \forall M \in [M_0], \exists M' \in [M]$ t.c. $M'(p) > 0$
- Un sistema (P, T, F, M_0) è morto se $\forall t \in T, \forall M \in [M_0], \nexists t \mid M \xrightarrow{t}$
o equivalentemente $\forall t \in T, \forall M \in [M_0], M \xrightarrow{t}$
- Un sistema (P, T, F, M_0) è limitato se $\exists k \in \mathbb{N}, \forall M \in [M_0], \forall p \in P, M(p) \leq k$
- Un sistema è free-choice se $\forall t, u \in T$ o vale $\bullet t = u\bullet$, oppure $\bullet t \cap u\bullet = \emptyset$

o equivalentemente $\forall p, q \in P$ o vale $p\bullet = q\bullet$, oppure $p\bullet \cap q\bullet = \emptyset$

- Un sistema (P, T, F, M_0) è un S-system (S-net se non ha tokens) se $\forall t \in T$, $|\bullet t| = 1 = |t\bullet|$
- Un sistema (P, T, F, M_0) è un T-system (T-net se non ha tokens) se $\forall p \in P$, $|\bullet p| = 1 = |p\bullet|$

-

Infine concludiamo questa rapida rassegna con la definizione di alcuni oggetti inerenti le reti di Petri che saranno utili in seguito in fase di analisi:

- Indicheremo con \mathbf{N} la matrice $\mathbf{N} : (P \times T) \rightarrow \{-1, 0, +1\}$ rlativa alla rete di Petri $N = (P, T, F)$ avente come elementi:

$$\mathbf{N}(p, t) = \begin{cases} -1, & \text{se } (p, t) \in F \wedge (t, p) \notin F \\ +1, & \text{se } (p, t) \notin F \wedge (t, p) \in F \\ 0, & \text{se } ((p, t) \notin F \wedge (t, p) \notin F) \vee ((p, t) \in F \wedge (t, p) \in F) \end{cases}$$

- Un vettore $\mathbf{I} : P \rightarrow \mathbb{Q}$ è un S-invariant di $N = (P, T, F)$ se $\mathbf{I} \cdot \mathbf{N} = 0$
o equivalentemente se $\sum_{p \in \bullet t} \mathbf{I}(p) = \sum_{p \in t\bullet} \mathbf{I}(p)$, $\forall t \in T$
- Un vettore $\mathbf{J} : T \rightarrow \mathbb{Q}$ è un T-invariant di $N = (P, T, F)$ se $\mathbf{N} \cdot \mathbf{J} = 0$
o equivalentemente se $\sum_{t \in \bullet p} \mathbf{J}(t) = \sum_{t \in p\bullet} \mathbf{J}(t)$, $\forall p \in P$
- Sia $N = (P, T, F)$ e sia $\emptyset \subset X \subseteq P \cup T$. Chiamiamo S-component di N una sottorete $N' = (P \cap X, T \cap X, F \cap (X \times X))$ t.c. N' è una S-net fortemente connessa in cui $\bullet p \cup p\bullet \subseteq X$, $\forall p \in X \cap P$
- Data $N = (P, T, F)$, un insieme $\mathbf{C} = \{N_1, N_2, \dots, N_H\}$ di H S-components t.c. $\forall p \in P$, $\exists j = 1, \dots, H$ t.c. $p \in N_j$ è detto S-copertura.
- Una piazza p ed una transizione t formano un PT-handle se esistono due cammini che vanno da p a t aventi come nodi comuni solo p e t .
- Una transizione t ed una piazza p formano un TP-handle se esistono due cammini che vanno da t a p aventi come nodi comuni solo t e p .
- Una rete N è detta well-handled se non ha né PT-handles né TP-handles.
- Una rete N è detta well-structured se N^* è well-handled.

3.2 Workflow Nets

Il tipo di reti di Petri utilizzate per studiare i processi aziendali sono reti di Petri con un'unica piazza iniziale $i \in P$ t.c. $|\bullet i| = 0$, un'unica piazza finale $o \in P$ t.c. $|o\bullet| = 0$ ed in cui ogni piazza ed ogni transizione sono raggiungibili attraverso un cammino che va da i a o : i cosiddetti Workflow Nets (d'ora in poi WF-Nets per brevità). Data un WF net (P, T, F) studieremo sistemi in cui $M_0 = i$ (dove uguagliando i ad un vettore stiamo, con un po' di abuso di notazione, indicando quel vettore che ha 0 dappertutto tranne che nella posizione corrispondente alla piazza i). Indicheremo spesso con N una rete (P, T, F) generica, e con N^* quella rete che si ottiene da N aggiungendo una transizione *reset* t.c. \bullet reset = { o } e \bullet reset = { i }.

Tra i vari WF-Nets quelli che rappresentano un processo correttamente progettato sono i cosiddetti sistemi sound. Questi sistemi sono definiti come processi tali per cui:

1. $\forall t \in T \exists M \in [i] \mid M \xrightarrow{t} (\text{NO DEAD TASKS})$.
2. $\forall M \in [i] \exists M' \in [M] \text{ t.c. } M'(o) \geq 1 \text{ (OPTION TO COMPLETE)}.$
3. $\forall M \in [i] M(o) \geq 1 \Rightarrow M = o \text{ (PROPER COMPLETION)}.$

Questa è la tipologia di sistemi che sarebbe desiderabile ottenere dalla traduzione di un diagramma BPMN in rete di Petri.

3.3 Traduzione dei diagrammi in reti di Petri

Procediamo adesso a tradurre i diagrammi BPMN di Figura 4 e Figura 5 in opportuni WF Nets. Per ogni evento e per ogni attività si inserisce una transizione, per ogni freccia si inserisce una piazza; anche per ogni connettore AND si inserisce una transizione; mentre per ogni XOR SPLIT si inserisce una piazza seguito da due transizioni in parallelo di modo che si possa scegliere quale transizione attivare impedendo al token di proseguire lungo quella non attivata; infine per ogni XOR JOIN si inserisce una piazza preceduto da due transizioni in parallelo di modo da raccordare il flusso che proviene o da un ramo o dall'altro; infine per ogni EBG su eventi si inserisce una piazza.

La traduzione dei connettori OR è più complicata e peraltro il loro uso è sconsigliato a causa della difficoltà nel derivare processi sound. Per questo motivo si è appositamente cercato di fare in modo di non avere queste strutture logiche nel diagramma iniziale, pertanto non verranno discusse in questa sede.

Al fine di rendere la rete un WF-system è necessario infine connettere tra di loro i nodi iniziali dei vari sottoprocessi. Per fare ciò si è inserita una piazza iniziale, si è inserito un token all'interno, e si è connessa tramite un AND SPLIT tale piazza alle transizioni che rappresentavano i nodi iniziali dei sottoprocessi. In questo modo, una volta attivata la transizione dell'AND SPLIT, comparirà un token all'inizio di ciascun sottoprocesso. Anche la connessione tra i nodi finali è necessaria, e di modo da raccordare il flusso in arrivo da ciascun sottoprocesso con coerenza, si è scelto di farlo con un AND JOIN. In questo modo non appena sarà arrivato almeno un token (esattamente uno in realtà, nel caso di processi sound) da ciascun sottoprocesso, sarà possibile attivare l'ultima transizione (quella dell'AND JOIN) per produrre un token nella (unica) piazza finale della rete.

Una volta effettuata la traduzione del diagramma BPMN in questo modo, quello che si sarà ottenuto sarà un WF-system a patto che ogni nodo del sistema appartenga ad un cammino orientato che va dal (unico) nodo iniziale al (unico) nodo finale. Quest'ultima proprietà può essere verificata ad occhio o in modo formale attraverso un software come vedremo meglio nella sezione successiva.

4 Analisi delle reti

Per via della presenza di numerosi scambi di messaggi tra i partecipanti del processo, è naturale aspettarci che la rete non sia un S-net. Infatti gli scambi di messaggi realizzano delle transizioni con più di un arco entrante o con più di un arco uscente, e di conseguenza la condizione che ogni transizione abbia un solo arco entrante e un solo arco uscente non è soddisfacile. Anche l'aver inizializzato il processo con un AND SPLIT (e averlo terminato con un AND JOIN) non permette alla rete di essere un S-net, questo per via del fatto che i connettori logici di tipo AND comprendono transizioni con più archi uscenti o con più archi entranti (a seconda che sia SPLIT o che siano JOIN). D'altra parte la rete non è nemmeno un T-net³; questo per via dei diversi connettori logici di tipo XOR che si sono utilizzati. L'uso degli XOR, infatti, rende il sistema non deterministico, caratterizzandolo da scelte, le quali si realizzano in piazze aventi più archi entranti o più archi uscenti. Per gli S-nets o per i T-nets la soundness è di più immediata verifica. Il fatto che i sistemi qui presentati non siano di questi tipi ci obbliga ad una analisi più approfondita.

Si potrebbe controllare se il sistema sia composto da elementi sound (come le sequenze, le iterazioni, gli XOR impliciti, etc.) per derivare il fatto che è sound nella sua interezza. L'idea con la quale sono stati progettati i diagrammi è effettivamente questa, purtroppo però lo scambio dei messaggi tra i vari partecipanti e il modo con cui i processi si raccordano fra di loro rende non banale questa verifica, ed è pertanto più sicuro affidarsi a dei software che in maniera sistematica e quantitativa effettuano questo tipo di analisi.

WoPeD è un ottimo strumento che permette di progettare i sistemi derivanti dai diagrammi BPMN fornendo risultati sulle principali caratteristiche che una buona rete dovrebbe avere. Tuttavia quando le dimensioni delle reti diventano troppo grandi i tempi di esecuzione delle analisi svolte da WoPeD risultano troppo lunghi. Per questo motivo si è preferito procedere alla progettazione con WoPeD

³Per essere più precisi nessun WF-net può essere un T-net, avendo piazza iniziale e piazza finale con rispettivamente 0 archi entranti e 0 archi uscenti; quello che si fa quando si ha a che fare con i WF-nets (che è anche ciò che fa WoPeD di default) è di considerare N^* , ma nel nostro caso nemmeno N^* è un T-net.

e all'analisi con il plug-in **Woflan** di **ProM**, un programma di Process Mining. Quest'ultimo utilizza algoritmi più efficienti di **WoPeD**, e risulta dunque più utile nel determinare la soundness di un sistema. Tuttavia nel caso in cui i sistemi siano sound, non fornisce ulteriori dettagli (ad esempio sui PT-handles, i TP-handles, gli S-invariants, le S-components etc.⁴). Per questo motivo, in questa trattazione, si ometterà l'elenco di queste caratteristiche per le reti relative allo scenario 2 e alla variante dello scenario 2. Ci limitiamo invece ad elencare alcune metriche generali che mostriamo in Tabella 1.

	Scenario 1	Scenario 2	Variante
Numero di Piazze	107	126	124
Numero di Transizioni	93	108	104
Numero di Archi	236	282	286
Densità degli Archi	0.59%	0.53%	0.55%

Tabella 1: Alcune metriche di base dei i 3 sistemi progettati.

La rete dello scenario 1 è invece più contenuta, essendo infatti 3 pools a differenza delle 4 degli altri due diagrammi⁵, per questo motivo è stato possibile attendere un tempo ragionevole per ottenere i risultati delle analisi di **WoPeD**. Dai risultati ottenuti possiamo osservare come nella rete siano presenti 240 PT-handles e 250 TP-handles; in linea di principio questi risulterebbero un brutto segnale ai fini della buona progettazione della rete, potendo condurre a dead tasks (i PT-handles) e alla mancata verifica della condizione di proper completion⁶ (i TP-handles). Tuttavia essi non sono il risultato di erronei raccordamenti del flusso (come ad esempio potrebbe esserlo chiudere uno XOR SPLIT con un AND JOIN), bensì derivano dall'intricato modo con cui i messaggi tra le varie pools si sono giunti tra loro.

Dalle analisi di **WoPeD** il sistema presenta ben 776 S-components e risulta S-copribile, il che rappresenta un ulteriore segnale positivo della qualità del sistema.

I WF-systems che si sono ottenuti non risultano free-choice. Questo significa che non è sempre possibile scegliere una transizione senza restrizioni. In altre parole, se ci sono più transizioni che possono scattare contemporaneamente da una stessa piazza, la scelta tra queste transizioni non è libera ed è soggetta a vincoli. Per sistemi di questo tipo l'analisi può richiedere un tempo molto lungo, e per questo motivo l'uso di **Woflan** è imprescindibile.

Dalle analisi di **Woflan**, che comunque hanno richiesto un discreto quantitativo di tempo (dell'ordine delle svariate decine di minuti), i diagrammi progettati risultano essere WF-systems sound, ovvero i sistemi corrispondenti lo sono. Questo vuol dire che ogni transizione è attivabile almeno una volta (la proprietà 1. dei sistemi sound è soddisfatta), che per ogni marcatura raggiungibile dalla piazza iniziale si può raggiungere un'altra marcatura in cui la piazza finale contiene almeno un token (in altri termini è sempre possibile terminare il processo e dunque la proprietà 2. è soddisfatta), ed infine che quando il processo è terminato non ci sono altri token in giro per la rete e in o il token è uno solo (proprietà 3.).

Dal Main Theorem, che dice che N sound $\Leftrightarrow N^*$ vivo e limitato, deriva il fatto che tutti i sistemi sono vivi e limitati se si considera la transizione *reset*.

Per quanto riguarda il coverability graph, ovvero quel grafo avente come nodi le possibili marcature del sistema e come archi le transizioni che portano da un nodo all'altro, essendo il sistema sound, esso risulta avere un unico nodo finale (in cui confluiscono tutte le marcature) rappresentante la marcatura $M = o$. Dalla limitatezza del sistema segue inoltre che tale grafo non contiene nodi con molteplicità infinita, pertanto esso coincide con il reachability graph.

5 Conclusioni

In conclusione, i diagrammi progettati sono stati realizzati in modo corretto, assicurando la soundness, un concetto fondamentale che garantisce che le reti di Petri rappresentino processi eseguibili in modo controllato e senza rischio di situazioni di blocco o conflitto.

Nelle Appendici a seguire sono riportati i diagrammi completi disegnati con **Camunda** in notazione BPMN.

⁴WoPeD, a differenza di ProM, fornisce tutte queste proprietà (a patto di attendere più tempo).

⁵La complessità computazionale scala vertiginosamente con l'incrementarsi del numero di pools.

⁶si veda la condizione 3. in §3.2

Appendice A: Diagramma scenario 1

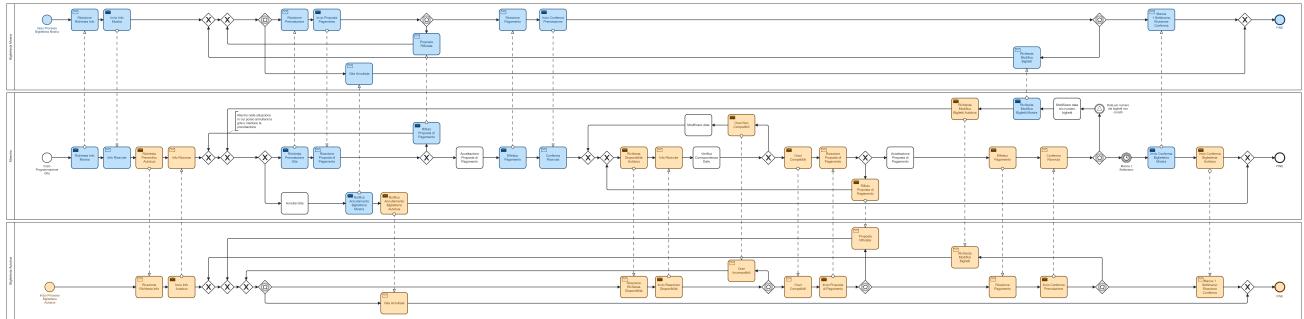


Figura 7: Diagramma completo dello scenario 1

Appendice B: Diagramma scenario 2

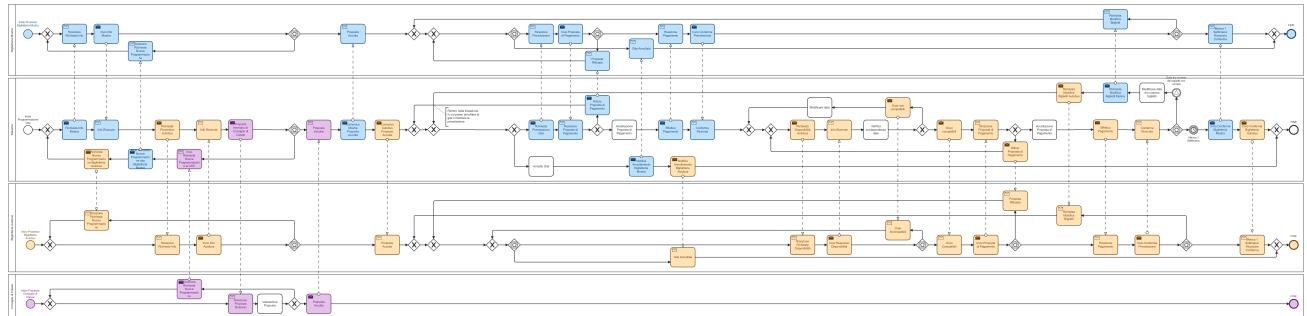


Figura 8: Diagramma completo dello scenario 2

Appendice C: Diagramma variante scenario 2

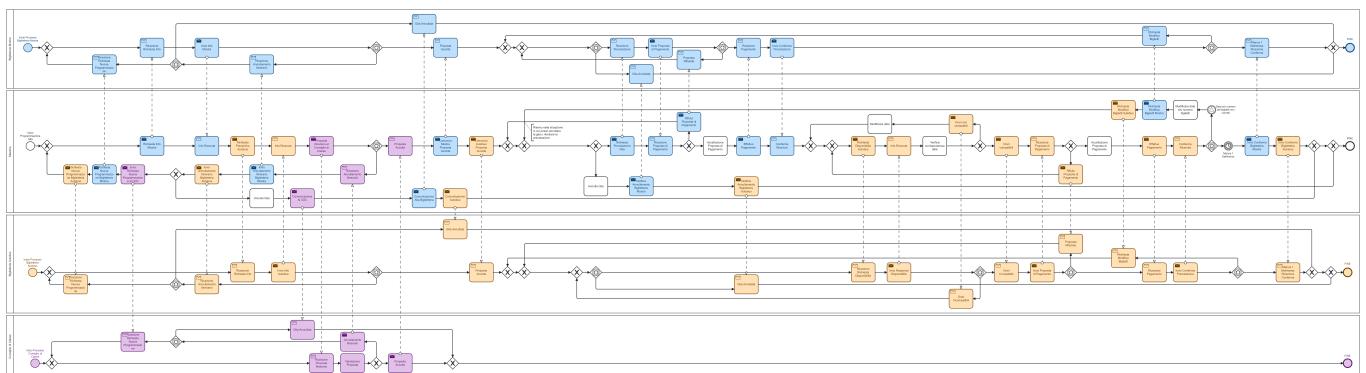


Figura 9: Diagramma completo della variante dello scenario 2