POLITECNICO MILANO 1863

BUSINESS PROCESS MODELING NOTATION (BPMN): ESERCIZI DI MODELLAZIONE

A cura di:
Cinzia Cappiello
Pierluigi Plebani
Monica Vitali

v.1.1 (10 Gennaio 2016)

Cinzia Cappiello è ricercatore presso il Dipartimento di Elettronica e Informazione del Politecnico di Milano. Svolge attività di ricerca nell'area di sistemi informativi prevalentemente su tematiche inerenti la gestione e la valutazione della qualità dei dati e della qualità del servizio. Su questi temi di ricerca ha scritto diversi articoli pubblicati su riviste e per conferenze accademiche internazionali. Negli ultimi anni gli interessi di ricerca si sono estesi anche ad altre aree quali il Green IT e la progettazione di applicazioni adattive. Il curriculum completo è disponibile alla pagina `http://home.deib.polimi.it/cappiell/`

Pierluigi Plebani è Ricercatore di Ruolo Confermato presso il Dipartimento di Elettronica Informazione e Bioingegneria del Politecnico di Milano dove ha conseguito il titolo di Dottore di Ricerca in Ingegneria dell'Informazione. È professore a contratto dei corsi di Sistemi Informativi per Ing. Gestionale e di Process and Service Design per Ing. Informatica. I suoi interessi di ricerca vertono sulla progettazione di Sistemi Informativi basati su architetture a Servizi e guidati dai processi di business. Recentemente si è occupato del ruolo dell'Internet of Things nei processi di business e di risparmio energetico in soluzioni Cloud. È autore di numerose pubblicazioni su riviste e conferenze internazionali nonché co-autore di alcuni libri di testo per corsi accademici e di certificazione EUCIP. Il curriculum completo è disponibile alla pagina `http://plebani.faculty.polimi.it`

Monica Vitali è Assegnista di Ricerca presso il Dipartimento di Elettronica Informazione e Bioingegneria del Politecnico di Milano dove ha conseguito il titolo di Dottore di Ricerca in Ingegneria dell'Informazione. Dal 2012 è assistente nei corsi di Sistemi Informativi per il Settore dell'Informazione (ing. informatica e telecomunicazioni) e di Sistemi Informativi (ing. gestionale). Dal 2015 è professore a contratto del corso di Sistemi Informativi per il settore dell'informazione. I suoi interessi di ricerca vertono sull'efficienza energetica dei sistemi informativi in ambienti cloud distribuiti e sullo studio di sistemi adattivi in grado di rilevare automaticamente problemi di funzionamento e di risolverli. Il curriculum completo è disponibile alla pagina `http://vitali.faculty.polimi.it`

Introduzione

Business Process Modeling Notation (BPMN) si sta sempre più affermando quale strumento standard per la modellazione di processi di business. La presente dispensa vuole offrire un compendio di facile e veloce consultazione per conoscere e capire quelli che sono i costrutti fondamentali di BPMN. La versione a cui si fa riferimento in questo testo è la 2.0, la cui specifica completa è disponibile all'indirizzo `http://www.omg.org/spec/BPMN/2.0`.

Essendo la specifica BPMN molto ampia, questa prima edizione della dispensa si concentra principalmente sulla modellazione dei processi e dei collaboration diagram, tralasciando gli aspetti legati alla coreografia dei processi. Il testo si suddivide nei seguenti capitoli:

Capitolo 1: come modellare le attività di un processo.
Capitolo 2: come definire il controllo di flusso di un processo attraverso i gateway.
Capitolo 3: come gestire e generare eventi in un processo.
Capitolo 4: come far dialogare i processi tra loro.
Capitolo 5: come dare un ruolo ai dati all'interno di un processo.
Capitolo 6: come gestire gli aspetti transazionali e di compensazione.
Capitolo 7: esercizi di autovalutazione per verificare l'apprendimento.

La dispensa si rivolge principalmente agli studenti dei corsi di *Sistemi Informativi* per Ingegneria Informatica e Gestionale tenuti al Politecnico di Milano. Riteniamo però che possa essere d'aiuto per tutti coloro i quali intendano avvicinarsi al linguaggio BPMN e capire la semantica che si cela dietro ai suoi numerosi costrutti grafici.

Come Donald Knuth, anche noi vorremmo ricompensare tutti i lettori con un dollaro per ogni errore trovato. Avendo però deciso di mettere a disposizione gratuitamente questa dispensa, ridimensioniamo la ricompensa ad un sentito grazie.

Milano, 22 Dicembre 2015

Cinzia Cappiello
Pierluigi Plebani
Monica Vitali

Ringraziamenti

Gli autori vogliono ringraziare in particolar modo Leonardo Bruni per l'aiuto nella redazione degli esercizi conclusivi. Si ringraziano inoltre Barbara Pernici e Maria Grazia Fugini per il supporto.

Indice

1 Modellare le attività di un processo: task e sottoprocessi

Un processo può essere definito come un insieme di attività che trasformano un input in output. Input ed output possono essere sia beni materiali che immateriali. In BPMN una attività è definita come un'unità di lavoro che richiede tempo per essere eseguita. Secondo la notazione BPMN, una attività è rappresentata mediante un rettangolo contenente il nome dell'attività stessa. Questo nome deve essere esplicativo e deve far capire cosa l'attività svolge. Per esprimere l'ordine in cui le attività sono eseguite durante il processo, vengono utilizzate delle frecce che collegano le attività tra di loro.

Esempio 1.1 - L'esempio riportato in Figura 1.1 mostra un primo esempio di processo. Ogni processo BPMN deve iniziare con un evento di tipo start (un cerchio con una linea sottile) e terminare con un evento di tipo end (un cerchio con una linea spessa). Per un approfondimento sugli eventi si rimanda al Capitolo 3. La relazione che lega due attività tramite una freccia direzionata dall'attività A all'attività B è il flusso sequenziale del processo e indica che l'attività B viene essere eseguita soltanto dopo che l'attività A è terminata.

La modellazione di un processo permette quindi di definire quali attività compongono un processo e l'ordine in cui devono essere eseguite.

Esempio 1.2 - Un processo può descrivere un qualsiasi insieme di attività orientate ad un obiettivo. In Fig. 1.2 è ad esempio riportato un semplice processo che descrive l'ordine in cui le attività necessarie alla preparazione di un caffè con la moka devono essere eseguite.

1.1 Ripetizione di una attività: loop e multi-instance

La notazione BPMN permette di definire dei tipi particolari di attività che possono essere utili a modellare attività che devono eseguire più volte il loro corpo prima di essere completate. Due tipologie possono essere definite, i cui simboli sono illustrati in Fig. 1.3:

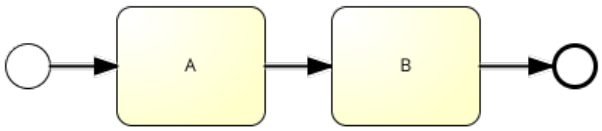


Figura 1.1: *Flusso sequenziale tra due attività in un processo*

Figura 1.2: *Processo di preparazione di un caffè*

- Attività di tipo *loop*: l'attività che è contraddistinta da un tipo loop viene eseguita finché una condizione booleana di terminazione non è verificata o finché non è stato superato un massimo numero di iterazioni. Nella notazione BPMN, l'attività di tipo loop è caratterizzata da una freccia circolare in basso.
- Attività di tipo *multi-instance parallelo o sequenziale*: l'attività contraddistinta da un tipo multi-instance genera un numero di istanze copia dell'attività stessa. Il numero di istanze da creare è stabilito da un espressione o è calcolato sulla base dei dati del processo, ma è comunque fisso nel momento in cui l'attività viene avviata. Le varie istanze possono essere eseguite in parallelo o in sequenza. Nella notazione BPMN, l'attività di tipo multi-instance è caratterizzata da tre linee parallele orizzontali (sequenziale) o verticali (parallelo) in basso.

Le due tipologie di attività appena descritte hanno molti punti di similarità e spesso possono essere utilizzate in modo intercambiabile. Le condizioni citate nella loro definizione sono implicite e non necessitano di essere specificate nel diagramma BPMN. È comunque possibile attaccare all'attività una annotazione che specifichi la condizione di terminazione del loop o l'espressione che genera il numero di istanze del multi-instance.

Esempio 1.3 - Il processo in Figura 1.4 mostra le attività coinvolte nella gestione di un ordine ricevuto da una società che si occupa di vendite on line. L'impiegato che prende in carico un ordine deve innanzi tutto inserire i dettagli dello stesso nel sistema e poi inviare al magazzino una richiesta per ogni prodotto presente nell'ordine. Attende poi che tutti i prodotti siano stati recuperati. Ogni prodotto viene quindi imballato, uno per volta, e a questo punto l'ordine viene spedito. L'impiegato invia la ricevuta dell'ordine al cliente e attende la ricezione del pagamento. Quando il pagamento viene ricevuto l'ordine viene archiviato.

Il processo modellato fa uso dei tipi di attività loop e multi-instance parallelo. Il multi-instance parallelo viene usato per l'attività *Invia Richiesta Magazzino*, che deve inviare una richiesta separata per ogni singolo prodotto presente nell'ordine. Queste richieste possono essere inviate contemporaneamente e il loro numero dipende dal numero di prodotti inseriti nell'ordine. Il tipo loop è invece usato nell'attività *Imballa Prodotto*. In questo caso il processo specifica che i prodotti devono essere

Figura 1.3: *Attività di tipo loop e multi-instance (parallelo e sequenziale)*

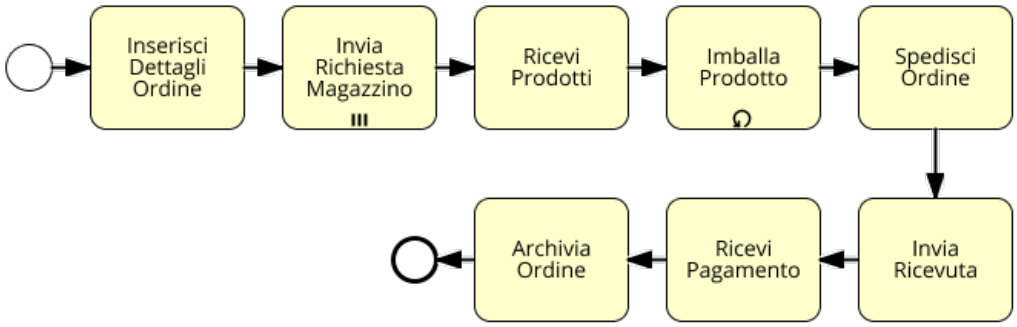


Figura 1.4: *Processo di gestione degli ordini di una società*

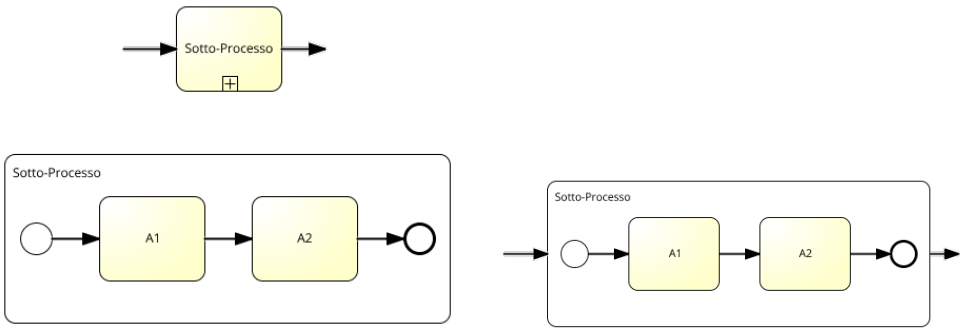
> imballati uno alla volta. L'attività sarà quindi ripetuta fino a quando tutti i prodotti non saranno stati imballati (condizione di terminazione del loop).

1.2 Sottoprocessi

Abbiamo definito un'attività come una unità di lavoro. Dai primi esempi è possibile vedere che in realtà alcune attività possono essere scomposte in unità più semplici che descrivono nel dettaglio come la singola attività deve essere eseguita. Possiamo quindi scomporre un'attività in un insieme di attività, anche queste caratterizzate da un ordine di esecuzione. Le attività che non possono essere ulteriormente scomposte sono dette attività atomiche. Le attività che possono invece essere dettagliate sono chiamate **sottoprocessi**. Due tipi di approcci possono essere utilizzati per l'individuazione dei sottoprocessi:

- scomposizione gerarchica: partendo da un'attività del processo la scompongo in unità più piccole che concorrono alla sua realizzazione;
- raggruppamento logico: raggruppo più attività del processo che concorrono insieme alla realizzazione di un sotto-obiettivo.

BPMN permette di modellare questi due approcci con due notazioni differenti mostrate in Figura 1.5. A sinistra è mostrata la rappresentazione di un sottoprocesso ottenuto per scomposizione gerarchica. Nel processo principale è inserita una versione compressa del sottoprocesso (caratterizzata da un quadrato contenente un "+" in basso) che viene poi esplosa a parte, mostrando i dettagli del sottoprocesso stesso. A destra



(a) Rappresentazione gerarchica (b) Scomposizione funzionale

Figura 1.5: *Rappresentazioni alternative di sottoprocessi in BPMN*

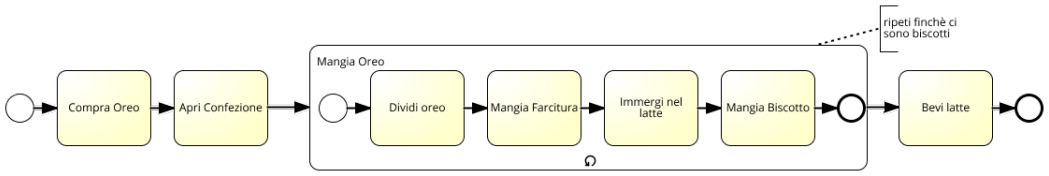


Figura 1.6: *Processo che descrive come mangiare una confezione di biscotti Oreo usando il raggruppamento funzionale*

invece è possibile vedere come rappresentare il raggruppamento logico. Il sottoprocesso è rappresentato nella sua versione espansa in cui in alto è mostrato il nome del sottoprocesso e all'interno dei suoi margini sono contenute le attività che lo compongono nell'ordine desiderato. È importante notare come il flusso del sottoprocesso è isolato dal resto del processo: un sottoprocesso ha un suo evento start ed un suo evento end e nessuna attività al suo interno può essere collegata con elementi al di fuori del sottoprocesso.

Esempio 1.4 - Un esempio di sottoprocesso è mostrato in Figura 1.6. Il processo descrive l'ordine delle attività necessarie per mangiare una confezione di biscotti Oreo. Le attività che concorrono ad ottenere il sotto-obiettivo di mangiare un biscotto sono raggruppate tra di loro nel sottoprocesso *Mangia Biscotto*, eseguito in loop. La condizione di terminazione è esplicitata usando una annotazione. Le attività che concorrono ad ottenere il sotto-obiettivo di mangiare un biscotto sono raggruppate tra di loro seguendo l'approccio del raggruppamento funzionale.

Esempio 1.5 - Lo stesso esempio è implementato usando la scomposizione gerarchica in Figura 1.7. Questo approccio è comodo quando non si vuole dare troppa enfasi al sottoprocesso. Il processo principale risulta più chiaro e leggibile nelle sue parti principali, lasciando i dettagli sul sottoprocesso a parte.

Come le attività, anche i sottoprocessi possono essere di tipo loop o multi-instance, indicando che tutte le attività che compongono il sottoprocesso sono ripetute più volte

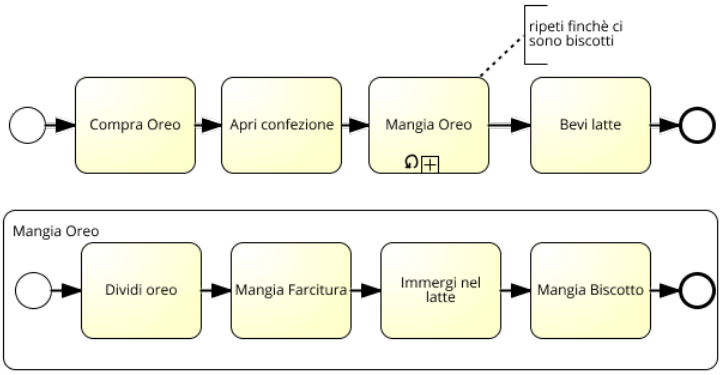


Figura 1.7: *Processo che descrive come mangiare una confezione di biscotti Oreo usando la scomposizione gerarchica*

sulla base delle condizioni di terminazione o del numero di istanze definito. Nel caso dei sottoprocessi è possibile definire anche altre tipologie illustrate di seguito.

1.2.1 Sottoprocessi di tipo ad-hoc

Un sottoprocesso può essere dichiarato di tipo *ad-hoc*. In questo caso, l'insieme delle attività che compongono il sottoprocesso non deve essere eseguita con un ordine stabilito, ma l'ordine e il numero di esecuzioni di ogni singola attività dipendono dall'esecutore. I sottoprocessi ad-hoc sono molto utili per modellare parti non strutturate del processo. Anche in questo caso, la terminazione di un sottoprocesso di tipo ad-hoc dipende da una condizione di terminazione implicita.

Esempio 1.6 - La Figura 1.8 mostra il processo di preparazione di una lezione. In questo caso, le attività che compongono il processo non devono seguire un ordine specifico e il numero di volte che ogni attività può essere eseguita può essere variabile. Per questo motivo, l'insieme delle attività può essere rappresentato usando un sottoprocesso di tipo ad-hoc.[a]

[a]Esempio tratto da Mathias Weske. Business process management: concepts, languages, architectures. Springer Science & Business Media, 2012.

1.2.2 Transazioni

Anche le transazioni sono un tipo particolare di sottoprocesso, rappresentati nella notazione BPMN da un doppio bordo. Le attività contenute all'interno di una transazione, anche se sono complesse, sono trattate come se fossero un'unica attività atomica. Questo vuol dire che nel sottoprocesso, o tutte le attività vengono eseguite con successo, o nessuna deve essere completata. Questo vuol dire che nel caso in cui un'attività fallisca, gli effetti delle altre attività incluse nella transazione devono essere cancellati. Di questo particolare tipo di sottoprocesso si discuterà più in dettaglio nel Capitolo 6.

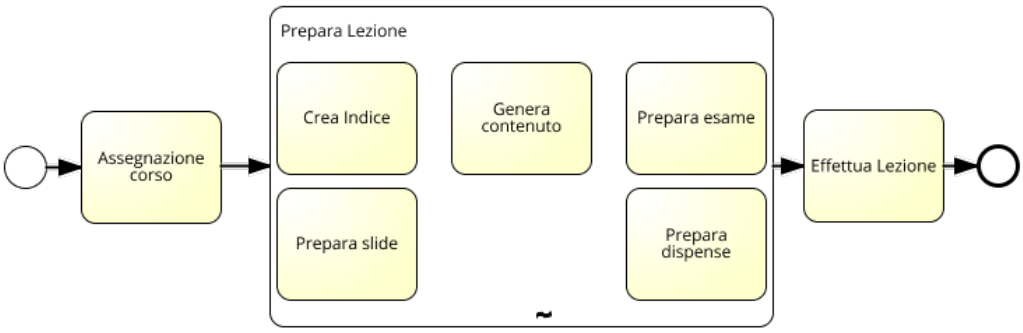


Figura 1.8: *Sottoprocesso di tipo ad-hoc descrivente le attività di preparazione di una lezione*

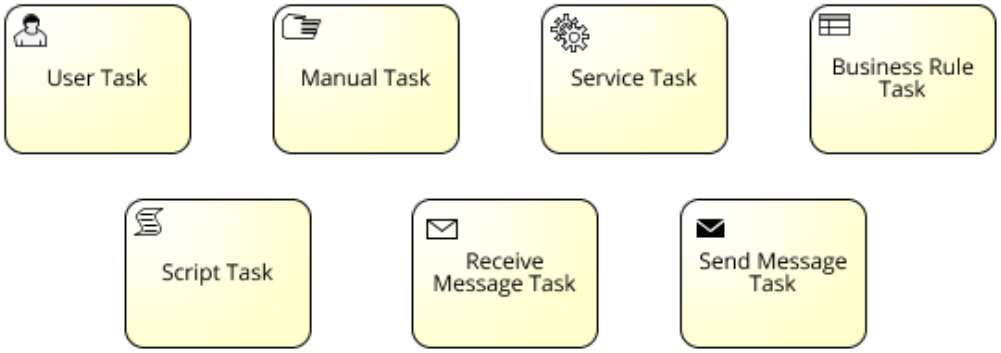


Figura 1.9: *Tipi di task*

1.3 Approfondimento - Tipologie di task in BPMN

Con la notazione BPMN 2.0 è possibile definire tramite delle icone nell'angolo in alto a sinistra la tipologia di ogni task. Diverse tipologie sono definite come mostrato in Figura 1.9 e sono brevemente descritte in questo paragrafo.

- User Task: sono attività che richiedono l'interazione con l'utente tramite il supporto del software (es. modifica della password);
- Manual Task: sono attività eseguite senza il supporto del software (es. installazione di un'apparecchiatura);
- Business rule: una business rule è una regola che deve essere interpretata. Un task che viene definito come business rule fornisce l'input all'interprete e ne ottiene un output nel momento in cui viene eseguito;
- Service Task: è una attività implementata tramite software;
- Script Task: un task che utilizza una qualche forma di linguaggio di scripting. Il task viene completato quando lo script termina;

- Receive Message Task: è un task che attende un messaggio proveniente da un'organizzazione esterna al processo durante la sua esecuzione;
- Send Message Task: è un task che invia un messaggio ad un'organizzazione esterna al processo durante la sua esecuzione.

È bene sottolineare che in BPMN il concetto di messaggio è esteso oltre che allo scambio di informazioni anche allo scambio di beni materiali. In generale, come sarà ricordato, i messaggi permettono di modellare una qualsiasi interazione tra il processo e un'entità esterna ad esso.

2 Modellare flussi alternativi: i gateways

I processi che abbiamo visto nel Capitolo 1 sono processi molto semplici in cui le attività che fanno parte del processo vengono eseguite una alla volta in modo sequenziale. In realtà, i processi aziendali sono molto più complessi di una semplice concatenazione di attività. Spesso infatti, le attività da eseguire possono essere differenti in base al contesto in cui vengono eseguite. Ad esempio, durante un processo di produzione, posso dover verificare se tutti i prodotti sono presenti in magazzino e, in caso non lo fossero, ordinarli. In altri casi, più attività possono essere eseguite in contemporanea, per cui stabilire un ordine di esecuzione sequenziale come fatto in precedenza è scorretto. Questo non può essere modellato con la semplice struttura che abbiamo visto. Per ovviare a questo problema, la notazione BPMN introduce i *gateways*.

I gateways sono degli snodi del processo da cui si diramano o in cui convergono più flussi alternativi. In base al tipo di gateway usato, è possibile attivare uno o più flussi sulla base del contesto di esecuzione del processo. In BPMN i gateways sono rappresentati con dei rombi, all'interno dei quali è contenuto un simbolo che ne definisce il tipo. I gateways possono essere usati in due modalità:

- modalità *split*: un gateway di questo tipo ha un flusso entrante e più flussi uscenti;
- modalità *join*: un gateway di questo tipo ha più flussi entranti e un solo flusso uscente.

I gateway possono anche essere usati in modalità ibrida (contemporaneamente split e join) ma questa soluzione non è raccomandata. Nei casi in cui sia necessario si consiglia infatti di utilizzare in successione un gateway split e un gateway join, o viceversa.

Nel resto del capitolo esamineremo i differenti tipi di gateway e come possono essere utilizzati in modalità split e join.

2.1 Exclusive Gateway (XOR)

Il gateway XOR, o gateway esclusivo, è un gateway che viene utilizzato quando durante l'esecuzione del processo è necessario effettuare una scelta, sulla base della quale viene eseguito un'insieme di azioni invece che un altro. Questo gateway è rappresentato in BPMN con un rombo contenente una X e può essere usato sia in modalità split che in modalità join. Usato in modalità split, il gateway XOR permette di modellare la

scelta di una sola opzione tra diverse alternative. Su ogni ramo uscente dal gateway è specificata una condizione e la prima che si verifica vera è quella che viene eseguita ed è quindi il percorso seguito dal token del processo. Nel gateway XOR è possibile inoltre indicare un ramo con condizione default, questo vuol dire che il ramo viene eseguito se nessuna delle condizioni sugli altri rami è verificata.

Esempio 2.1 - Nella parte sinistra di Figura 2.1 è mostrato un gateway XOR in modalità split. Quando il token arriva al gateway prenderà uno dei tre flussi possibili. Le condizioni espresse sui rami vengono valutate una alla volta, finché non si trova una condizione vera. L'ultimo ramo al posto della condizione ha una linea che sta ad indicare l'opzione di default: il ramo eseguito se nessuna delle altre condizioni è verificata. Una volta selezionato il ramo, il token arriverà alle attività su quel ramo, eseguendole seguendo il flusso del processo. Le attività sugli altri rami non saranno invece eseguite.

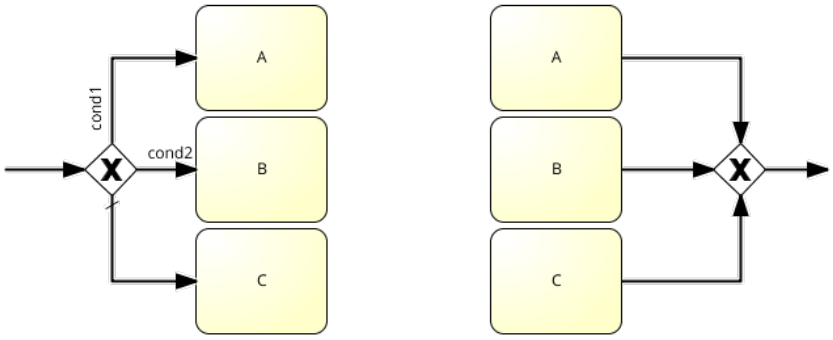


Figura 2.1: *Il gateway XOR*

Il gateway XOR può essere usato in modalità join. Questa modalità viene usata per ricongiungere flussi alternativi generati da gateway precedenti nel processo.

Esempio 2.2 - Il funzionamento di un gateway XOR in modalità join è mostrato nella parte destra di Figura 2.1. Quando un token arriva ad un gateway XOR viene fatto passare, senza aspettare il completamento degli altri flussi entranti nel gateway. In questo caso, ai diversi rami non è attaccata nessuna condizione, in quanto non si sta operando una scelta ma si stanno ricongiungendo più flussi.

Il gateway XOR può essere usato anche per modellare un loop nel processo in alternativa alle attività di tipo loop descritte nel Capitolo 1.

Esempio 2.3 - Un esempio di utilizzo di un gateway XOR in modalità split e join è mostrato in Figura 2.2. Una compagnia che si occupa di vendita di materiale elettrico vuole modellare un processo per la gestione degli ordini dei clienti. Il processo inizia al ricevimento di un ordine. L'impiegato inserisce quindi i dettagli dell'ordine nel sistema e la correttezza dei dati inseriti viene verificata. Se i dati sono corretti si verifica la disponibilità del materiale e si inoltra una notifica al cliente con la stima dei tempi di consegna. Se invece i dati non sono corretti si invia una richiesta

di aggiornamento dei dati. Quando l'aggiornamento viene ricevuto si riesegue il processo partendo dall'inserimento dei dati.

Il secondo gateway è usato in modalità split. La compagnia valuta se i dati dell'ordine sono corretti e in base a questa valutazione il token del processo passa alla verifica della disponibilità nel caso in cui lo siano, o alla richiesta di aggiornamento nel caso in cui non lo siano. Una volta ricevuto l'aggiornamento, è necessario ripetere le due operazioni di inserimento dei dettagli e di verifica della correttezza. Per unificare i due flussi che puntano all'attività "Inserisci Dettagli Ordine" si è usato un gateway XOR in modalità join. Questo vuol dire che ogni volta che un token arriva al gateway viene fatto procedere immediatamente all'attività successiva.

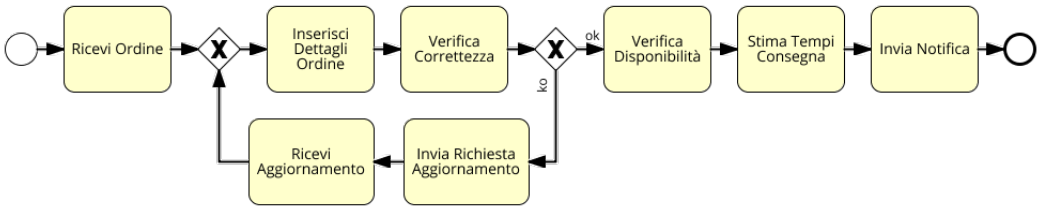


Figura 2.2: *Processo di gestione degli ordini di una compagnia con l'uso del gateway XOR*

2.2 Inclusive Gateway (OR)

Il gateway di tipo OR o gateway inclusivo è molto simile al gateway XOR. In BPMN è rappresentato con un rombo contenente un cerchio. Anche in questo caso il gateway può essere usato in modalità split o in modalità join. Nel caso di uso in modalità split, come nel caso precedente, ad ogni ramo uscente è attaccata una condizione. In questo caso però, se più condizioni sono verificate, il token del processo viene duplicato e un token viene generato per ogni ramo. Quindi più percorsi possono essere intrapresi contemporaneamente. Anche in questo caso è possibile avere la condizione di default su uno dei rami. Il ramo a cui è attaccata la condizione di default viene eseguito solo se le condizioni su tutti gli altri rami sono false, questo per garantire che almeno un ramo venga sempre attivato.

Esempio 2.4 - Nella parte sinistra di Figura 2.3 è mostrato un gateway OR in modalità split. Quando il token arriva al gateway, esso viene duplicato per ognuno dei rami su cui le condizioni sono verificate. I token arriveranno alle attività su quei rami, eseguendole anche contemporaneamente, senza un ordine prefissato. Ad esempio, se fossero verificate le condizioni cond1 e cond3, le attività A e C (e tutte quelle che le seguono) verrebbero eseguite. Sebbene su ogni singolo ramo verrà rispettato l'ordine in cui le attività sono modellate, tra i due rami non c'è nessun ordine stabilito di esecuzione.

Il gateway OR può essere usato in modalità join. Questa modalità viene usata per ricongiungere flussi alternativi generati da gateway precedenti nel processo. Questo utilizzo dell'OR permette quindi di sincronizzare tra loro i flussi attivi del processo. Il

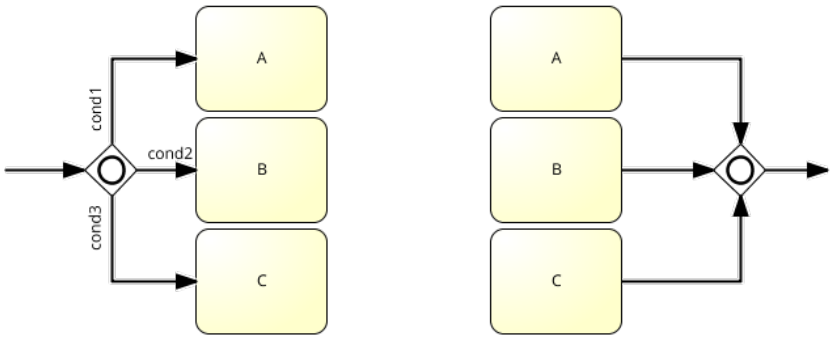


Figura 2.3: *Il gateway OR*

join determina quanti token erano stati attivati in origine e attende soltanto quei token, mentre ignora gli altri rami.

Esempio 2.5 - Il funzionamento di un gateway OR in modalità join è mostrato nella parte destra di Figura 2.3. Il gateway OR aspetta i token provenienti da tutti i rami entranti attivi nel processo. Quando tutti i token sono stati raccolti, un unico token procede il percorso passando attraverso il gateway. Se nell'esempio in figura fossero attive le attività B e C, il gateway aspetterebbe il completamento di entrambe per procedere con l'esecuzione del processo. Per questa sua proprietà permette la *sincronizzazione* dei soli flussi attivi nell'istanza del processo.

Esempio 2.6 - Un esempio di utilizzo di un gateway OR in modalità split e join è mostrato in Figura 2.4. Una banca riceve del denaro che deve essere depositato sul conto corrente del suo cliente. Prima del deposito è necessario effettuare delle operazioni. Nel caso in cui l'ammontare del deposito sia superiore a 10000 euro è necessario effettuare delle operazioni di gestione di importi elevati. Nel caso in cui invece il versamento provenga da una banca straniera è necessario svolgere delle operazioni per gestire la provenienza del denaro. Se invece nessuna delle due condizioni è verificata, il deposito viene trattato con le procedure standard. Una volta effettuate tutte le operazioni necessarie, la pratica di deposito viene archiviata.

Il primo gateway OR è usato in modalità split. Una o più condizioni possono essere verificate. Nel caso in cui il deposito sia straniero e di importo alto, due token vengono generati e le attività sul primo e sul terzo ramo sono eseguite. Se una sola delle due condizioni è verificata un solo token arriva sul ramo corrispondente. Se nessuna condizione è verificata il token passa sul ramo di default. Dal momento che prima di procedere all'archiviazione tutte le lavorazioni devono essere completate, un gateway inclusivo è usato come join per sincronizzare tutte i rami attivi che lo precedono. Quando tutti i token attivi lo raggiungono, il flusso procede con un unico token che arriva all'attività "Archivia deposito".

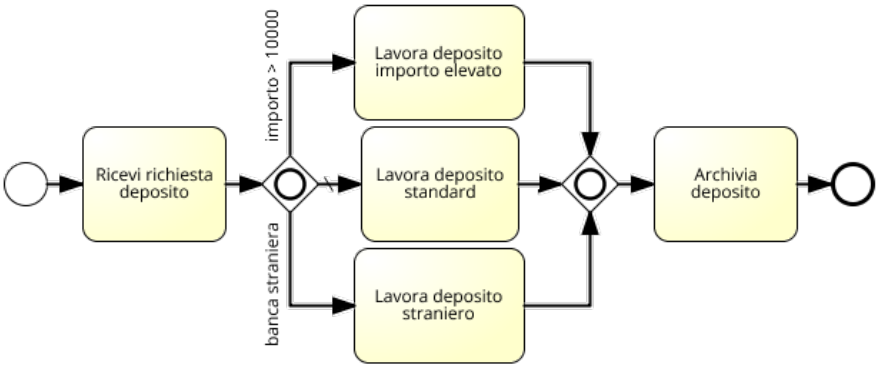


Figura 2.4: *Processo di gestione di un deposito di denaro in una banca*

2.3 Parallel Gateway (AND)

Il gateway di tipo AND o gateway parallelo permette di modellare situazioni in cui più attività possono essere svolte contemporaneamente o con un ordine non definito. In BPMN è rappresentato con un rombo contenente un "+". Anche in questo caso il gateway può essere usato in modalità split o in modalità join. Nel caso di uso in modalità split tutti i rami uscenti dal gateway devono essere eseguiti. Di conseguenza nessuna condizione è attaccata ai rami e un token viene sempre generato per ogni possibile percorso.

Esempio 2.7 - Nella parte sinistra di Figura 2.5 è mostrato un gateway AND in modalità split. Quando il token arriva al gateway, esso viene duplicato per ognuno dei rami. Le attività su quei rami saranno sempre eseguite, senza rispettare un ordine prefissato (quindi anche contemporaneamente). Come nel caso di più rami attivi con un gateway OR, su ogni singolo ramo verrà rispettato l'ordine in cui le attività sono modellate, mentre tra i rami non c'è nessun ordine stabilito di esecuzione.

Il funzionamento di un gateway AND in modalità join è mostrato nella parte destra della figura. Il gateway aspetta i token provenienti da tutti i rami entranti. Quando tutti i token sono stati raccolti, un unico token procede il percorso passando attraverso il gateway. Nell'esempio tutte le attività (A, B, e C) dovranno essere completate prima che il token possa andare avanti. Per questa sua proprietà permette la *sincronizzazione* di flussi alternativi.

Esempio 2.8 - Un esempio di utilizzo di un gateway AND in modalità split e join è mostrato in Figura 2.6. Una società riceve un ordine da un cliente. A questo punto deve preparare la spedizione dei prodotti e inviare la fattura al cliente. L'ordine in cui queste attività vengono eseguite non è importante. Quando entrambe sono state completate il processo può proseguire.

Il primo gateway OR è usato in modalità split. Tutti i rami sono attivati contemporaneamente e, nel caso specifico, due token vengono generati attivando le due

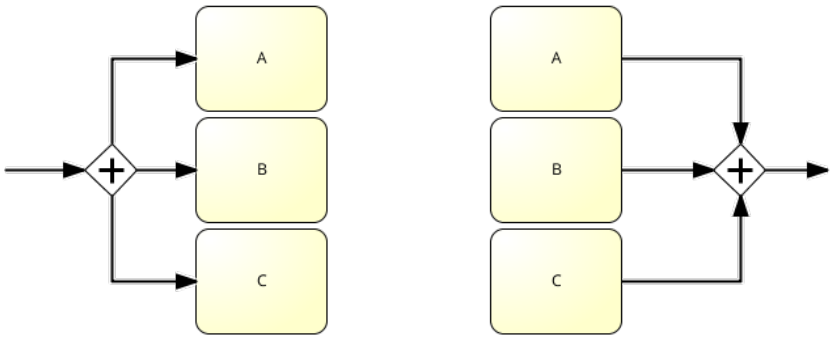


Figura 2.5: *Il gateway AND*

attività sui rami. Un gateway AND di tipo join viene invece usato per attendere il completamento di entrambe le attività prima di procedere.

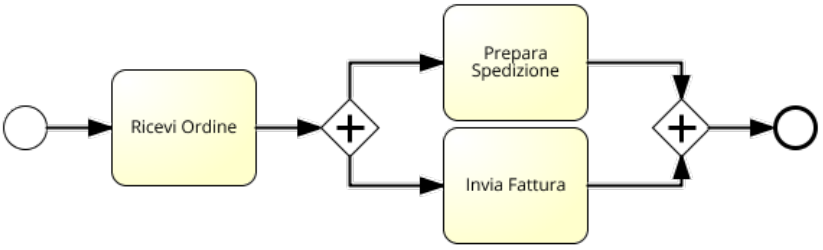


Figura 2.6: *Processo di preparazione di un ordine utilizzando il gateway parallelo*

2.4 Complex Gateway

Il gateway di tipo complesso permette di modellare situazioni in cui le condizioni di attivazione dei rami sono composite. Può ad esempio esprimere condizioni come "qualsiasi coppia di rami può essere attivata". In BPMN è rappresentato con un rombo contenente un "*". La condizione in questo caso non è attaccata ai rami ma è definita dall'utente e implicitamente rappresentata dal gateway. Dato che il comportamento del gateway complesso non è esplicitato in modo visuale, dovrebbe essere usato con cautela nella modellazione. Un esempio di complex gateway è mostrato in Figura 2.7.

2.5 Event-based Exclusive Gateway

I gateway che sono stati affrontati finora all'interno di questo capitolo basano la decisione del ramo del processo in cui proseguire su dati interni al processo. In alcuni casi però, la decisione su quale azione intraprendere può dipendere da qualcosa che avviene all'esterno. Ad esempio, nel caso di un processo di business che invia una proposta di vendita ad un cliente, l'azione da intraprendere dopo l'invio della proposta può dipendere dal fatto che il cliente l'accetti o la rifiuti. In questi casi, in cui la decisione

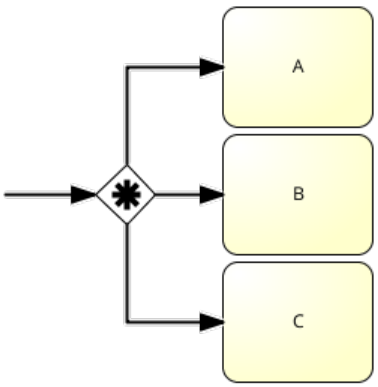


Figura 2.7: *Il gateway complesso*

si basa su qualcosa che avviene all'infuori del processo, si utilizza un altro tipo di gateway esclusivo: l'event-based gateway. Questo gateway sarà discusso in dettaglio nel Capitolo 3 quando introdurremo il concetto di evento.

2.6 Riepilogo sui gateway

I gateway permettono di modellare flussi alternativi all'interno di un processo. Con il gateway si decide quali sono i percorsi da seguire sulla base di alcune condizioni sui dati utilizzati nel processo. In base al tipo di gateway utilizzato uno o più flussi possono essere percorsi. Per ricongiungere più flussi i gateway possono essere utilizzati in modalità join.

I gateway sono elementi che non contengono logica al loro interno. Essi semplicemente indirizzano i token sulla base di alcune condizioni che devono essere valutate sui dati presenti nel processo. Nel caso in cui i dati su cui i gateway devono eseguire la valutazione non siano presenti in modo esplicito, questi devono essere calcolati prima di effettuare la decisione. Può quindi essere necessario far precedere il gateway XOR o il gateway OR da una attività che si occupi di calcolare i dati su cui le condizioni saranno valutate. Un processo che utilizza diversi tipi di gateway per modellare il suo flusso è descritto nell'esempio seguente.

Esempio 2.9 - Una società che fornisce apparecchiature elettroniche, riceve una richiesta di un preventivo da parte di un partner. Per prima cosa si verifica che il partner non abbia debiti pendenti con l'azienda. Se così è, la richiesta viene rifiutata. Altrimenti, l'azienda verifica la disponibilità delle apparecchiature richieste e, se non sono presenti in magazzino, stima la data in cui lo saranno. Nel frattempo, mentre verifica la disponibilità, stima il tempo necessario per la spedizione. Quando queste operazioni sono completate invia la proposta al partner.

La modellazione del processo descritto è mostrata in Figura 2.8. Il primo gateway XOR permette di seguire due percorsi alternativi: se il partner è debitore la richiesta viene rifiutata, altrimenti si procede con la preparazione del preventivo. Dal momento che la stima dei tempi di spedizione può essere eseguita in modo indipendente dalle altre attività, viene usato in gateway AND da cui partono due percorsi

paralleli. Grazie all'uso di un gateway XOR l'attività "Stima Tempi Magazzino" viene eseguita solo se i prodotti non sono disponibili. Subito dopo uno XOR di tipo join ricongiunge i due percorsi alternativi (solo uno sarà attivo). Prima di inviare la proposta le attività su entrambi i rami paralleli devono essere completate. È quindi necessario inserire un gateway AND per sincronizzare i due flussi prima di procedere.

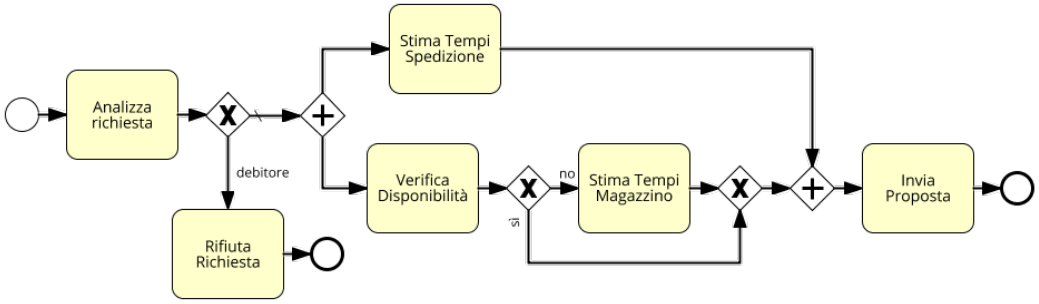


Figura 2.8: *Processo di creazione di un preventivo*

Esempio 2.10 - In Figura 2.9 il processo relativo al deposito di un assegno tramite lo sportello automatico di un istituto di credito. L'utente si reca presso l'istituto bancario per depositare un assegno. Una volta effettuata l'autenticazione, l'utente sceglie tra le opzioni disponibili quella di depositare un assegno. Dietro richiesta del sistema, l'utente inserisce le informazioni su chi ha erogato l'assegno, la data e la cifra e inserisce l'assegno nell'apposita fessura. Il sistema verifica le informazioni. Se i dati inseriti dall'utente non corrispondono con quelli letti dall'assegno, il sistema chiede all'utente di inserire nuovamente i dati. Se l'assegno non viene considerato valido dal sistema, viene restituito all'utente, che lo ritira. Altrimenti, se tutto è corretto, l'utente sceglie se richiedere al sistema l'erogazione della ricevuta. In tal caso, invia la richiesta e ritira la ricevuta. In fine, in ogni caso, il sistema mostra il saldo aggiornato all'utente.

Le operazioni necessarie al deposito dell'assegno sono state racchiuse nel sottoprocesso "Deposita Assegno".

I gateway che sono stati analizzati in questo capitolo permettono di definire il percorso che deve seguire un'istanza del processo sulla base di informazioni contenute all'interno del processo stesso. In alcuni casi invece, la scelta del percorso può dipendere da informazioni provenienti da eventi che accadono durante l'esecuzione del processo. Questo caso sarà trattato nel Capitolo 3.

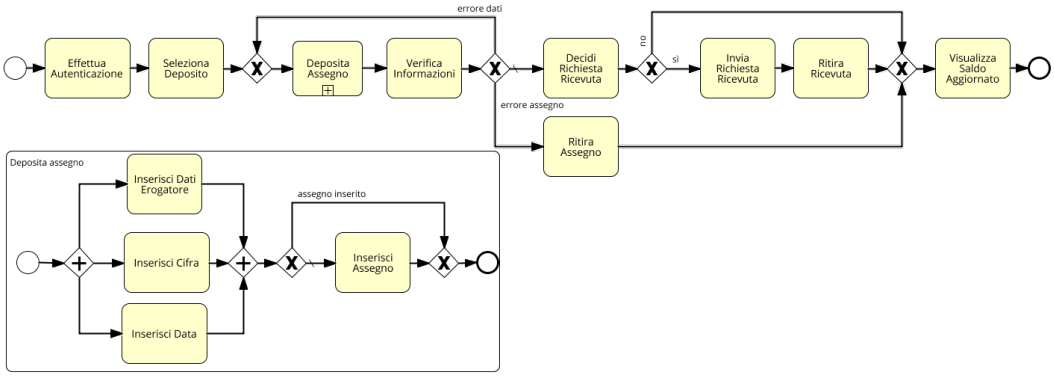


Figura 2.9: *Processo di deposito di un assegno*

3 Gli eventi

Attraverso il concetto di evento, BPMN permette di modellare situazioni in cui il flusso di un processo è condizionato da accadimenti esterni di varia natura, oppure di modellare situazioni in cui il processo è esso stesso colui il quale genera tali accadimenti. Il simbolo generico che identifica un evento è il cerchio al cui interno è possibile inserire l'icona corrispondente alla tipologia di evento che si sta modellando.

Secondo la specifica BPMN esistono tre macro classi di eventi:

start event (cerchio con bordo sottile): eventi che producono una nuova istanza del processo. L'istanza può riferirsi sia ad un processo che ad un sottoprocesso;

intermediate event (cerchio con doppio bordo sottile): eventi che accadono durante l'esecuzione del processo. Possono essere inseriti all'interno di un flusso di processo (intermediate catching/throwing event) oppure essere inseriti come boundary di una attività (boundary intermediate event);

end event (cerchio con bordo spesso): eventi che definiscono quando un processo termina e in che modo.

Esempio 3.1 - Un semplice esempio del loro utilizzo è rappresentato in Figura 3.1. In questo caso si ha un semplice processo che inizia (start event), esegue una attività A, attende l'accadimento di un evento (intermediate event) e, dopo aver eseguito l'attività B, termina (end event). In questo caso l'evento intermedio corrisponde allo scadere di un timeout. E' importante sottolineare come l'aver posizionato l'intermediate event nel flusso di processo costringa il processo ad attendere l'accadimento dell'evento (in questo caso la scadenza del time out) prima di eseguire l'attività B.

Esempio 3.2 - Un secondo esempio, riportato in Figura 3.2, illustra una diversa semantica degli intermediate event dettata dal diverso posizionamento dell'evento. In questo caso, infatti, l'evento è posizionato sul boundary di una attività (si parla di

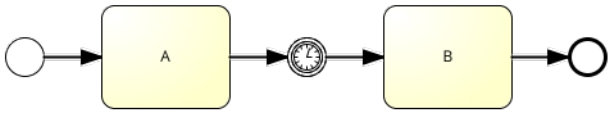


Figura 3.1: *Processo con intermediate event (di tipo timer) nel flusso di processo.*

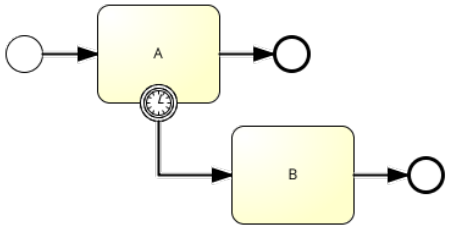


Figura 3.2: *Processo con intermediate event nel boundary di attività.*

boundary intermediate event) e il suo accadimento modifica il flusso di esecuzione. Si vengono quindi a creare:

- un *normal flow* (detto anche *happy flow*), corrispondente al flusso eseguito nel caso in cui l'evento boundary non accade. L'attività quindi termina normalmente e il processo termina.
- un *exceptional flow*, l'evento accade prima della terminazione dell'attività e il flusso del processo prosegue secondo quanto indicato dalle transazioni uscenti dall'intermediate event.

Nel caso di Figura 3.2 l'intermediate event rappresenta lo scadere di un timeout. Quindi l'attività B sarà eseguita solo se l'attività A non termina la sua esecuzione prima dello scadere di tale time out. Se invece l'attività A termina prima dello scadere del time out, l'attività B non sarà mai eseguita.

3.1 Eventi catching e throwing

Fino a questo punto gli eventi sono stati utilizzati per definire un accadimento che viene catturato (*catching*) e gestito durante il processo. In BPMN è però possibile anche modellare situazioni in cui il processo genera (*throwing*) eventi. La distinzione tra i due casi si evidenzia dal colore assunto dall'icona inserita all'interno del cerchio che identifica l'evento. Se l'icona non ha un colore di riempimento allora identifica un catching event, se invece è totalmente nera identifica un throwing event.

Esempio 3.3 - La distinzione tra catching e throwing trova una sua possibile esemplicazione in Figura 3.3. In questo caso si ha un processo che ha inizio alla ricezione di un messaggio (catching). Successivamente viene svolta l'attività A, dopodiché il processo termina generando un messaggio (throwing).

Figura 3.3: *Distinzione tra event catching and event throwing.*

Va sottolineato come gli intermediate event possano essere posizionati sul boundary sia di task (attività atomiche) sia di attività che racchiudono dei sottoprocessi. Nel secondo caso, la definizione del sottoprocesso può includere anche la generazione dell'evento che poi viene catturato a livello di processo.

Esempio 3.4 - Un esempio di questo tipo è rappresentato in Figura 3.4 dove a sinistra è rappresentato un processo simile a quanto descritto nell'Esempio 4.2, con l'unica differenza che l'intermediate event posizionato sul boundary è di tipo *error*: questo per indicare una situazione di errore che si viene a creare all'interno dell'attività. Il dettaglio dell'attività A è mostrato nella parte destra della figura. Come si può notare il sottoprocesso può terminare in due modi: tutte le informazioni sono a disposizione, quindi è possibile completare le operazioni richieste, oppure alcune informazioni mancano quindi il sottoprocesso termina generando una situazione di errore (end event di tipo throw).

Attraverso questa struttura, BPMN permette di modellare situazioni in cui gli accadimenti generati a livello di sottoprocesso (in questo caso la situazione di errore), sono gestiti a livello superiore.

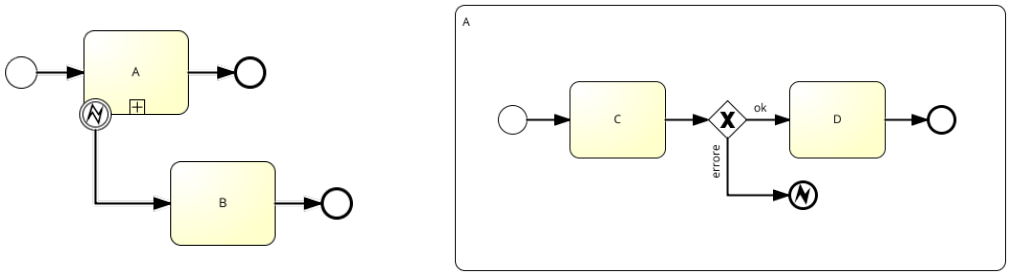


Figura 3.4: *Boundary intermediate event associato ad un sottoprocesso.*

3.2 Trigger

Negli esempi precedenti sono stati introdotti solo alcuni tipi di evento: timer, message ed error. In realtà BPMN permette di definire altri tipi di *trigger* evento dove per alcuni di essi ha senso parlare sia di catching sia di throwing event, mentre per altri solo di catching:

Message		L'evento è collegato ad un qualsiasi scambio con l'esterno (e.g., dato, materiale) proveniente o indirizzato ad un partecipante del processo (pool differente da quello che contiene l'evento di tipo messaggio).
Timer		L'evento corrisponde a un qualsiasi evento temporale, quale la scadenza di un timeout (anche ripetibile) oppure il raggiugimento di un dato giorno o ora.
Error		Permette di gestire situazioni di errore durante l'esecuzione di un processo. Nel caso un errore venga manifestato all'interno di un sottoprocesso, è il processo che contiene tale sottoprocesso a dover gestire la situazione di errore.
Conditional		L'evento si scatena al verificarsi di una condizione definita nel processo.
Parallel Multiple		Utilizzato per identificare situazione in cui l'accadimento di *tutti* gli eventi appartenenti ad un insieme di eventi, anche di diverse tipologie, scatena l'evento stesso.
Link		Permette di modellare salti all'interno di un flusso di processo. Spesso sono utilizzati per migliorare la leggiblità di uno schema.
Escalation		Identifica un evento scatenato all'interno di un sotto-processo e che deve essere gestito dal processo che contiene tale sottoprocesso. Come struttura è simile all'evento error con la differenza che l'escalation non necessariamente fa riferimento ad una situazione anomala.
Cancel		È utilizzato solo nella modellazione di sotto-processi di tipo transazionale (v. Capitolo 6) e indica la necessità di cancellare una transazione.
Compensation		è utilizzato solo nella modellazione di sotto-processi di tipo transazionale (v. Capitolo 6) e permette di gestire le attività di compensazione a fronte della richiesta di cancellazione di una transazione.
Signal		Un signal identifica un evento generato da un processo e ad un altri processi (broadcast), oppure a differenza dei messaggi, può anche essere inviato al processo stesso a patto che sia ad un livello differente.

Multiple		Utilizzato per identificare situazioni in cui l'accadimento di *almeno* uno tra gli eventi di serie di eventi, anche di diverse tipologie, scatena l'evento stesso.
None		Evento generico di cui non si conosce la reale natura, o la cui natura non ricade nei casi precedenti. In tal caso il cerchio, simbolo dell'evento, non contiene alcuna icona.

3.3 Start event

Un evento di tipo start definisce un evento che genera una nuova istanza del processo e, dato un processo, possono esservi anche più start event. Ovviamente il processo parte nel momento in cui il primo degli eventi accade e segue il flusso generato da quell'evento. Gli eventi start possono essere di tipo diverso secondo quanto definito nella sezione precedente. Eccezione fanno i sotto-processi, per i quali l'unico evento di tipo start consentito è lo start event generico. BPMN ammette solo start event di tipo catch. Difatti non ha senso parlare di processi che iniziano con la generazione di eventi (throw).

Esempio 3.5 - Considerando il diagramma posto a sinistra della Figura 3.5, il processo di backup dei dati può iniziare per due motivi: allo scadere della mezzanotte di ogni giorno, oppure a seguito di una richiesta esplicita. Normalmente il backup viene iniziato allo scoccare della mezzanotte, quindi una nuova istanza del processo viene generata. Supponendo che l'attività di backup duri meno di 24 ore, non potranno mai esistere due istanze parallele del processo. Immaginando invece di ricevere una richiesta di backup alle 11.59, il processo ha inizio ma questo non esclude il fatto che a mezzanotte una nuova istanza del processo abbia inizio.

A destra della Figura 3.5 è utilizzato un gateway di tipo *event-based parallel* che definisce anch'esso la regola di istanziazione del processo, imponendo però che entrambe gli eventi siano accaduti poiché questo avvenga: è passata la mezzanotte ed è stata ricevuta una richiesta. Va notato che non necessariamente questi due eventi devono accadere contemporaneamente. L'accadimento di solo uno degli eventi impone che il processo non si istanzi fino all'accademimento del secondo. Ad esempio, se la richiesta di backup avviene prima della mezzanotte, il processo non viene istanziato finché non scatta la mezzanotte.

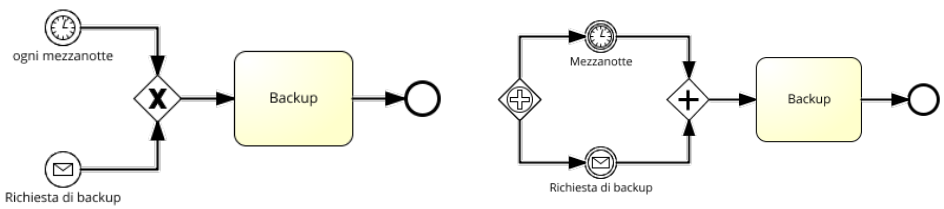


Figura 3.5: *Start event di processo multipli.*

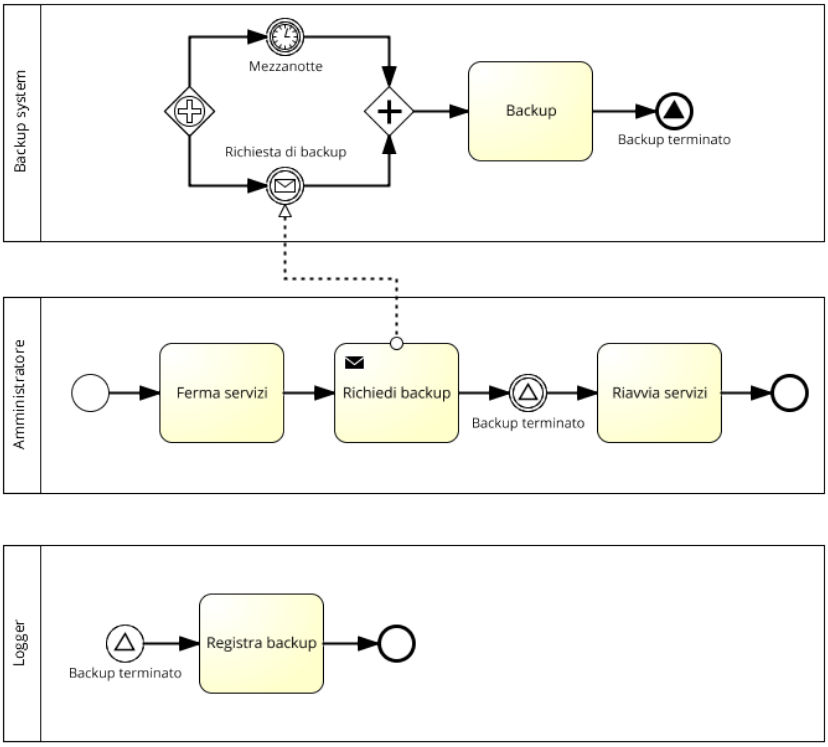


Figura 3.6: *End event con signal throwing.*

3.4 End event

Un end event ha la particolarità di attivarsi quando riceve in ingresso un token e di consumarlo al fine di dichiarare terminato il processo. A seconda della natura dell'evento, appena prima della terminazione del processo, è possibile generare un evento che sarà poi consumato da altri processi. Pertanto, dualmente rispetto a quanto detto per gli start event, non possono esservi end event di tipo catching ma solo di tipo throwing.

Esempio 3.6 - L'esempio di Figura 3.6 arricchisce lo scenario del processo di backup illustrato in precedenza. Una volta terminato il backup viene generato un segnale che sarà gestito dai processi in ascolto per quel determinato segnale. In questo caso i processi interessati sono due: quello di amministrazione che ha fatto richiesta per il backup, e quello di logging che registra l'attività una volta conclusa. Questo esempio mostra anche la differenza tra gli eventi di tipo message (che hanno un origine e una destinazione) e uno di tipo signal (che sono di tipo broadcast).[a]

[a] Questo esempio utilizza il costrutto pool che sarà approfondito nel Capitolo 4 e che permette di associare un processo ad un attore. In questo caso si hanno tre processi gestiti rispettivamente da il backup system, l'amministratore e il logger.

Va sottolineato che l'end event solitamente consuma solo il token che riceve in ingresso. Pertanto se vi sono più rami in esecuzione nel flusso, i restanti rami rimangono attivi. Arrivare quindi ad un end event non necessariamente significa terminare un'istanza del processo. Infatti, una istanza del processo ha termine quando tutti i token generati per quell'istanza sono stati consumati.

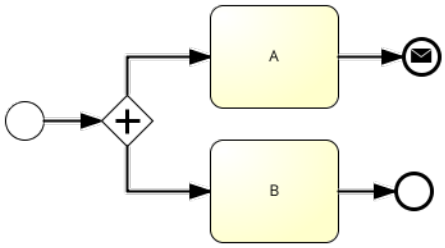


Figura 3.7: *End-event paralleli.*

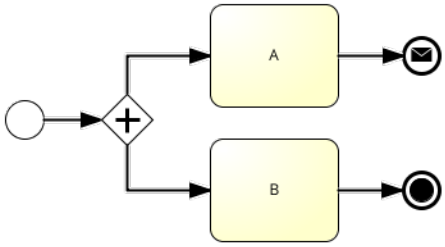


Figura 3.8: *End-event paralleli con terminazione forzata dell'istanza.*

Esempio 3.7 - Nel processo di Figura 3.7, si modella un processo che grazie al gateway parallel presenta due rami concorrenti ognuno dei quali termina in un end event: uno di tipo message, l'altro di tipo none. In questo caso, lo start event genera un token, il gateway parallel lo consuma per generarne due (uno per ramo), e gli end event consumeranno ognuno il token corrispondente al proprio ramo. Ciò vuol dire che il processo termina solo quando l'esecuzione arriva a tutti e due gli end event.

Una variante di questo tipo di comportamento è dettato dall'end event di tipo terminate. Questo end event ha infatti la particolarità di consumare il token che riceve e di terminare l'intero processo. Questo vuol dire che tutte le attività che sono in esecuzione in altri rami sono automaticamente annullate[1].

Esempio 3.8 - La variante dell'esempio precedente, riportata in Figura 3.8, rappresenta un processo che si comporta in modo diverso a seconda del fatto che l'attività A termini prima dell'attività B. Infatti, se l'attività A termina prima l'end event di tipo message consuma il token e invia il messaggio. Una volta che anche l'attività B termina, allora il token consumato dal terminate è l'unico ancora attivo e il processo termina.

Se invece l'attività B ha una durata minore, l'esecuzione del terminate end event porta al blocco immediato dell'attività A e pertanto il messaggio non sarà mai mandato.

[1]Come sarà discusso nella sezione dedicata alle transazioni, l'end event di tipo terminate non genera alcuna richiesta di compensazione

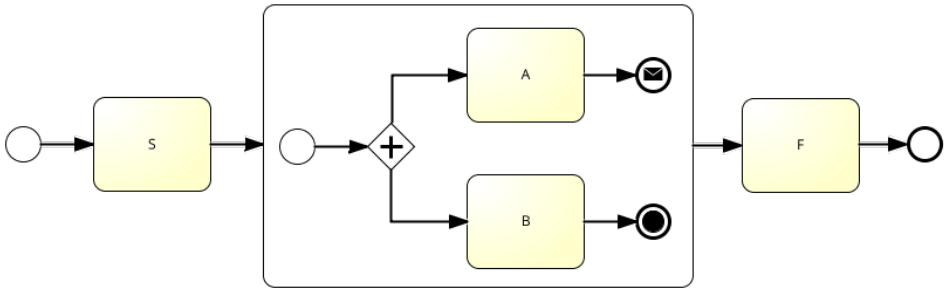


Figura 3.9: *End event terminate all'interno di un sotto-processo.*

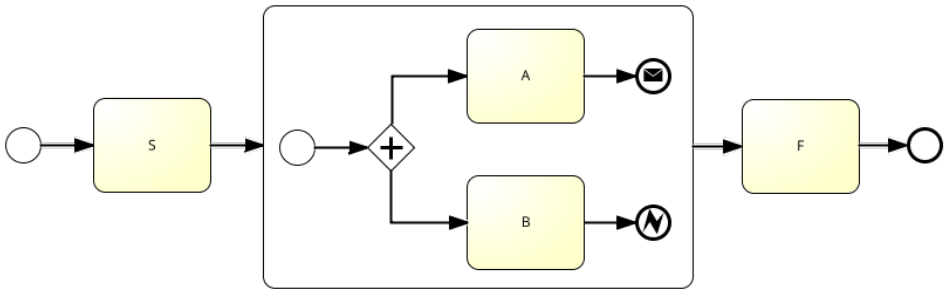


Figura 3.10: *End event error all'interno di un sotto-processo.*

In un processo che include anche sottoprocessi, l'ambito di un evento di end event è limitato al livello di processo in cui è inserito. Questo è particolarmente importante per l'end event di tipo terminate. Infatti, se un end event di tipo terminate è posizionato all'interno di un sottoprocesso, esso indica la necessità di interrompere forzatamente non l'intero processo, ma solo il sotto-processo al quale appartiene.

Esempio 3.9 - L'esempio di Figura 3.9 mostra un processo in cui una attività è definita da un sottoprocesso la cui struttura è identica a quello dell'esempio precedente. In questo caso, una volta eseguito il task S, il sottoprocesso ha inizio e indipendentemente da come esso termina (end event mail o terminate), l'esecuzione riprende poi con il task F.

Sempre nel caso in cui si stia specificando un sottoprocesso, l'end event permette anche di comunicare, al processo di livello superiore, come il sottoprocesso ha avuto termine. Questo è possibile utilizzando gli end event di tipo error, escalation, cancel, compensation, signal.

Esempio 3.10 - Variando leggermente l'esempio precedente, il sottoprocesso inserito nel processo di Figura 3.10 può terminare inviando un messaggio, oppure notificando un errore (invece che con un terminate). Nel secondo caso, la gestione della situazione di errore può essere demandata al processo di più alto livello, come si descriverà meglio nel prossimo paragrafo attraverso l'uso degli intermediate event di tipo boundary.

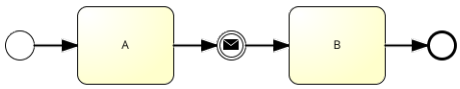


Figura 3.11: *Intermediate event di tipo throwing.*

3.5 Intermediate event

La classe degli intermediate event è molto ricca e permette di gestire molte situazioni che si vengono a verificare durante l'esecuzione di in un business process. Come illustrato in precedenza, la semantica di questo tipo di eventi si distingue a seconda del fatto che l'evento venga posizionato nel flusso di esecuzione o sul boundary di una attività: se l'intermediate event è posizionato sul flusso, la sua semantica impone che il processo continui solo all'accadimento dell'evento; se l'intermediate event è posizionato sul boundary di una attività un exception flow viene attivato e a seconda del fatto che l'event sia di tipo bloccante o meno, l'attività prosegue la sua esecuzione.

Quando un intermediate event è posizionato nel flusso, allora è possibile definire non solo il catching di un evento ma anche la generazione di un evento (throwing). In tal caso, il processo non rimane in attesa dell'accadimento di un evento, bensì l'evento viene generato e successivamente il processo potrà continuare.

> **Esempio 3.11 -** Come mostrato in Figura 3.11, una volta terminata l'attività A viene inviato un messaggio e quindi il processo continua con l'esecuzione dell'attività B. Va sottolineato che l'attività B inizia nel momento in cui il messaggio viene inviato. Non c'è alcun controllo rispetto al fatto che il messaggio venga effettivamente ricevuto prima dell'esecuzione dell'attività B.

Nel caso di intermediate event di tipo boundary, questa classe di eventi è molto utile per modellare la dipendenza tra processi e sottoprocessi. All'interno dei sottoprocessi possono accadere situazioni particolari che non possono essere risolte localmente ma devono essere gestite a livelli più alti. Pertanto, il throwing di eventi (sia di tipo intermediate che di tipo end) a livello di sottoprocesso, possono essere catturati a livello di processo e quindi gestiti. Come detto in precedenza questo è possibile solo per alcuni trigger: error, escalation, cancel, compensation e signal.

> **Esempio 3.12 -** L'esempio di Figura 3.12 completa la modellazione del processo di Figura 3.10 utilizzando un boundary intermediate event in grado di catturare l'eventuale errore generato all'interno del sottoprocesso. A fronte di tale errore, infatti, il sottoprocesso risulta terminato, mentre il processo di livello superiore seguirà il flusso dell'exceptional path che richiede l'esecuzione dell'attività G.

3.6 Event-based exclusive gateway (event-based XOR)

Ora che si è chiarito il significato e il ruolo degli eventi in BPMN, è possibile descrivere in modo compiuto il gateway event-based exclusive citato solo brevemente nel Capitolo 2.

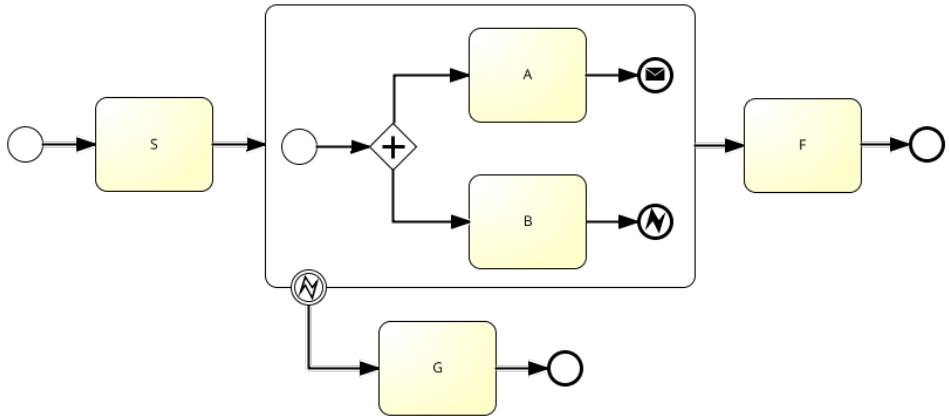


Figura 3.12: *Intermediate event generato all'interno di un sottoprocesso.*

Questo gateway è ammesso in sola modalità split: non è quindi possibile avere più di un flusso entrante al gateway. Inoltre, per ogni flusso in uscita è necessario avere la presenza di un evento di un intermediate event di tipo catching.

Grazie a questa configurazione, all'accadimento di uno degli eventi associati ai flussi in uscita, l'esecuzione presegue per quel determinato percorso. A differenza dello XOR tradizionale, che basa la scelta del flusso in uscita predicando su dati conosciuti, l'event-based XOR, come recita il nome, discrimina tra i flussi in uscita sulla base dell'accadimento di un evento.

Esempio 3.13 - Un esempio di utilizzo dell'event-based XOR è mostrato in Figura 3.13. Il processo modellato prevede che dopo aver inviato una richiesta un altro attore (qui non rappresentato), tale processo rimanga in attesa dell'accadimento di uno dei tre eventi collegati a valle dell'event-based XOR. Infatti, secondo il modello di processo esemplificato, l'attore destinatario della richiesta può reagire in 3 modi: accettare la proposta inviata, rifiutarla, o non rispondere.

Essendo un gateway di tipo esclusivo, solo uno degli eventi può accadere, e il flusso di esecuzione continua solo per il ramo associato a tale evento, mentre gli eventi associati agli altri rami sono invalidati. Quindi a fronte di una accettazione da parte del cliente, il processo memorizza la risposta e il processo si chiude. Se invece il cliente non risponde in alcun modo, il processo si chiude allo scadere del settimo giorno. Supponendo infine che la arrivi una risposta di accettazione all'ottavo giorno, tale richiesta viene semplicemente ignorata in quanto l'esecuzione è già andata oltre al gateway che gestiva gli eventi.

L'exclusive event-based XOR può anche essere utilizzato per discriminare tra gli eventi che scatenano l'inizio di un processo. Ripredendo l'esempio 3.5, a destra della Figura 3.14 si rappresenta un modo alternativo per modellare il processo di sinistra, utilizzando il gateway di tipo *event-based XOR*. Dal punto di vista grafico, va notato che in questo caso, il simbolo di event-based XOR presenta un singolo cerchio al suo interno, e non doppio come nel caso di gestione degli eventi di tipo intermedio. In particolare, il processo modellato inizia nel momento in cui uno dei due eventi posti a valle del gateway accada, per poi eseguire l'attività di backup.

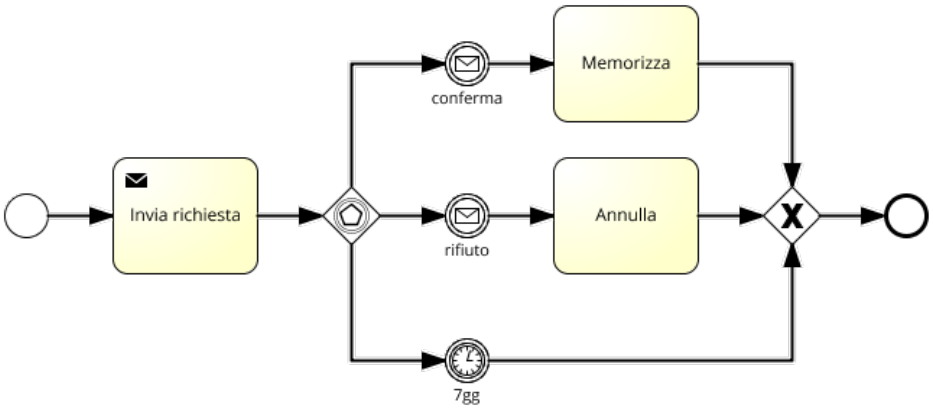


Figura 3.13: *Event-based XOR.*

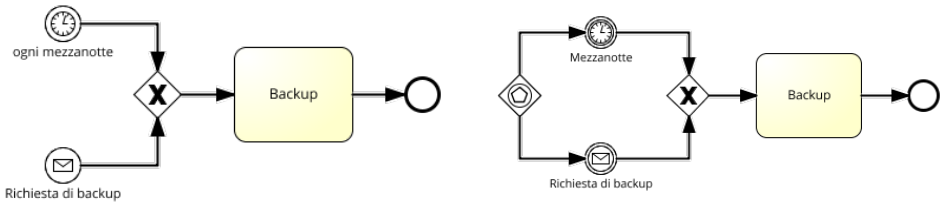


Figura 3.14: *Start event di processo multipli.*

3.7 Eventi bloccanti e non bloccanti

Ortogonalmente alla classificazione start, intermediate, end event, e alla loro tipologia, gli eventi possono essere distinti tra:

 *eventi bloccanti* (il bordo del cerchio è composto da una linea continua)

 *eventi non-bloccanti* (il bordo del cerchio è composto da una linea tratteggiata).

Tale distinzione è possibile solo per gli start event e per gli intermediate event; non ha senso invece parlare di end event di tipo non-bloccante, in quanto sono tutti bloccanti per definizione.

Nel caso degli start event, un evento bloccante o non bloccante distingue il modo in cui, all'interno di un sottoprocesso, diversi flussi possono essere mandati in esecuzione e sono utilizzati all'interno dei cosiddetti event-based sub-process.

Esempio 3.14 - Come riportato in Figura 3.15[a], il sottoprocesso di gestione di prenotazione di un viaggio si attiva all'arrivo dell'ordine e pertanto il flusso di attività corrispondenti alla prenotazione del viaggio ha inizio. Durante l'esecuzione di tale flusso può accadere che arrivi un messaggio di richiesta cancellazione. Questo, es-

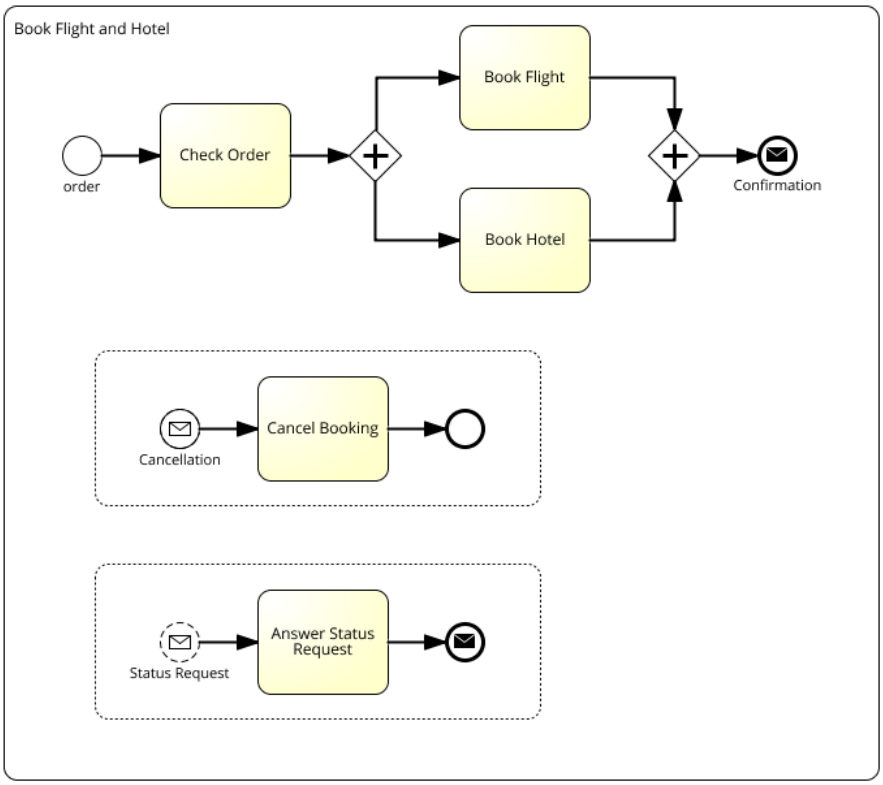


Figura 3.15: *Eventi bloccanti e non bloccanti*

sendo uno start event bloccante, blocca il flusso in corso e attiva il flusso riguardante la cancellazione della prenotazione. Il comportamento è differente nel caso in cui invece arrivi, durante l'esecuzione del flusso di gestione della prenotazione, il messaggio di richiesta di stato. In tal caso, essendo lo start event di tipo non bloccante, attiva il sottoprocesso di notifica dello stato della prenotazione senza interrompere la prenotazione stessa.

[a] Fonte: http://en.bpmn-community.org/tutorials/6/

Va sottolineato che è possibile specificare un trigger per uno start event solo per il processo principale o per un event sub process ma non per un sottoprocesso. Infatti, quando si ha a che fare con un sottoprocesso la sua attivazione deve essere sempre legata alla terminazione di una attività che lo precede a livello superiore. In tal modo è sempre sicuro che gli event sub-process gestiscano situazioni particolari quando il sottoprocesso è già in esecuzione.

Nel caso invece degli intermediate event, solo quelli di tipo boundary possono essere di tipo non-bloccante. Non è possibile quindi definire non bloccante un intermediate event posizionato nel flusso di processo.

Esempio 3.15 - Ad esempio, in Figura 3.16 si ha la stessa struttura dell'esempio 3 dove l'intermediate event non è più bloccante (doppio bordo continuo) bensì non bloccante (doppio bordo tratteggiato). Anche in questo caso, essendo l'evento di ti-

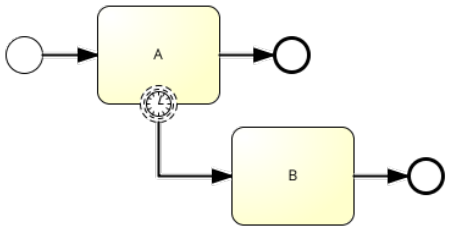


Figura 3.16: *Intermediate event non bloccante su boundary*

po boundary, si vengono a definire il normal flow e l'exceptional flow. A differenza però dall'Esempio 3, l'evento non bloccante indica che allo scadere del timeout sia il normal flow che l'exceptional flow rimangono attivi e pertanto l'esecuzione dell'attività B non esclude la conclusione dell'attività A. Ovviamente nel caso l'attività A termini prima dello scadere del timeout, l'exceptional flow non sarà mai attivato.

A conclusione di questa panoramica sugli eventi, la Figura 3.17 [2] schematizza tutti e soli gli eventi che si possono inserire all'interno di un diagramma BPMN.

E' importante sottolineare che non tutti i tipi di eventi possono essere legati a tutti i trigger. Ci sono situazioni in cui alcuni eventi non possono essere attivati da specifici trigger. Ad esempio, il trigger di tipo timer ha senso per gli start e intermediate event ma non per gli end event. Al tempo stesso, un trigger di tipo Cancel non ha motivo d'esistere quale generatore di processi.

Esempio 3.16 - Come esempio conclusivo si riprende il caso fornito dalle specifiche BPMN e mostrato in Figura 3.18 [a]. A sinistra vi è la porzione del processo che definisce la politica di acquisto di merce, mentre a destra le attività di acquisto vere e proprie. Nel momento in cui è richiesta l'attività di Procurement, se l'attività non riscontra errori nella sua esecuzione il processo termina correttamente. Se invece la merce che si vuole acquistare risulta essere indisponibile allora l'articolo viene rimosso dal catalogo. Questa situazione eccezionale è modellata intercettando l'evento *undeliverable* di tipo error. Vi è un terzo caso che modella la situazione in cui accade un evento (di tipo escalation) che non influisce sulla corretta terminazione del processo, essendo di tipo non bloccante. Questo evento gestisce infatti il caso in cui l'articolo da acquistare risulta disponibile ma con tempi non brevi.

Seguendo una scomposizione gerarchica, la parte destra della Figura 3.18 mostra il dettaglio dell'attività di Procurement dove sono modellati gli eventi poi gestiti a livello superiore. Infatti a fronte di una verifica con il fornitore sulla disponibilità della merce, si possono verificare le seguenti situazioni: (i) essere disponibile in tempo ragionevole (massimo 2 giorni), (ii) essere disponibile con un po' di ritardo, (iii) non essere disponibile. Nel secondo e terzo caso, i due eventi di tipo throwing informano il livello superiore sulla situazione. In particolare, l'evento di errore è di tipo end e pertanto conclude l'attività. Questo è consistente rispetto al fatto che l'esecuzione a livello di processo porti all'esecuzione dell'exceptional flow, escludendo la possibilità che il processo continui dopo l'attività di Procurement. Al contrario, l'evento

[2] Fonte: http://blog.goodelearning.com/bpmn/common-bpmn-modeling-mistakes-best-practices-basic-events/

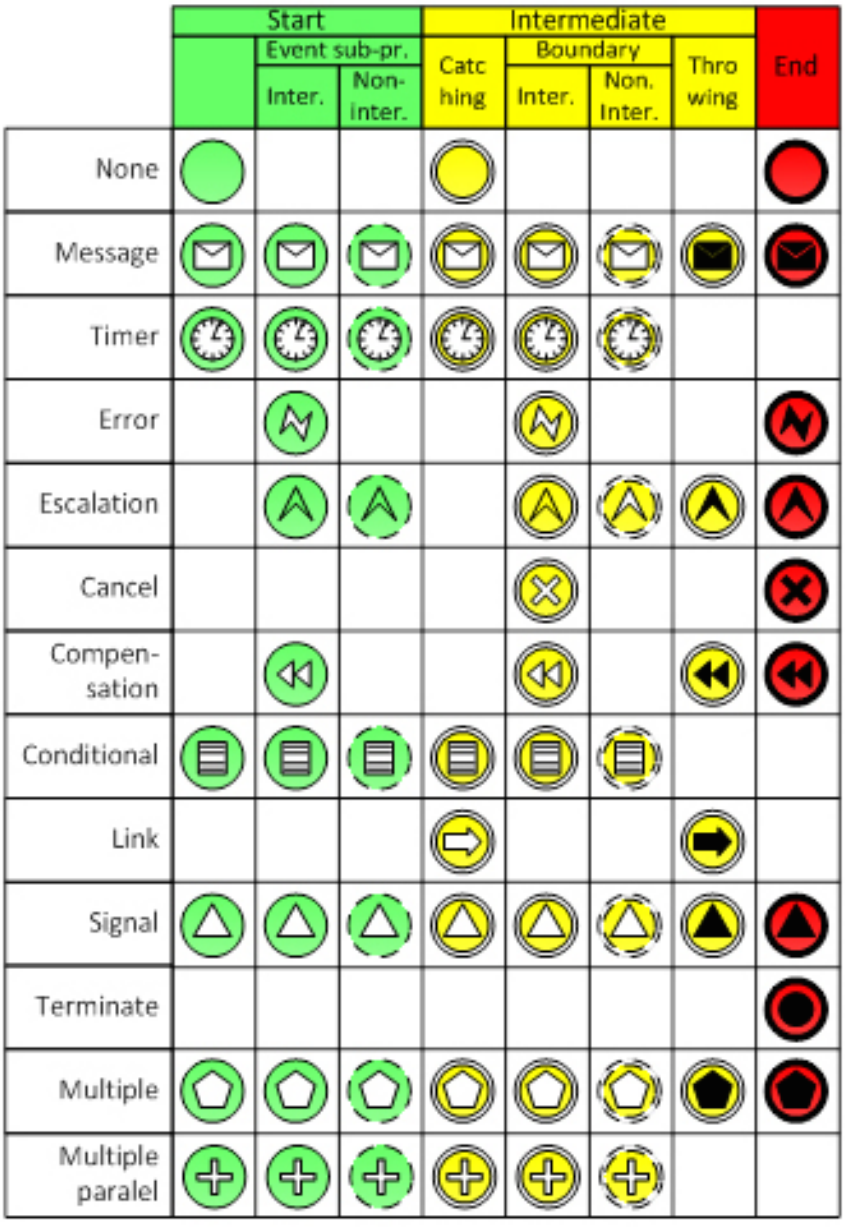


Figura 3.17: *Tipologia di eventi ammessi*

intermediate di tipo escalation informa il processo della situazione non ottimale (*late delivery*) ma l'attività continua comunque. Ciò è correttamente gestito a livello di processo attraverso l'intermediate event di tipo boundary non bloccante che attiva il task per informare il cliente sul ritardo ma non chiude il processo. L'end event infatti che segue l'attività *inform customer* chiude solo il flusso derivato dall'attività di procurement che, al suo interno, sta continuando la sua esecuzione.

[a] Esempio tratto da "Object Management Group, BPMN 2.0 by Example, June 2010, `http://www.omg.org/spec/BPMN/20100601/10-06-02.pdf`

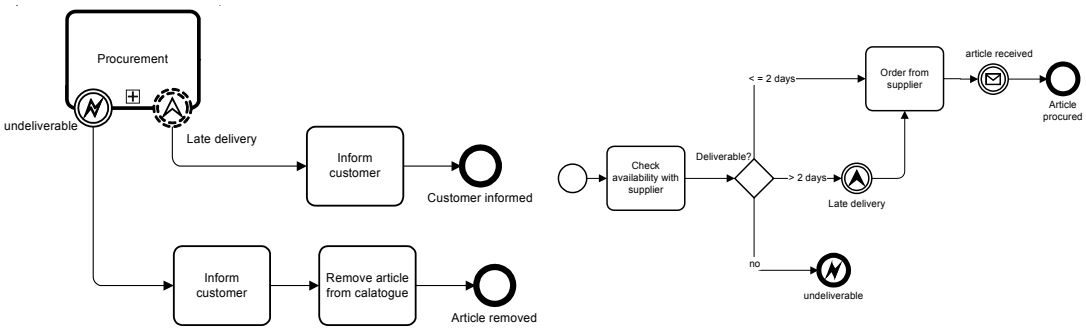


Figura 3.18: *Processo di gestione stock e sottoprocesso di procurement.*

4 Collaborazione tra processi: pool e lane

I costrutti BPMN e gli esempi proposti finora si sono focalizzati sulla modellazione di un singolo processo. Questo processo è costituito da diverse attività o declinato in sottoprocessi. Attraverso i gateway è possibile organizzare il flusso di controllo e catturare o generare eventi.

Spesso però la definizione di un singolo processo non è sufficiente per modellare un business process aziendale: va tenuto conto del contesto in cui opera. Tale contesto è definito da altri processi che altre organizzazioni eseguono e con cui il processo che si sta definendo spesso si scambia beni e informazioni.

4.1 Collaboration e pool

Secondo la terminologia BPMN, la modellazione di processi interdipendenti è detta *collaboration*. Come definito dalla specifica BPMN, un collaboration diagram è definito da due o più partecipanti, dal flusso di messaggi tra i partecipanti ed, eventualmente, dai processi svolti dai partecipanti. Un partecipante è quindi una entità che gode di una sua autonomia e che collabora nell'esecuzione di un processo e che, in BPMN, è rappresentato da un *pool*. Un pool è rappresentato da un elemento rettangolare che si sviluppa orizzontalmente o verticalmente nel diagramma. Un pool può essere rappresentato come una *white box* o una *black box*. Nel primo caso, all'interno del pool è inserito il processo che è responsabilità del partecipante eseguire e controllare. Si suppone quindi che il partecipante rappresenti anche l'orchestratore di tale processo. Nel secondo caso invece, all'interno del pool non è rappresentato nulla. Questo indica che non si conosce, o non è importante sapere ai fini della modellazione del business process, le attività che un partecipante svolge.

Definiti i partecipanti attraverso i loro pool, la collaborazione vera e propria tra i partecipanti avviene nel momento in cui vi è uno scambio di *messaggi* tra essi. Tali messaggi indicano lo scambio di beni e informazioni tra i partecipanti e sono indicate da una freccia tratteggiata in cui un terminale è a forma circolare, mentre l'altro terminale è a forma triangolare.

Esempio 4.1 - Si supponga di voler modellare il processo di prenotazione di una vacanza ipotizzando di appoggiarsi ai servizi forniti da una agenzia viaggi. La Fi-

gura 4.1 mostra il collaboration diagram che comprende due partecipanti, il cliente e l'agenzia viaggi, ognuno rappresentato con il proprio pool. Il cliente, di cui si sta modellando il processo, è ulteriormente definito dalla sequenza di attività che esso svolge. Dell'agenzia viaggi invece non è necessario, o non si è a conoscenza, delle attività. Ciò che è importante ai fini della modellazione è sapere che tra cliente e agenzia viaggi vi è uno scambio di 3 messaggi in un preciso ordine: data di viaggio (da cliente a agenzia), elenco di opzioni (da agenzia a cliente) e, infine, di scelta dell'opzione (da cliente a agenzia).

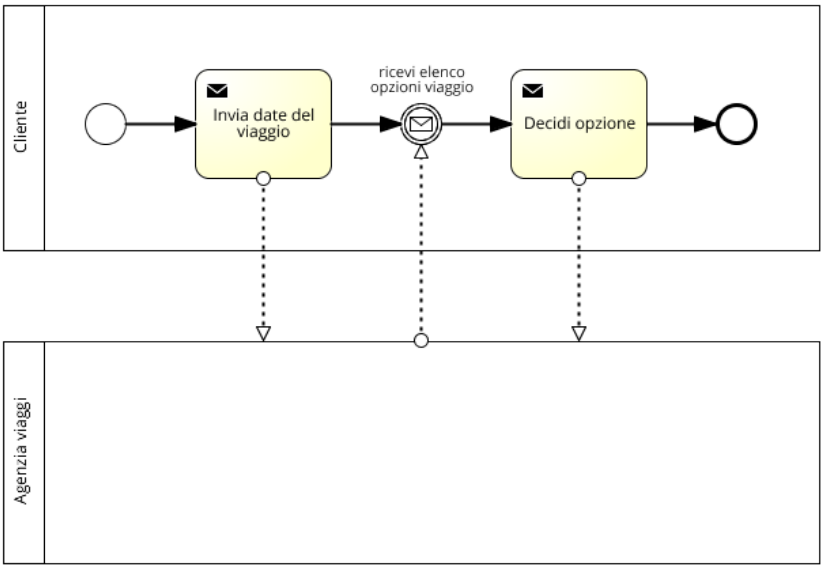


Figura 4.1: *Collaboration diagram*

Come evidenziato dall'esempio, lo scambio di messaggi è strettamente legato al concetto di evento di tipo messaggio descritto nella sezione precedente. Un messaggio inviato da un pool e atteso da un altro pool può essere modellato come un intermediate event posizionato sul control flow. Lo scambio dei messaggi ha inoltre impatto sulla natura delle attività. Infatti quando le attività generano messaggi, come nel caso di *Invia date del viaggio* e *Decidi opzione* queste possono essere modellate come attività di tipo *Send Message Task*. È ovviamente possibile utilizzare, in modo del tutto equivalente, al posto del message intermediate event di tipo catch *Ricevi elenco opzioni viaggio* una attività di tipo *Receive Message Task*.

4.2 Processi privati e processi pubblici

L'esempio appena proposto permette di introdurre il concetto di *processo privato* e *processo pubblico*. Il processo privato riguarda le attività, opportunamente organizzate in un flusso di controllo, che sono interne e controllate da un determinato partecipante. Questo processo è il processo che il partecipante deve eseguire e che orchestra. Il processo pubblico, invece, è composto da un processo privato e dall'interazione che esso ha con altri processi. Nel caso dell'esempio di Figura 4.1, il processo privato

è costituito da quanto contiene il pool cliente, mentre il processo pubblico è l'intero collaboration diagram.

Esempio 4.2 - Considerando ancora l'esempio precedente, si consideri il processo di Figura 4.2. In questo caso si è voluto modellare anche il processo privato dell'agenzia viaggi. In questo modo è anche possibile sapere chi consuma e produce i messaggi scambiati anche lato-agenzia. Si hanno quindi due processi privati (interni ai pool) e due processi pubblici (considerando lo scambio di messaggi e, in un caso il processo privato del cliente, e nel secondo caso il processo privato dell'agenzia).

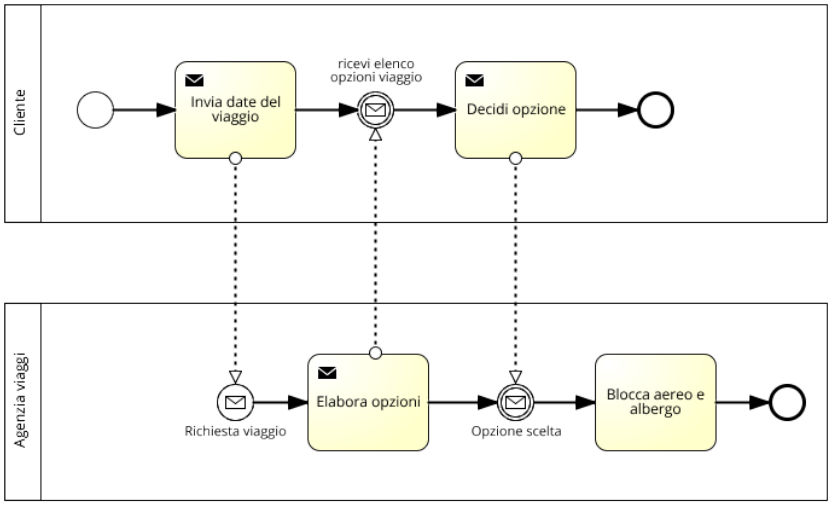


Figura 4.2: *Collaboration diagram con pool white-box*

Per meglio enfatizzare quello che è il processo privato modellato in un diagramma, il partecipante che svolge tale processo può anche non essere inserito all'interno di un pool.

Esempio 4.3 - Ad esempio, il diagramma di Figura 4.3 mostra sempre il processo di prenotazione viaggio in cui le attività del cliente non sono raccolte all'interno di un pool. Questo indica che il modellatore ha preso il punto di vista del cliente e sta considerando il processo del cliente come il processo privato. Questo significa che, sebbene anche il pool dell'agenzia viaggi sia una white-box, non necessariamente il processo inserito è il vero processo interno all'agenzia. Potrebbe infatti essere una sua semplificazione in cui sono modellate le sole attività necessarie a meglio specificare lo scambio dei messaggi con il cliente.

Va sottolineato che in un collaboration diagram solo un processo privato può non essere inserito all'interno di un pool.

Essendo un pool la rappresentazione concettuale di un partecipante al processo - che pur collaborando con altri partecipanti ha una propria autonomia - non è ammesso che il flusso di controllo di un processo superi i confini di un pool. L'unico modo per mettere in comunicazione pool differenti è utilizzando i messaggi. Questa restrizione trova senso nella semantica del control flow associato ad un processo. Esso infatti definisce il

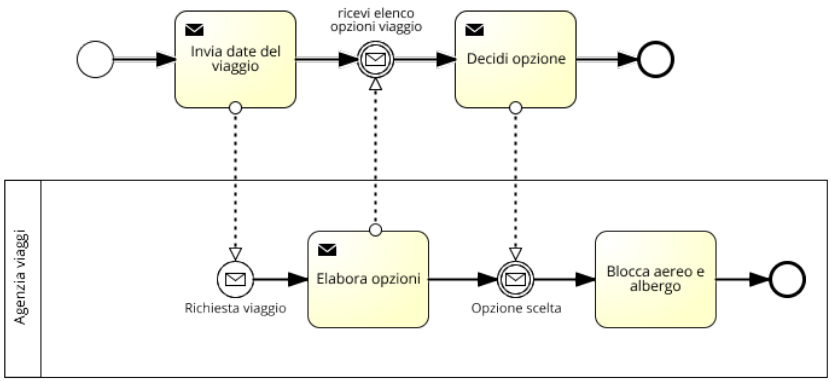


Figura 4.3: *Collaboration diagram con enfasi sul processo privato del cliente*

flusso che deve essere controllato e gestito da un orchestratore. Pertanto, si suppone che all'interno di ogni partecipante al processo esista un orchestratore in grado di decidere come e quando eseguire le attività del processo sulla base di quanto definito nel control flow. Quindi permettere di unire attraverso un flusso di controllo due partecipanti significa permettere ad un orchestratore di uno dei partecipanti di "comandare" all'interno di un altro partecipante. Per la autonomia dei partecipanti questo non è ammesso. Al contrario, quando un partecipante richiede la collaborazione di un altro partecipante, lo strumento messo a disposizione è quello del messaggio. Un partecipante invia un messaggio ad un altro partecipante implicitamente informandolo sullo stato di esecuzione del suo processo ed eventualmente richiedendo informazioni di risposta.

4.3 Multi-instance pool

Così come per le attività e i sottoprocessi, in BPMN è possibile definire pool di tipo multi-instance. Il marker utilizzato per questo scopo è il medesimo utilizzato per le attività (tre linee verticali parallele). Questo indica la possibilità di specificare interazioni con partecipanti diversi il cui ruolo nei confronti del processo che si sta modellando è identico, così come il flusso dei messaggi che supportano.

Esempio 4.4 - Si supponga che la prenotazione di un viaggio da parte di un cliente non veda l'interazione di quest'ultimo con un'unica agenzia ma con diverse agenzie. Il cliente infatti propone le date di viaggio a diverse agenzie e, dopo aver selezionato l'offerta migliore conclude la transazione solo con una di esse. Il processo corrispondente è illustrato in Figura 4.4 dove le attività del cliente di invio date e ricezione opzioni sono state raggruppate funzionalmente in una macro-attività che, per supportare la comunicazione con le diverse agenzie viaggio, è anch'essa di tipo parallel multi-instance.

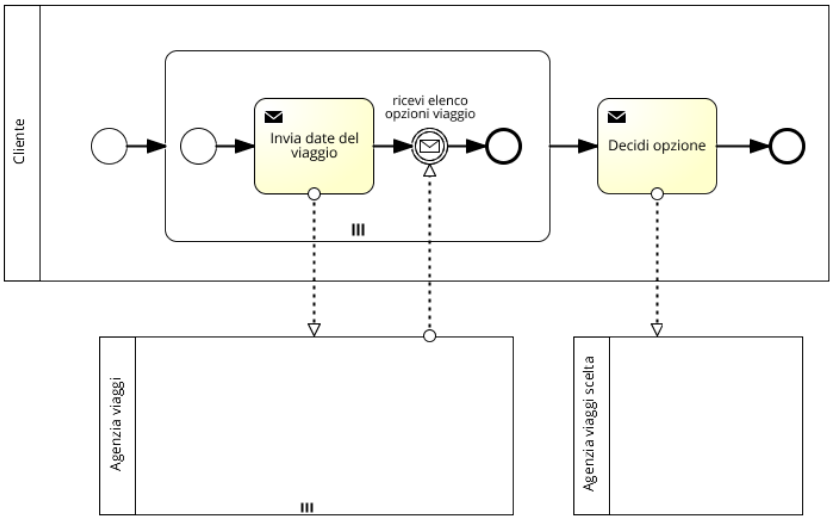


Figura 4.4: *Pool di tipo multi-instance*

4.4 Lane

Dato un pool, BPMN permette di specificare ulteriormente le caratteristiche del partecipante corrispondente attraverso il costrutto *lane*. Dal punto di vista grafico una lane ha la stessa forma del pool. Dal punto di vista semantico, una lane rappresenta un ruolo o uno specifico attore appartenente ad un partecipante. Per questo motivo una lane è un elemento inserito all'interno di un pool.

Esempio 4.5 - Supponendo di voler meglio specificare il processo interno dell'agenzia viaggi, il diagramma di Figura 4.5 mostra la suddivisione del relativo pool in due lane. Le due lane si riferiscono alle due sotto organizzazioni dell'agenzia viaggi. La prima, i.e., l'ufficio commerciale, si occupa della gestione della relazione con il cliente e della prenotazione alberghiera; la seconda, i.e., la divisione voli, si occupa solo della prenotazione aerea.

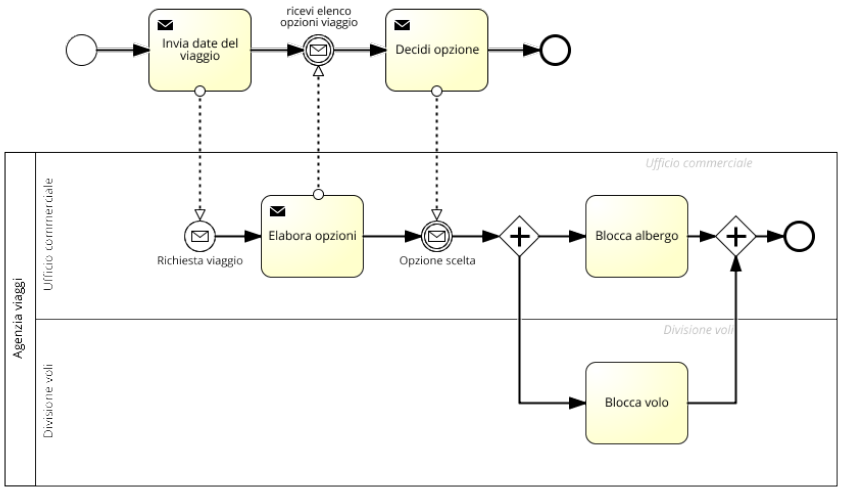


Figura 4.5: *Utilizzo delle lane*

5 Gestione dei dati con BPMN

Questo capitolo si focalizza sulla rappresentazione dei dati in BPMN. In particolare, si presentano qui i costrutti utili per modellare il flusso di dati. Detti costrutti sono utili per evidenziare su quali dati si lavora all'interno di un'attività e/o quali dati sono oggetto di una comunicazione tra due attività.

5.1 Data Modeling

La modellazione dei processi deve consentire a chi progetta il processo di specificare come vengono creati, aggiornati e cancellati i dati utilizzati nello stesso. Per questo motivo BPMN mette a disposizione diversi costrutti: Data Object, Data Store, Message e Data Association.

Un *Data Object* rappresenta l'informazione utilizzata all'interno di un processo come un documento di business, una mail o altre comunicazioni. Un Data Object deve essere definito all'interno di un processo o un sottoprocesso. Se si vuole rappresentare un insieme di informazioni (es. un insieme di ordini) allora si può utilizzare il construtto di *Collection Data Object* (Figura 5.1).

Un *Data Store* fornisce alle attività di un processo un meccanismo per ritrovare e aggiornare informazioni che sono memorizzate nel sistema e che hanno un ciclo di vita più lungo di quello del processo in cui sono utilizzate. Solitamente un Data Store viene utilizzato per rappresentare una unità informativa (es. record di un database).

La notazione per utilizzare questo concetto in BPMN é riportata in Figura 5.2. Un costrutto di Data Association è quello che lega i Data Object o Data Store a un'attività o un evento e può essere rappresentato dal connettore in Figura 5.3.

Per rappresentare il fatto che un documento è scambiato tra due attività è possibile utilizzare una doppia notazione come mostrato in Figura 5.4.

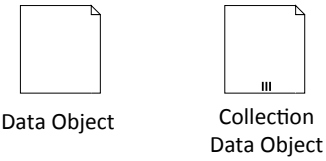


Figura 5.1: *Costrutti per specificare Data Object*

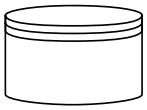


Figura 5.2: *Rappresentazione di un Data Store*

Figura 5.3: *Costrutto per definire l'associazione di Data Object o Data Store a un'attività o un evento*

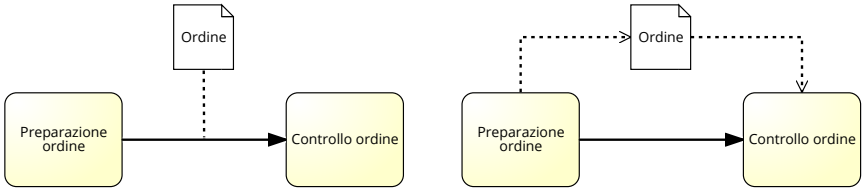


Figura 5.4: *Possibili notazioni per rappresentare lo scambio di un documento tra due attività*

I costrutti di Data Association sono anche utilizzati per legare un Data Store a una attività e quindi per modellare l'accesso a database. Per rappresentare le diverse operazioni che un'attività può eseguire su un database si deve utilizzare la notazione illustrata in Figura 5.5.

Esempio 5.1 - Si supponga di dover modellare il processo per la produzione su commessa di un divano. Il processo inizia con l'invio dell'ordine da parte del cliente. L'Ufficio Commerciale dell'azienda che riceve l'ordine ricava la distinta base e verifica se i materiali necessari siano tutti disponibili. Se sono disponibili procede con la produzione del divano altrimenti l'Ufficio Commerciale manda le schede dei prodotti da ordinare all'ufficio acquisti. Quando tutti i materiali sono arrivati si procede con la produzione. Il processo corrispondente è illustrato in Figura 5.6 dove, tramite i costrutti di Data Object e Collection Data Object si evidenzia il fatto che l'ufficio Commerciale riceve il documento dell'ordine e che dopo l'analisi dello stesso eventualmente manda all'Ufficio Acquisti un insieme di schede relative ai materiali non disponibili.

Esempio 5.2 - Si supponga di dover modellare il processo per la richiesta di un libro in biblioteca. Il processo inizia con il cliente che chiede la possibilità di prendere un libro in prestito al bibliotecario. Il bibliotecario controlla la disponibilità del libro accedendo al database dei prestiti. Se il libro non è disponibile il processo termina. In caso contrario il bibliotecario va a prendere il libro dallo scaffale e poi registra il prestito. Per la registrazione del prestito il bibliotecario accede al database dei clienti della biblioteca per recuperare i dati del cliente che fa la richiesta e poi registra il prestito nel database dei prestiti. Il processo è illustrato in Figura 5.7.

Nel diagramma di un processo, oltre ad un'indicazione generica dei dati manipolati,

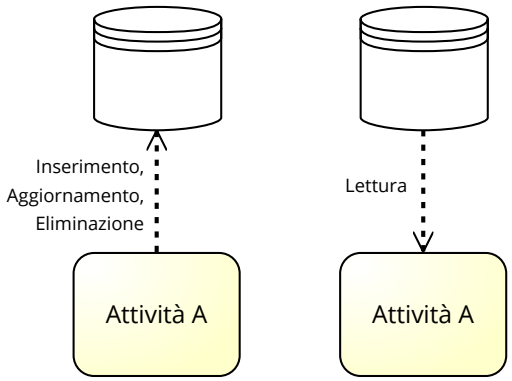


Figura 5.5: *Notazioni per rappresentare le diverse operazioni che un'attività può compiere su un data store*

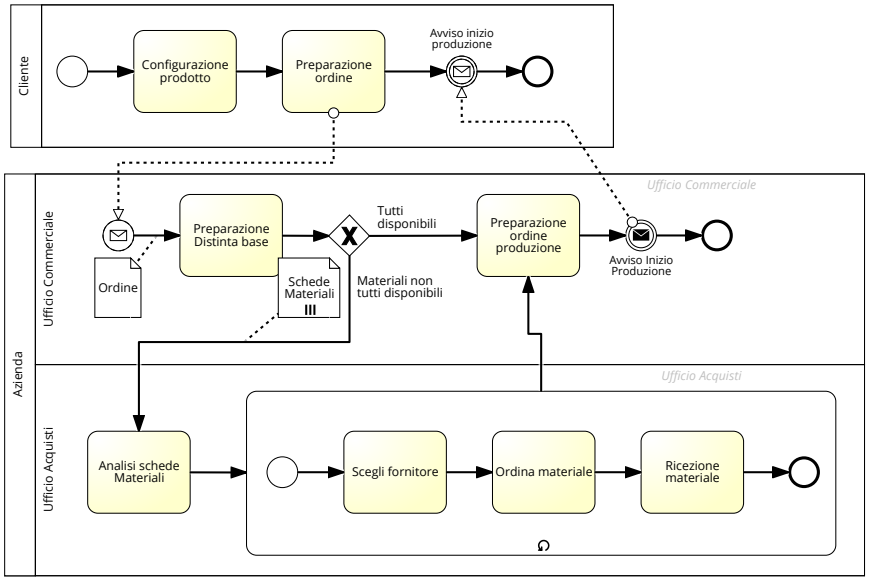


Figura 5.6: *Esempio con utilizzo di costrutti Data Object e Collection Data Object*

è possibile anche specificare gli insiemi di dati che sono input o output di una determinata attività o di un determinato processo. Questo si può fare utilizzando i costrutti di Data Input e Data Output. Un *Data Input* specifica i dati da utilizzare in input ad una certa attività. Un *Data Output* specifica i dati che costituiscono l'output di una certa attività. Graficamente questi costrutti vengono differenziati come illustrato in Figura 5.8. I pattern di interazione con i dati descrivono le modalità con cui i dati possono essere passati tra attività e processi. Infatti, i dati possono essere passati tra attività dello stesso processo, tra attività e sottoprocessi dello stesso processo e anche tra attività di processi diversi. I dati possono essere anche oggetto di comunicazione tra il processo di business e il sistema di gestione tra processi.

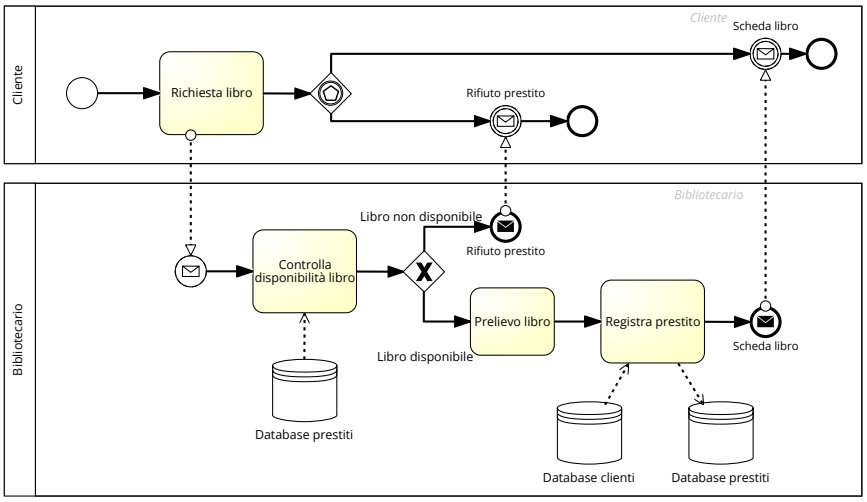


Figura 5.7: *Processo relativo alla richiesta di prestito di libri - esempio in cui si utilizzano i Data store*

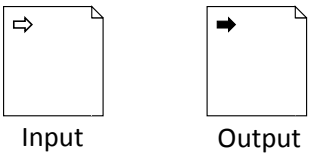


Figura 5.8: *Costrutti per specificare i Data Input o Data Output*

Esempio 5.3 - Si supponga di dover modellare il processo per la richiesta della carta di identità. Nel momento in cui il cittadino si presenta allo sportello del comune deve compilare la richiesta per la carta di identità. L'addetto allo sportello ricevuta la richiesta compilata deve controllare se le informazioni contenute nell'archivio dell'anagrafe sono da aggiornare o meno. Se i dati sono da aggiornare l'addetto procede all'inserimento delle modifiche nel sistema. Una volta terminata l'eventuale attività di modifica l'addetto prepara la carta di identità. Il processo corrispondente è illustrato in Figura 5.9 dove, si evidenzia che la richiesta compilata è un documento di input mentre la carta di identità è un documento di output.

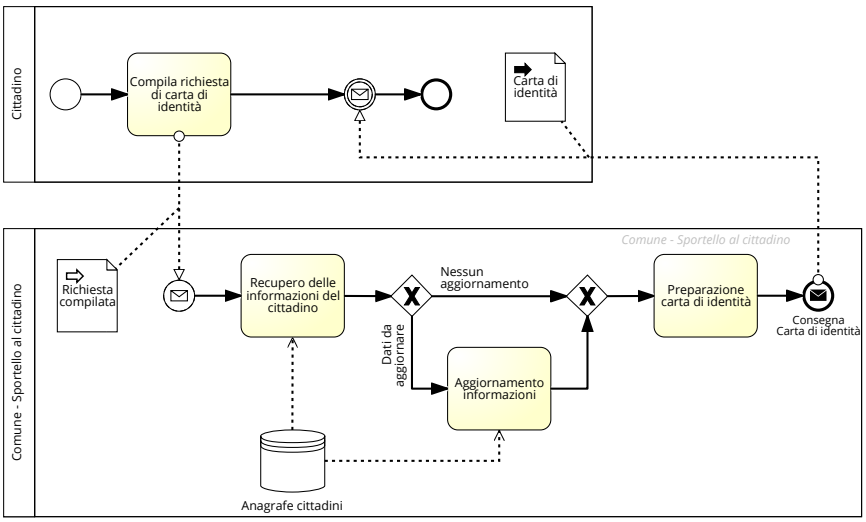


Figura 5.9: *Processo per la richiesta della carta di identità - esempio in cui si utilizzano i costrutti di Data Input e Data Output*

6 Transazioni e compensazione

Questo capitolo si focalizza su aspetti avanzati di BPMN. In particolare, si presenta qui come modellare le transazioni, il flusso di compensazione e la cancellazione.

6.1 Compensation handling

La compensazione rientra tra le procedure di gestione delle eccezioni. Viene usata per fare il rollback delle attività di processo già **completate**. Questo tipo di attività è necessaria tutte le volte che, in caso di fallimento di un processo, debba essere ripristinato lo stato iniziale del sistema. Ad esempio nel momento in cui si prenota un posto su un volo o un treno, il posto viene bloccato in attesa del completamento dell'acquisto. Nel momento in cui, per esempio la carta di credito inserita non è valida e il processo non può essere portato a termine, il posto riservato deve essere lasciato libero e di conseguenza è necessario fare il rollback della transazione di prenotazione.

In BPMN la compensazione è gestita tramite compensation throwing event e compensation handler. I *compensation throwing event* (vedi Figura 6.1) sia in versione intermediate sia in versione end generano l'evento di compensazione mentre il compensation handler gestisce l'evento e deve essere sempre associato all'attività da compensare.

Figura 6.1: *Compensation Throwing event*

In dettaglio, per effettuare il rollback, si usa un *compensation handler* che è attivato tramite il Compensation Catching event (vedi Figura 6.2).

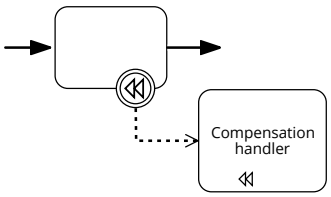


Figura 6.2: *Compensation Handler*

Esempio 6.1 - La Figura 6.3 mostra come sia possibile dichiarare che una prenotazione del posto in treno può essere compensata usando un'attività di "cancellazione prenotazione".

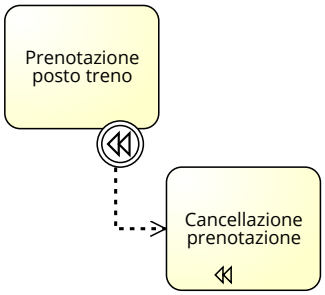


Figura 6.3: *Uso dei compensation Handler*

Esempio 6.2 - La Figura 6.4 mostra come si utilizzano compensation event e compensation handler per richiedere il rollback di un'attività di un processo. Nello specifico esempio, si procede alla prenotazione di un posto in treno e al successivo pagamento. Se l'attività di pagamento non va a buon fine si richiede il rollback della prenotazione in modo che il posto riservato venga reso nuovamente disponibile.

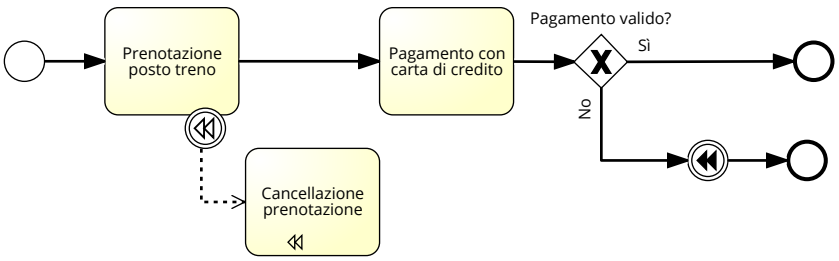


Figura 6.4: *Esempio di processo in cui si richiede il rollback di un'attività utilizzando compensation event e handler*

6.2 Transazioni

Le transazioni possono essere utilizzate per raggruppare diverse attività in un'unità logica di lavoro che deve essere eseguita secondo una logica "all-or-nothing". Questo è un concetto noto nel mondo delle basi di dati in cui le transazioni devono essere eseguite seguendo i principi delle proprietà ACID: atomicità, consistenza, isolamento e durabilità (o persistenza). Le transazioni possono essere definite anche a livello di processi di business. Un insieme di attività che formano una transazione di business devono essere tutte eseguite con successo altrimenti nessuna attività deve essere eseguita: questo garantisce l'atomicità della transazione. Le transazioni di business possono durare giorni o settimane e per questo motivo non si possono gestire adottando le tecniche normalmente utilizzate nell'area delle basi di dati (es. lock delle risorse). Le transazioni a livello di processo di business si gestiscono con la definizione di un sotto processo transazionale, l'ausilio delle eccezioni, eventi di cancellazione e costrutti di compensazione. Ci si può trovare in tre diverse situazioni:

1. **esecuzione riuscita**: la transazione è eseguita con successo e il processo continua seguendo il suo flusso normale;
2. **esecuzione fallita** (evento di cancellazione): quando una transazione viene cancellata è necessario ripristinare le attività che sono già state completate. In questo caso **vengono attivate le attività di compensazione** associate ai task che hanno apportato variazioni allo stato del sistema, cercando, per quanto possibile di riportare il sistema allo stato iniziale.
3. **esecuzione interrotta**: l'esecuzione è interrotta da un errore inaspettato e non è possibile procedere con la normale esecuzione o cancellazione. In questo caso, le attività sono interrotte **senza compensazione** e il flusso continua dall'evento errore definito.

Dal punto di vista della notazione le attività che fanno parte della transazione devono essere raccolte in un sottoprocesso che ha il bordo caratterizzato da una doppia linea (vedi figura 6.5). Gli eventi di errore e cancellazione devono essere attaccati al processo e modellati in modo che il flusso del processo sia gestito in maniera adeguata anche in seguito al fallimento di una transazione.

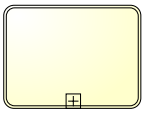


Figura 6.5: *Simbolo di sottoprocesso di natura transazionale*

Esempio 6.3 - La Figura 6.6 mostra come si può abilitare la compensazione utilizzando le transazioni e l'evento di cancellazione. In questo casi si attiva la cancellazione (e quindi la compensazione) se non va a buon fine il pagamento con la carta di credito. L'attività da compensare riguarda la prenotazione del posto: in fase di compensazione il posto bloccato viene rilasciato e reso disponibili per altre istanze

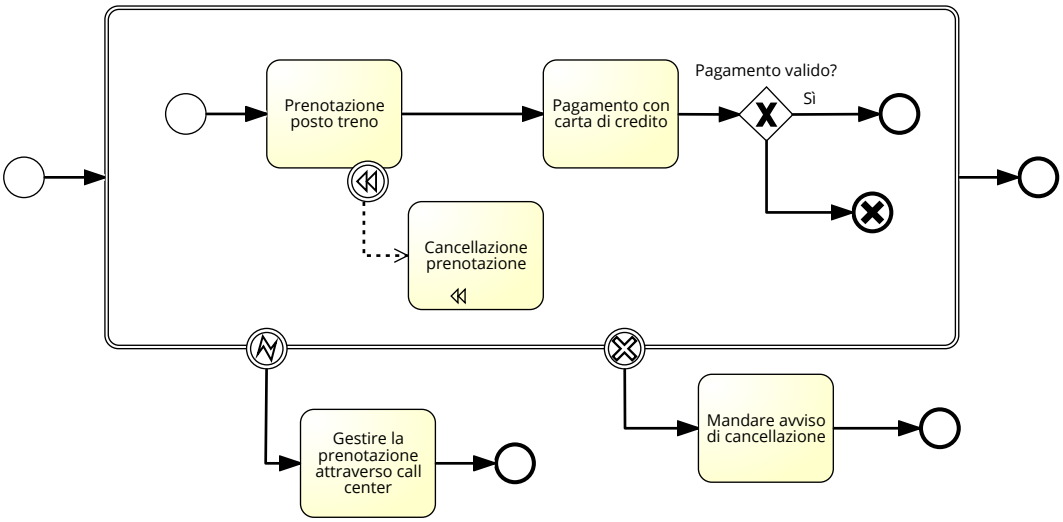


Figura 6.6: *Esempio di transazione*

di prenotazione. In caso di errore, la prenotazione deve essere gestita dal call center; nessuna attività di compensazione viene iniziata.

7 Esempi di riepilogo

7.1 PalmaViaggi s.r.l.

Modellare utilizzando BPMN il seguente processo di prenotazione di un pacchetto viaggio.

Il processo inizia quando l'utente, impiegato della società TalDeiTali s.r.l., accede al sito di prenotazioni on line della società PalmaViaggi s.r.l. e indica le date del suo viaggio. La società, ricevute le date, ricerca nel suo database tutte le possibili soluzioni di viaggio e le propone al cliente che ne seleziona una. Selezionato il viaggio, la società propone una lista di hotel eventualmente prenotabili nella città di arrivo. Il cliente può decidere se prenotare o meno uno degli hotel proposti o se rifiutare la richiesta. Terminata questa fase, la PalmaViaggi invia un riepilogo delle prenotazioni e si procede al pagamento. Questo viene effettuato dalla parte amministrativa della società TalDeiTali. Ricevuto il pagamento, la PalmaViaggi procede a finalizzare le due prenotazioni di volo e hotel. Le prenotazioni sono svolte in maniera contemporanea (la prenotazione dell'hotel deve essere effettuata solo nel caso in cui quest'ultimo è stato selezionato). Finalizzate le prenotazioni, la PalmaViaggi invia conferma dell'itinerario.

Il processo deve rispettare il seguente vincolo temporale: il pagamento deve essere ricevuto entro 5 minuti dalla ricezione dall'invio del riepilogo delle informazioni. In caso contrario, la PalmaViaggi cancella il processo e notifica la cancellazione della procedura di prenotazione (Figura 7.1).

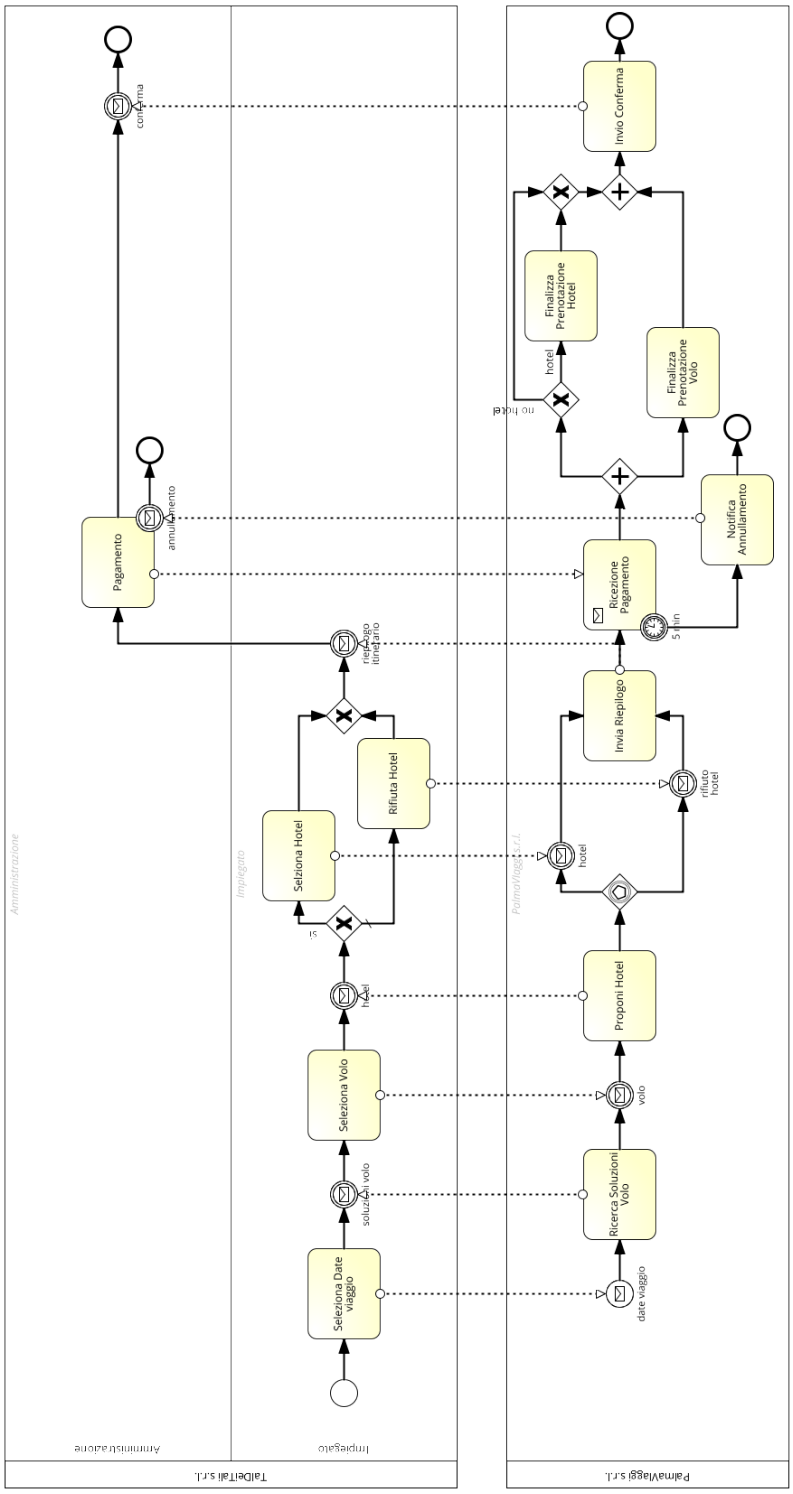


Figura 7.1: *PalmaViaggi s.r.l. - timer pagamento*

7.2 PalmaViaggi s.r.l. - variazione

Modellare utilizzando BPMN il seguente processo di prenotazione di un pacchetto viaggio.

Il processo inizia quando l'utente, impiegato della società TalDeiTali s.r.l., accede al sito di prenotazioni on line della società PalmaViaggi s.r.l. e indica le date del suo viaggio. La società, ricevute le date, ricerca nel suo database tutte le possibili soluzioni di viaggio e le propone al cliente che ne seleziona una. Selezionato il viaggio, la società propone una lista di hotel eventualmente prenotabili nella città di arrivo. Il cliente può decidere se prenotare o meno uno degli hotel proposti. Terminata questa fase, la PalmaViaggi invia un riepilogo delle prenotazioni e si procede al pagamento. Questo viene effettuato dalla parte amministrativa della società TalDeiTali. Ricevuto il pagamento, la PalmaViaggi procede a finalizzare le due prenotazioni di volo e hotel. Le prenotazioni sono svolte in maniera contemporanea (la prenotazione dell'hotel deve essere effettuata solo nel caso in cui quest'ultimo è stato selezionato). Finalizzate le prenotazioni, la PalmaViaggi invia conferma dell'itinerario.

Il processo deve rispettare il seguente vincolo temporale: dal momento in cui la PalmaViaggi inizia la procedura di prenotazione del pacchetto viaggio (ricezione delle date del viaggio), il processo di prenotazione deve concludersi entro 15 minuti. In caso contrario, la PalmaViaggi cancella il processo e notifica la cancellazione della procedura di prenotazione (Figura 7.2).

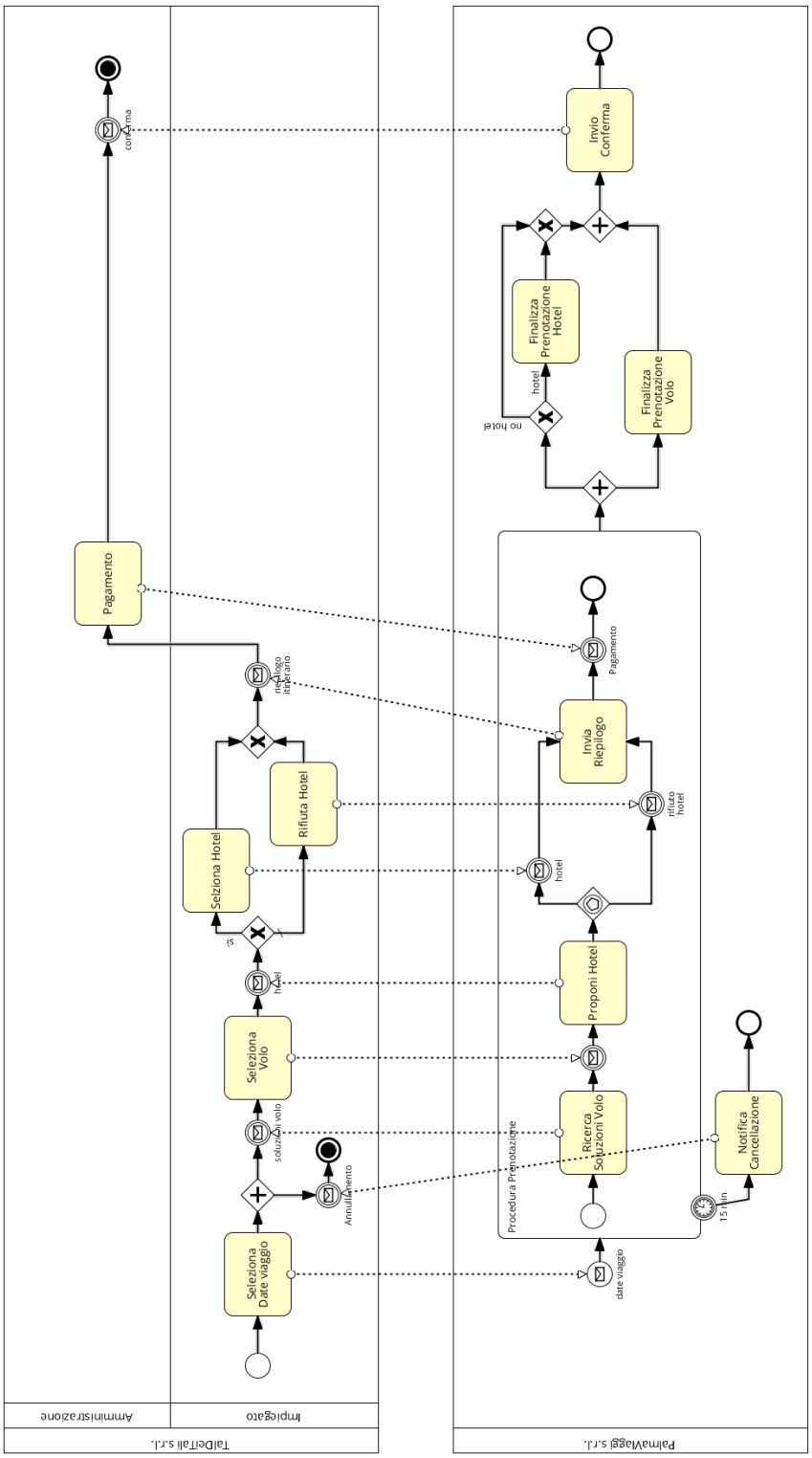


Figura 7.2: *PalmaViaggi s.r.l. - timer prenotazione*

7.3 Request for Quotation

Un sistema RFQ (request for quotation) è un tipico processo il cui scopo è quello di invitare i fornitori a fare offerte relative a prodotti. Il processo coinvolge quindi un cliente, interessato al prodotto, e un venditore che lo fornisce.

A iniziare il processo è il cliente, che contatta il venditore per ottenere un'offerta relativa a un prodotto. Nel momento in cui riceve la richiesta, il venditore verifica come prima cosa l'affidabilità del cliente. Se esso non è affidabile, la richiesta viene rifiutata e il processo termina. Se il cliente è affidabile, il venditore stima la data in cui il prodotto sarà disponibile.. Contemporaneamente, il venditore determina anche i tempi di consegna. Quest'ultima operazione deve essere effettuata solo nel caso di spedizione in uno Stato straniero. Nel caso di spedizioni all'interno dello stesso Stato, l'operazione di stima della data di consegna non è necessaria. I dati elaborati vengono quindi inviati al cliente sotto forma di proposta, e il venditore rimane in attesa dell'ordine definitivo del cliente. Nel momento in cui riceve l'ordine, il venditore invia un messaggio di conferma e il processo termina. Il processo termina anche se il cliente rifiuta l'offerta, o se non arriva nessuna comunicazione entro 48 ore.

Modellare sia il processo del venditore che quello del cliente (Figura 7.3).

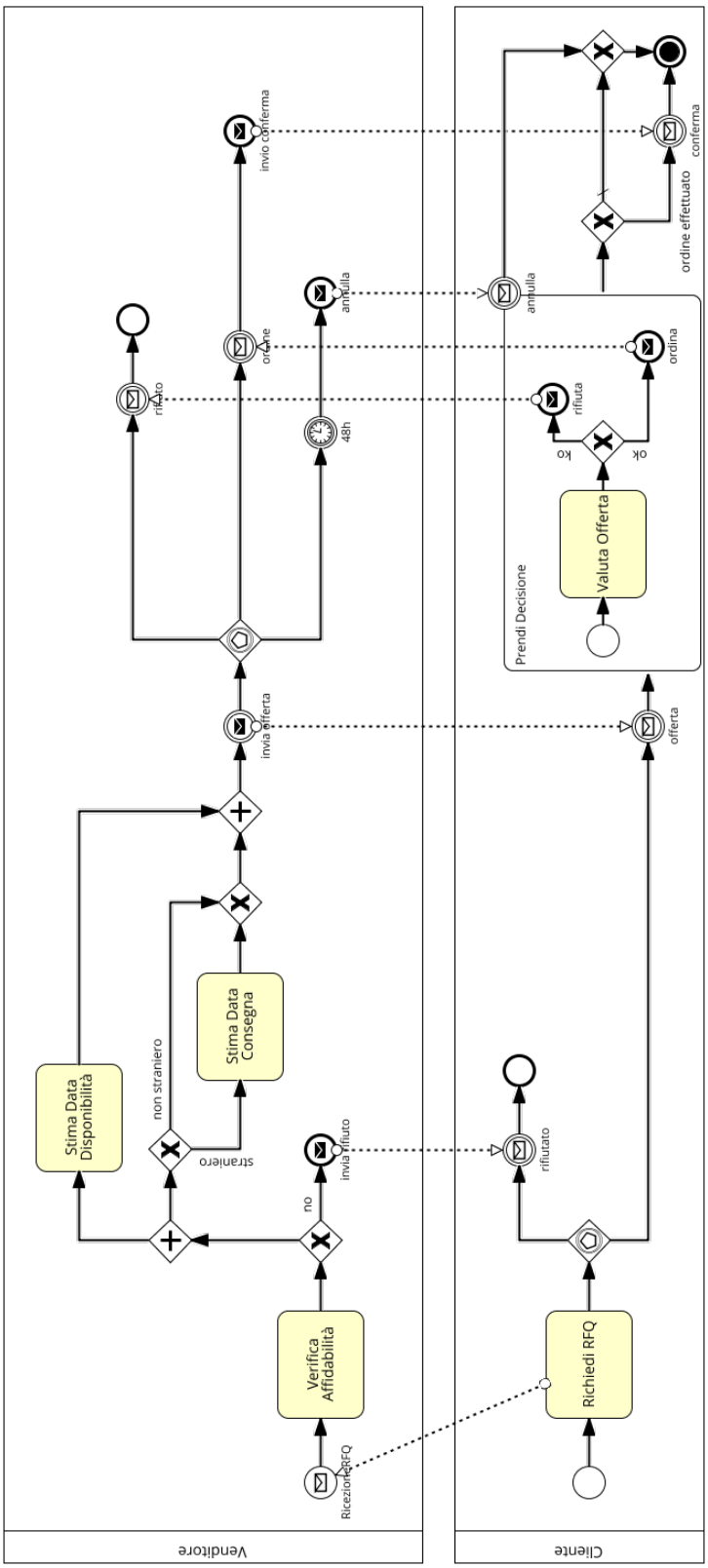


Figura 7.3: *Request for Quotation*

7.4 Software ABC

Si descriva, utilizzando il formalismo BPMN, il processo di seguito riportato.

Il contesto operativo del processo è quello della gestione dei problemi riscontrati da un cliente nell'utilizzo del prodotto software ABC fornito dalla società XYZ. Il cliente è in contatto diretto con una persona dello staff della società XYZ che costituisce l'unica interfaccia tra cliente e fornitore. Questa scelta scaturisce dalla strategia aziendale di gestione del cliente e non viene mai disattesa. Il cliente segnala alla persona di riferimento l'insorgenza del problema e, se richiesto, fornisce ulteriori spiegazioni relativamente allo stesso. La persona dello staff verifica se possiede le competenze necessarie per rispondere direttamente al cliente ed eventualmente vi provvede immediatamente. Nel caso in cui il problema non sia direttamente risolvibile viene chiamato in causa l'help desk di primo livello, mediante una dettagliata descrizione del problema. Se l'help desk di primo livello è in grado di risolvere la questione comunica direttamente la soluzione alla persona dello staff responsabile dell'interazione con il cliente, altrimenti il problema viene inviato al team di sviluppo. Il team di sviluppo esamina il problema e se la soluzione non prevede integrazioni al software esistente comunica la risposta all'help desk e la soluzione viene fatta pervenire al cliente mediante la stessa trafila operativa (help desk, persona dello staff). Nel caso in cui invece la soluzione preveda lo sviluppo software il team procede all'integrazione. Una volta terminata l'integrazione la soluzione viene fatta pervenire al cliente mediante la stessa trafila operativa (help desk, persona dello staff). Nel caso in cui, a partire dalla presa in carico del problema da parte dell'help desk non si arrivi alla comunicazione di una soluzione entro il termine di 10 giorni, l'help desk comunicherà comunque al personale dello staff un messaggio da far pervenire al cliente, redatto a cura dell'help desk stesso in cui si accusa il ritardo relativamente all'invio della soluzione. Una volta trovata la soluzione questa verrà comunque comunicata al cliente usando le modalità previste. Al termine del processo la persona dello staff comunica comunque una soluzione al cliente (anche se non pienamente risolutiva). Soluzione in Figura 7.4.

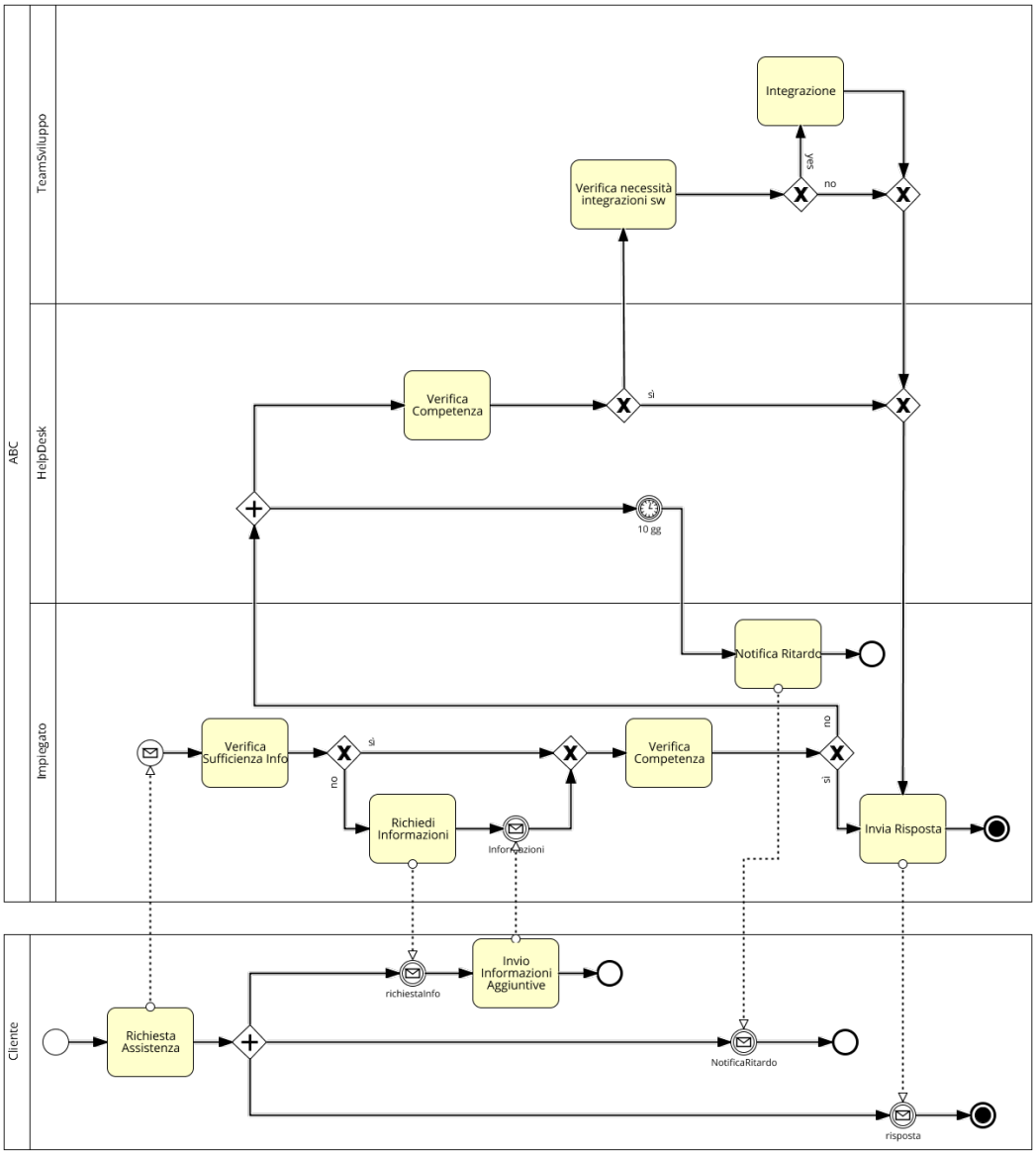


Figura 7.4: *Software ABC*

7.5 Comitato di redazione

Si rappresenti il processo di seguito descritto (Figura 7.5). La rivista LMN utilizza un metodo per la selezione e la pubblicazione degli articoli fortemente collaborativo. I partecipanti a questo processo sono costituiti dai seguenti soggetti:

- Il Direttore del Comitato di Redazione
- Il Comitato di Redazione: composto da 10 membri
- I Collaboratori: sono le persone (circa 15) che materialmente scrivono gli articoli per la rivista.

L'intero processo inizia quando il Direttore decide che è tempo di raccogliere i contributi per la prossima pubblicazione. Tale comunicazione viene inviata ai membri del Comitato di Redazione che possono compilare una Lista di Argomenti da inviare al direttore. L'invio della lista non è obbligatorio, ma deve avvenire entro 5 giorni dalla richiesta. Le liste inviate in ritardo verranno ignorate dal Direttore. Trascorsi 5 giorni, il Direttore conta le Liste degli argomenti che gli sono pervenute e verifica se è possibile predisporre una lista unica degli argomenti utilizzando i seguenti criteri:

- ci sono almeno 7 Liste degli Argomenti: il Direttore compone una lista di 20 argomenti, ordinati per importanza, nel caso in cui non abbia a disposizione i 20 argomenti, aggiunge personalmente gli argomenti mancanti nell'ordine da lui ritenuto più opportuno
- ci sono da 2 a 6 Liste degli Argomenti: il Direttore compone una lista di 15 argomenti, ordinati per importanza, eventualmente integrando come al punto precedente
- non c'è nessuna Lista degli argomenti oppure al più ce n'è una: il Direttore comunica che non è possibile creare una Lista degli Argomenti al Comitato di Redazione.

Se le liste sono sufficienti, il Direttore comunica la lista definitiva di argomenti al Comitato di Redazione e assegna a suo arbitrio uno o due argomenti a ciascun Collaboratore che scriverà il suo contributo e lo invierà al Direttore. Il Direttore attende i contributi per un massimo di 5 giorni. Eventuali contributi arrivati in ritardo saranno ignorati.

Nella soluzione indicare anche la gestione dei dati.

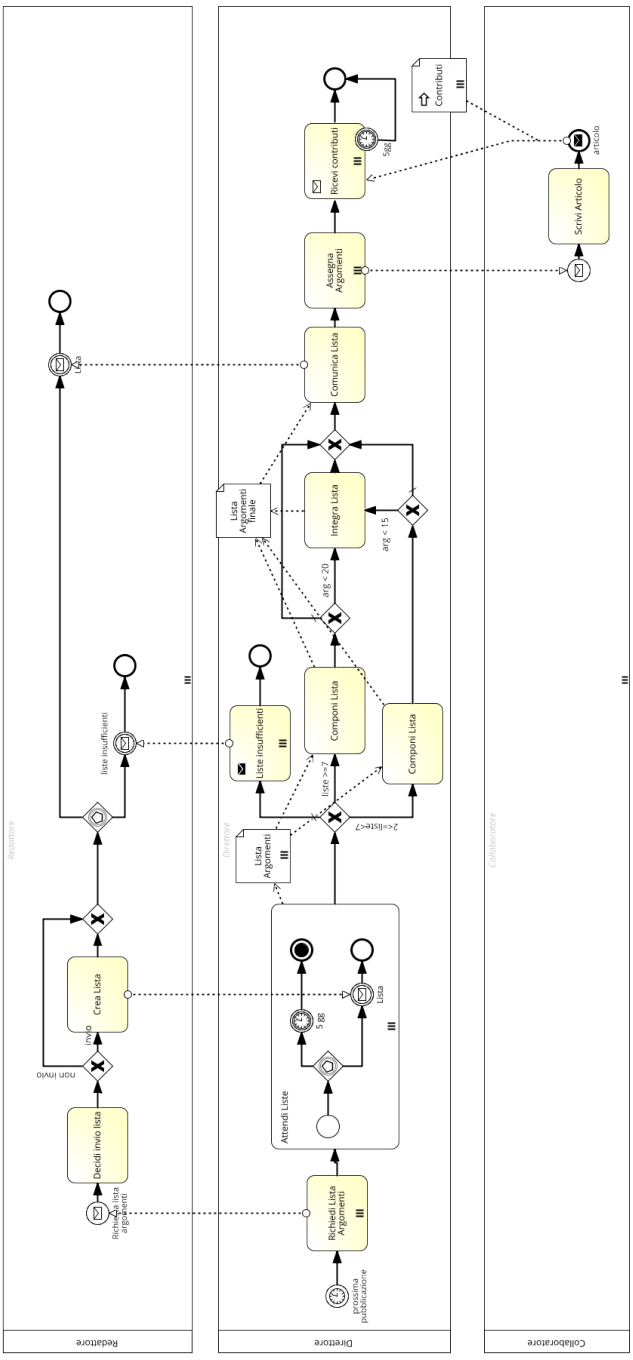


Figura 7.5: *Comitato di redazione - lista argomenti*

7.6 Comitato di redazione - variazione

Si rappresenti il processo di seguito descritto (Figura 7.6). La rivista LMN utilizza un metodo per la selezione e la pubblicazione degli articoli fortemente collaborativo. I partecipanti a questo processo sono costituiti dai seguenti soggetti:

- Il Direttore del Comitato di Redazione
- Il Comitato di Redazione: composto da 10 membri
- I Collaboratori: sono le persone (circa 15) che materialmente scrivono gli articoli per la rivista.

L'intero processo inizia quando il Direttore decide che è tempo di raccogliere i contributi per la prossima pubblicazione. A tale scopo notifica una lista di argomenti al comitato di redazione e assegna a suo arbitrio uno o due argomenti a ciascun Collaboratore che scriverà il suo contributo e lo invierà al Direttore. Il Direttore attende i contributi per un massimo di 5 giorni. Eventuali contributi arrivati in ritardo saranno ignorati. Trascorsi i cinque giorni il Direttore procede nel seguente modo:

- se gli sono pervenuti almeno 10 articoli la rivista verrà pubblicata, altrimenti comunica al Comitato di Redazione che gli articoli non sono abbastanza;
- se ci sono più di 10 articoli il Direttore compone una lista ordinata e la sottopone al voto del Comitato di Redazione.

Ogni membro del Comitato di Redazione entro due giorni deve indicare quali Articoli a suo giudizio vanno pubblicati. Il Direttore raccoglie i voti del comitato di Redazione e identifica i 10 articoli che andranno pubblicati: quelli cioè che hanno raccolto più voti. Comunica infine alla commissione gli articoli selezionati.

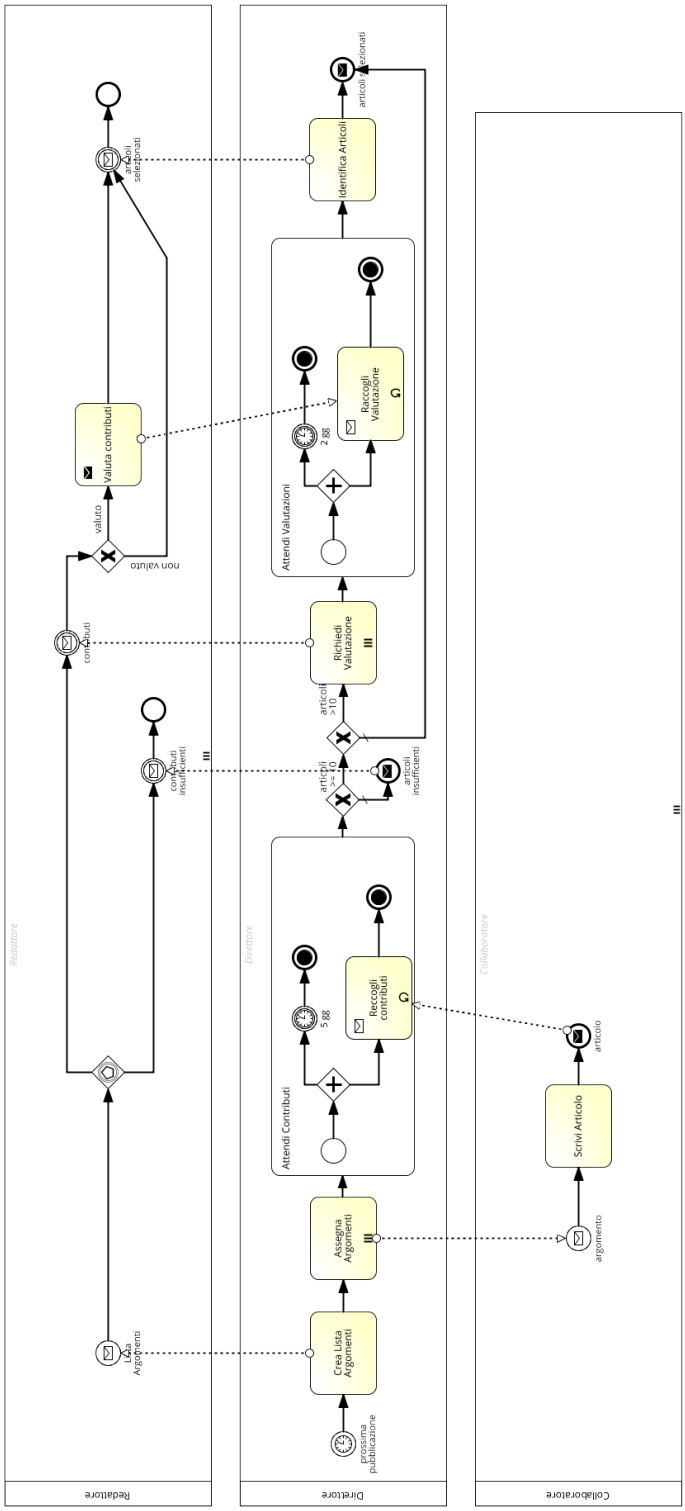


Figura 7.6: *Comitato di redazione - contributi*

7.7 ArteInStrada

Il comune di Milano ha recentemente inaugurato il progetto "ArteInStrada" che permette agli artisti di strada di prenotare location sparse nel centro storico per esibirsi in modo gratuito.

L'artista che vuole esibirsi accede al sistema e inserisce le informazioni relative alla sua esibizione (titolo, categoria, livello di rumorosità), seleziona poi una data e infine le dimensioni dello spazio di cui ha bisogno. Terminate queste operazioni, invia la richiesta al settore prenotazione di ArteInStrada. Il sistema, ricevuta la richiesta, verifica tra le location disponibili per la data segnalata, quelle che rispondono ai requisiti di spazio indicati dall'utente e invia l'elenco all'artista. Se nessuna location corrisponde ai requisiti indicati, il sistema invia la segnalazione all'artista e il processo termina.

Nel momento in cui riceve la lista delle location, l'artista ne seleziona una e poi attende l'approvazione finale della richiesta da parte di ArteInStrada. La richiesta completa viene presa in esame dalla commissione artistica di ArteInStrada che verifica che la proposta sia in linea con il regolamento del progetto. In questo caso viene comunicata l'approvazione all'artista, altrimenti la richiesta viene rifiutata e il processo termina. Per evitare di ricevere solleciti da parte degli artisti in attesa di conferma, se il processo di approvazione non è completato entro un giorno il sistema invia una notifica all'utente in cui viene segnalato che la richiesta è sotto esame.

Quando l'artista riceve la notifica di approvazione, prepara una versione definitiva della proposta seguendo eventuali indicazioni presenti nella notifica e invia la conferma a ArteInStrada. Il settore prenotazione, ricevuta la conferma termina il processo. Il settore prenotazione attende la conferma da parte dell'artista per 2 giorni, scaduti i quali annulla la prenotazione e comunica l'annullamento all'artista, terminando il processo (Figura 7.7).

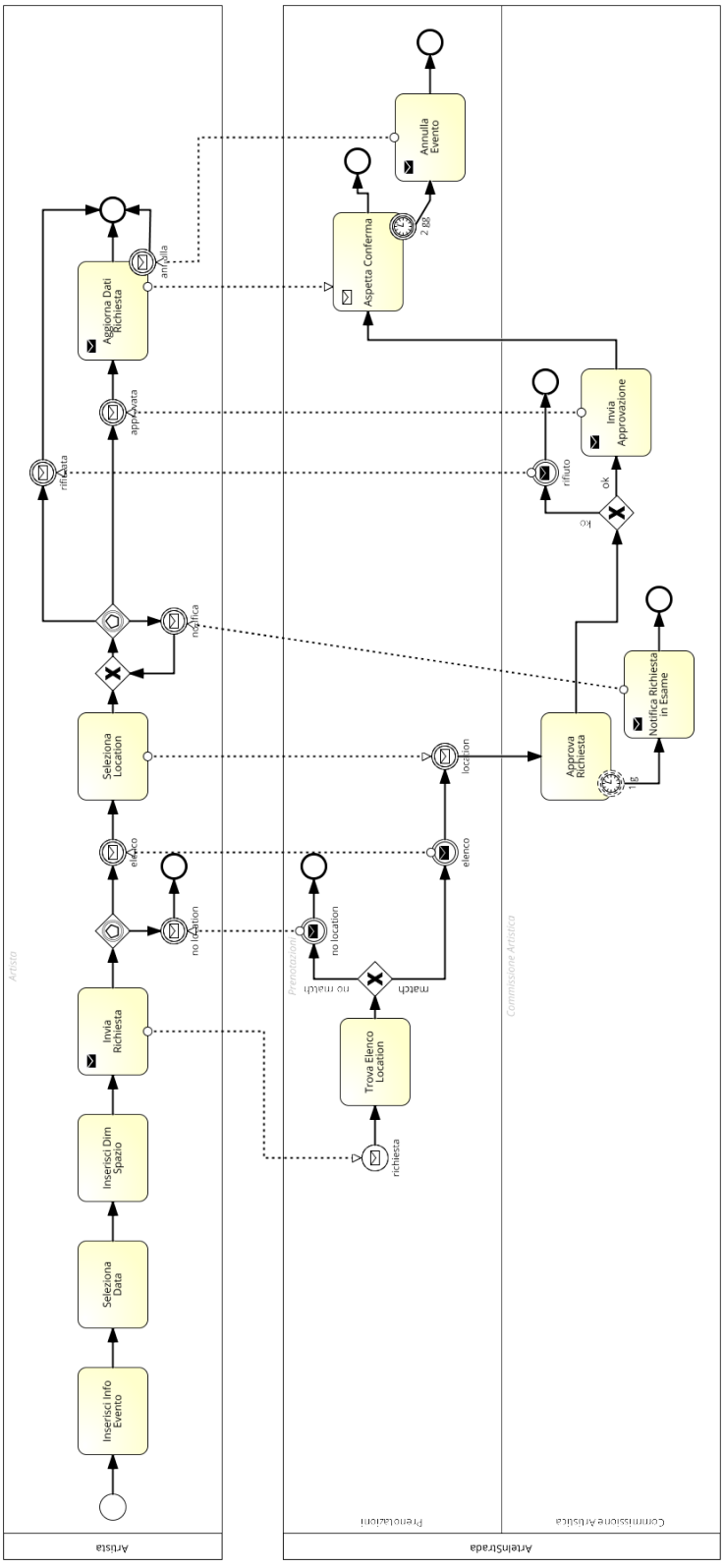


Figura 7.7: *ArteInStrada*

7.8 Amministrazione associazione XP

Utilizzando la notazione BPMN si rappresenti il processo descritto di seguito.

L'associazione XP prende decisioni relative all'amministrazione del patrimonio societario secondo la seguente logica. Le decisioni vengono prese da:

- un responsabile;
- un consiglio di amministrazione, detto CDA;
- un supervisore.

Le comunicazioni tra i soggetti avvengono tutte tramite posta elettronica (considerare i soggetti come entità separate). Il responsabile è il soggetto incaricato di proporre investimenti riguardo al patrimonio. In base al tipo di investimento, la decisione può avvenire con tre modalità:

- Decisione Spontanea (DS): il responsabile comunica la sua decisione al CDA a al supervisore. Il CDA prende atto della decisione e ne attende conferma. Il responsabile attende per 3 giorni eventuali interventi del supervisore, trascorsi i quali la proposta viene approvata. Se il supervisore interviene entro gli 8 giorni, la proposta viene annullata e il processo termina. Altrimenti l'approvazione viene confermata al CDA.
- Decisione con Consultazione del CDA (DC): il responsabile comunica una proposta al CDA che indice una riunione interna e raccoglie i pareri. Al termina della riunione invia un report al responsabile. Il responsabile valuta il report e decide se annullare la proposta o approvarla. In caso di approvazione, questa viene comunicata al supervisore che ha 3 giorni per esprimere parere contrario. Al termine dei 3 giorni il responsabile approvazione definitivamente la proposta e comunica l'approvazione al CDA. In caso di intervento del supervisore la proposta viene annullata.
- Decisione a Voto di Maggioranza del CDA (DV): in questo caso le opinioni del CDA sono vincolanti, e il responsabile richiede una votazione della proposta. Al termine della procedura di votazione, il conteggio dei voti viene inviato al responsabile. Il responsabile valuta l'esito e se i voti favorevoli sono sufficienti approva in modo definitivo la proposta e invia comunicazione al CDA. E' necessaria la maggioranza assoluta dei membri del CDA. Il responsabile attende il risultato della votazione per un massimo di 8 giorni, allo scadere dei quali la proposta viene annullata. In questo caso il supervisore non viene coinvolto nella decisione.

La soluzione è mostrata in Figura 7.8.

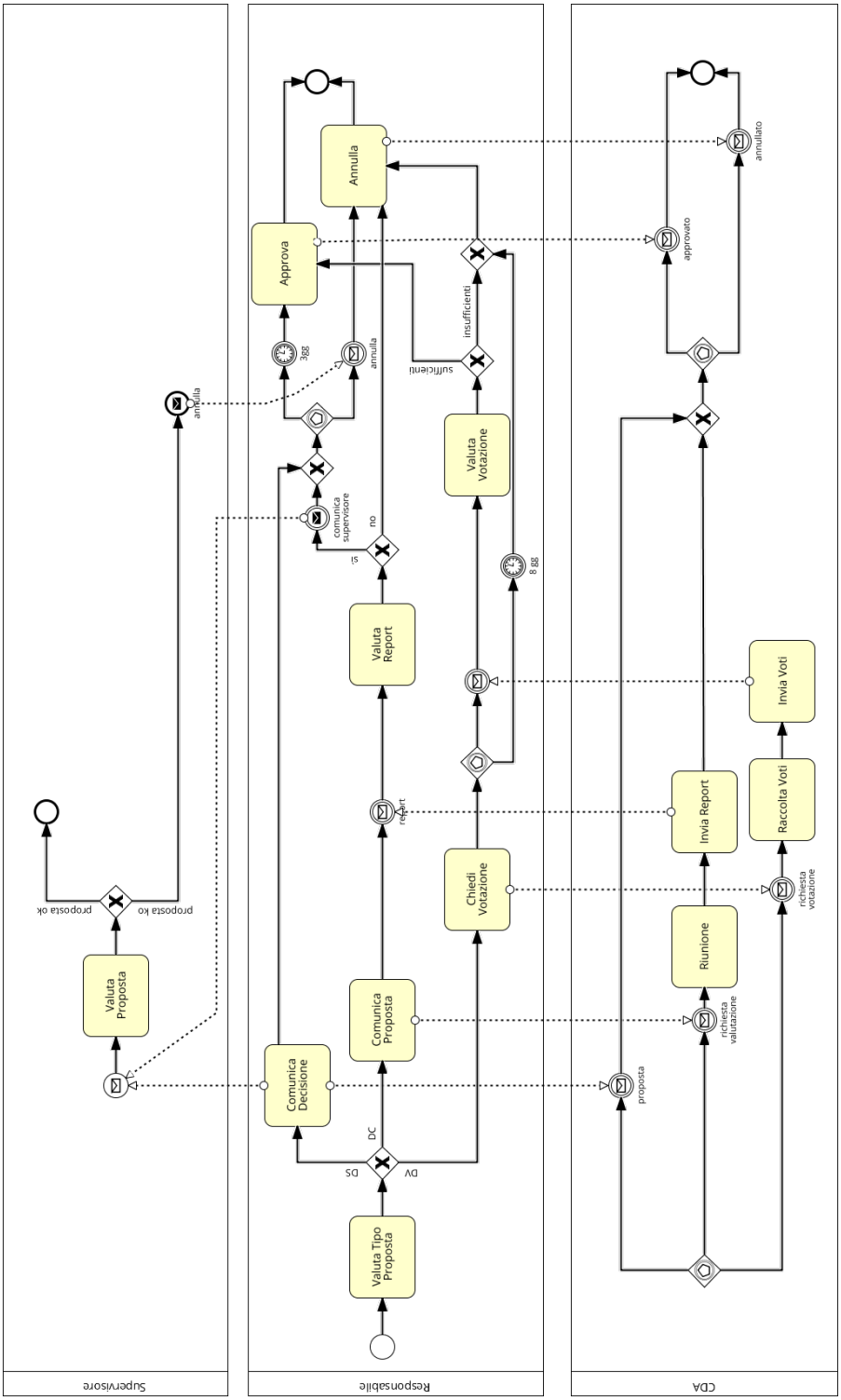


Figura 7.8: *Amministrazione associazione XP*

7.9 Virgo

Si modelli il processo di selezione di nuovo personale della società Virgo. Nel momento in cui si richiede l'inserimento di una nuova persona nell'organico della società, l'ufficio tecnico provvede a preparare tutta la documentazione necessaria comprensiva dei requisiti che il candidato deve possedere. Completata la documentazione, questa viene pubblicata online per dar modo agli utenti interessati di sottomettere la propria candidatura. Le candidature sono raccolte in maniera automatica dal sistema (senza cioè interventi manuali), il quale provvede a registrare i dati ricevuti all'interno della base di dati. Il processo di ricezione di eventuali candidature dura esattamente 15 giorni, al termine dei quali non deve più essere possibile ricevere candidature. Al termine del periodo di raccolta delle candidature si avvia la fase di valutazione delle stesse. Il personale dell'ufficio tecnico provvede a vagliare le candidature una alla volta in ordine di arrivo. Per ognuna di esse, provvede ad inviare i dati del candidato a tre membri di una commissione esterna all'azienda e attende che questi inviino il proprio giudizio sul candidato (il giudizio è basato su una scala a valori discreti). Una volta ricevute tutte le valutazioni provvede a controllare se il candidato ha ottenuto almeno due voti sufficienti su tre. In caso negativo, si procede direttamente con la valutazione del candidato successivo (se ve ne sono ancora), altrimenti si provvede a registrare i dati del candidato nel sistema. Una volta esauriti i candidati, il dipartimento tecnico sceglie a propria discrezione uno dei candidati che ha superato la fase di votazione (se ve ne sono). Si noti come non sussistano vincoli temporali legati alle valutazioni della commissione. D'altro canto però l'intera fase di valutazione di tutti i candidati deve, per politiche aziendali, concludersi entro 5 giorni dal suo avvio. Se così non fosse, la fase di valutazione deve terminare e si procede di nuovo all'intero iter di processo come già descritto ripartendo dalla preparazione di nuova documentazione. Qualora il vincolo dei 5 giorni venga soddisfatto senza trovare però un candidato adatto, si procede pubblicando nuovamente l'annuncio online senza produrre nuova documentazione in attesa di nuovi candidati da valutare.

Si richiede di modellare tutti gli attori coinvolti nel processo ad eccezione del candidato (Figura 7.9).

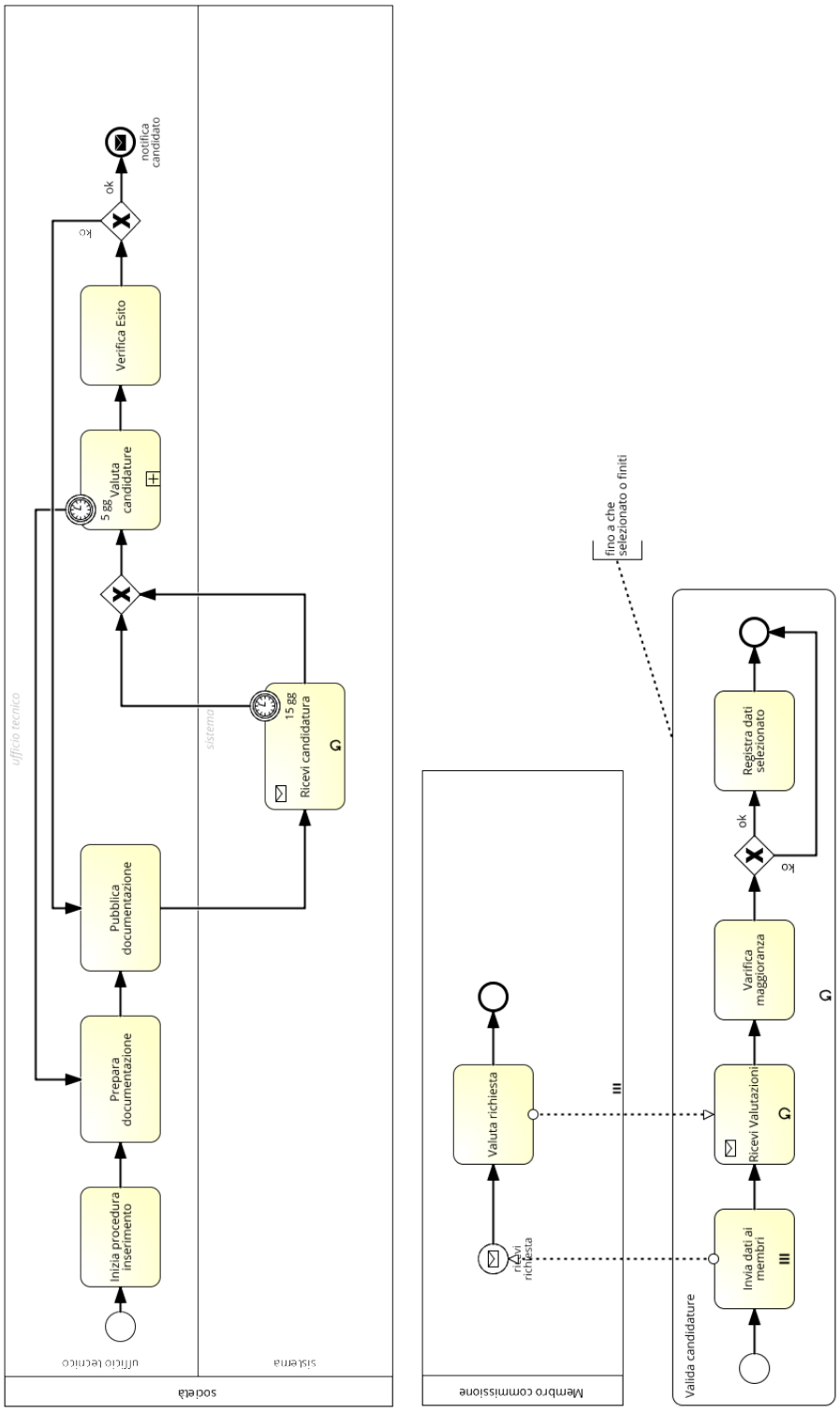


Figura 7.9: *Virgo - selezione personale*

7.10 Slow food

Una catena di slow food specializzata in hamburger vuole usufruire di un sistema informativo per la gestione del suo locale.

Modellare utilizzando la notazione BPMN il processo di inoltro di un ordine in remoto qui descritto. Il cliente, usando il sito web della catena, seleziona in un primo momento la categoria di prodotto che vuole ordinare (hamburger, contorni, dolci), selezionato poi uno specifico prodotto può decidere se personalizzarlo o meno con ingredienti opzionali. Selezionato il prodotto invia l'informazione al sistema che lo aggiunge al carrello. Il cliente può decidere se aggiungere nuovi prodotti (seguendo di nuovo lo stesso processo fino a qui descritto), oppure confermare l'ordine. Si noti come una volta aggiunto un primo prodotto, il processo deve necessariamente essere finalizzato entro 10 minuti, oltre i quali il sistema annulla l'ordine, notificandolo al cliente. Una volta selezionati tutti i prodotti desiderati e confermato l'ordine, il sistema elabora la fattura e la mostra al cliente, il quale dovrà inserire gli estremi della propria carta di credito per finalizzare il pagamento. Il sistema controlla quindi la correttezza dei dati e notifica al cliente l'esito. Se positivo il processo termina, altrimenti il cliente ha la possibilità di inserire nuovamente i dati o di annullare l'ordinazione (Figura 7.10).

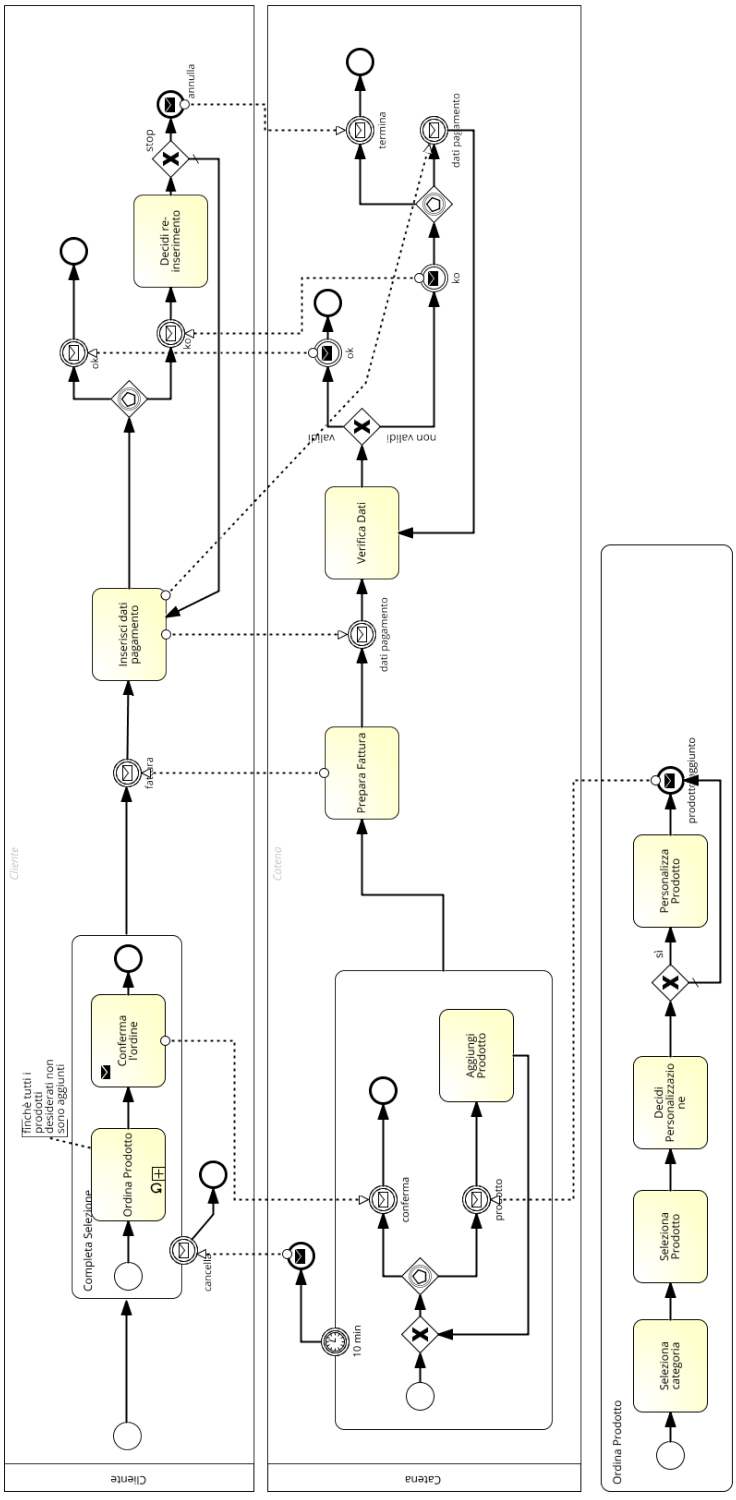


Figura 7.10: *Slow Food*

7.11 Offerta Prestito

L'ufficio Marketing di una società di mediazione creditizia decide di avviare una campagna di marketing mandando ad alcuni clienti un'offerta di prestito a tasso agevolato. L'Ufficio prepara una lista di clienti da contattare a cui poi manda la proposta di prestito. Se dopo due settimane dall'invio l'ufficio non riceve alcuna risposta o se il cliente risponde declinando l'offerta l'ufficio registra la risposta e cancella il nominativo dalla lista. Se invece il cliente invia alla società l'accettazione dell'offerta insieme ai documenti richiesti, la gestione della pratica passa all'Amministrazione.

L'Amministrazione controlla i documenti inviati e se sono completi, convoca il cliente in sede per la firma della pratica. Se i documenti inviati dal cliente non risultano essere completi, l'amministrazione richiede i documenti mancanti. Alla ricezione dei documenti da parte del cliente, l'Amministrazione ripete il controllo di completezza. Se passano 10 giorni senza ricevere risposta dal cliente, l'Amministrazione manda una comunicazione di sollecito al cliente. In ogni caso dopo un mese dalla richiesta l'ufficio registra il fallimento dell'operazione e cancella il nominativo dalla lista.

Modellare mediante la notazione BPMN il processo sopra descritto (Figura 7.11

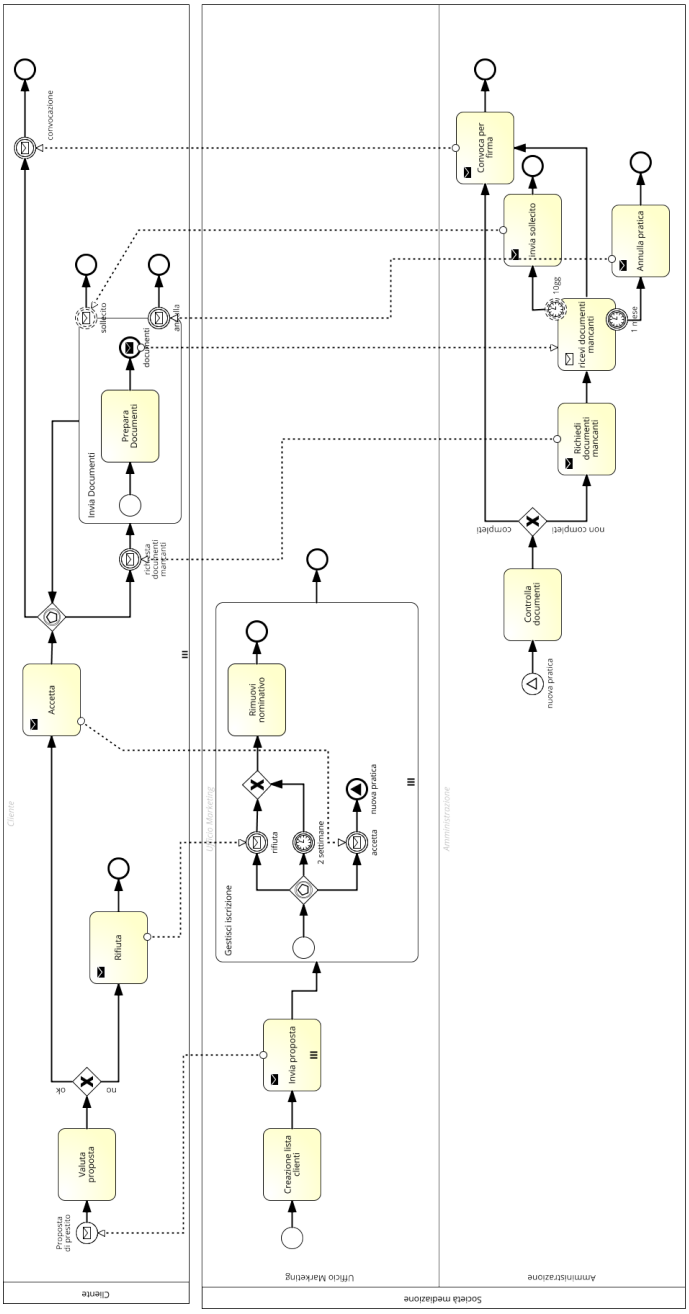


Figura 7.11: *Offerta prestito*

7.12 FabbricaLib

La compagnia FabbricaLib offre ai suoi clienti un sistema per la creazione di librerie personalizzate. Si modelli utilizzando BPMN il processo che descrive l'interazione tra il cliente e la società durante la configurazione e l'acquisto di una libreria personalizzata.

Il cliente che vuole acquistare una libreria deve innanzitutto sceglierne le dimensioni e scegliere il materiale di cui vuole che la libreria sia fatta. Queste due operazioni possono essere eseguite contemporaneamente. Una volta configurata la libreria, invia la richiesta di preventivo alla società FabbricaLib. La società, ricevuta la richiesta, calcola il preventivo sulla base della configurazione e lo invia al cliente. Il cliente può quindi scegliere se accettare il preventivo e procedere all'acquisto o se annullare la procedura. Nel caso di annullamento il processo termina, previo invio di una notifica alla società. Altrimenti, il cliente conferma il suo ordine e la società, ricevuta la conferma, invia le informazioni di pagamento al cliente. La società attende quindi l'invio del pagamento che deve avvenire entro 10 minuti, allo scadere dei quali l'ordine viene annullato. Il cliente, ricevuti i dettagli di pagamento, può inoltre decidere di non procedere con l'acquisto e di cancellare l'ordine. Nel caso in cui il pagamento venga effettuato, la società inoltra l'ordine e invia la fattura al cliente. L'eventuale cancellazione da parte del cliente deve essere gestita dalla società che deve terminare il suo processo.

Modellare sia la società che il cliente (Figura 7.12).

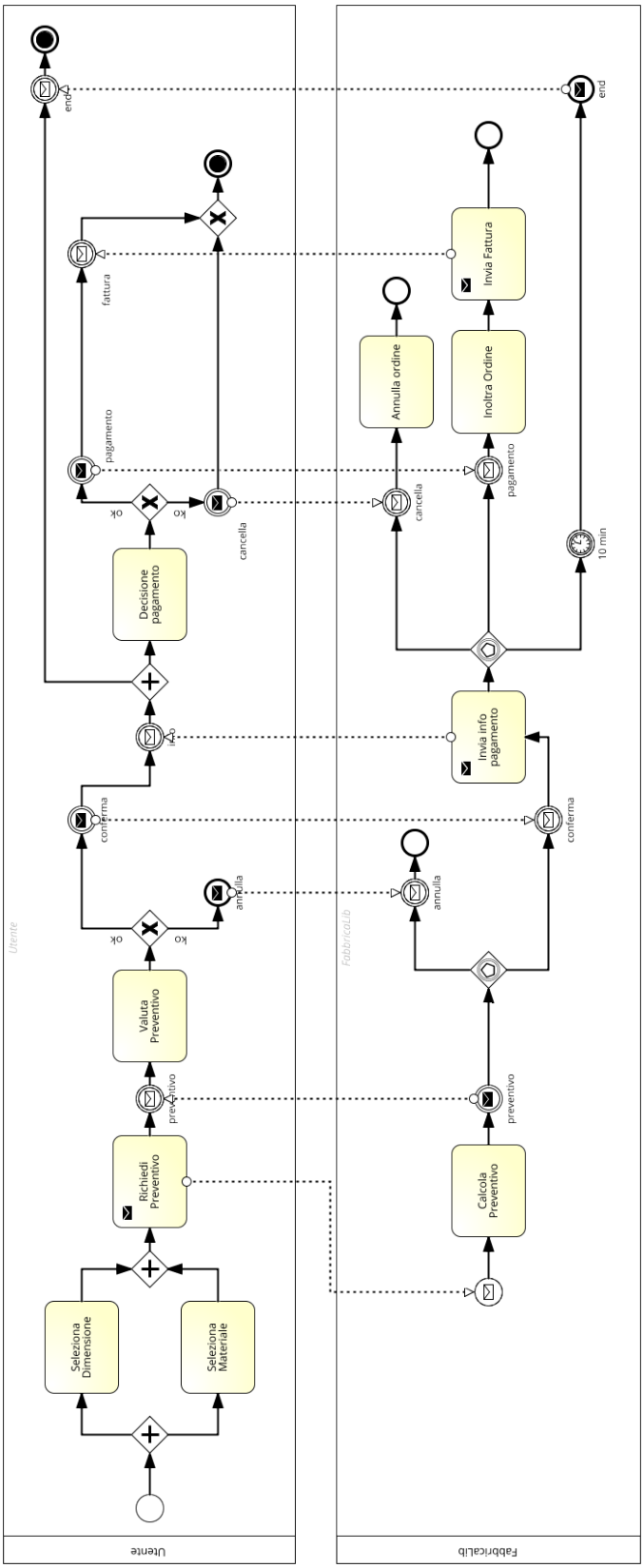


Figura 7.12: *FabbricaLib versione 1*

7.13 FabbricaLib - variazione

La compagnia FabbricaLib offre ai suoi clienti un sistema per la creazione di librerie personalizzate. Si modelli utilizzando BPMN il processo che descrive l'interazione tra il cliente e la società durante la configurazione e l'acquisto di una libreria personalizzata.

Il cliente che vuole acquistare una libreria deve innanzitutto sceglierne le dimensioni e scegliere il materiale di cui vuole che la libreria sia fatta. Queste due operazioni possono essere eseguite contemporaneamente. Una volta configurata la libreria, invia la richiesta di preventivo alla società FabbricaLib. La società, ricevuta la richiesta, calcola il preventivo sulla base della configurazione e lo invia al cliente. Questa operazione può durare al massimo 2 giorni, al termine dei quali un messaggio di errore viene inviato al cliente e il processo termina. Se il preventivo viene inviato correttamente, il cliente può quindi scegliere se accettare il preventivo e procedere all'acquisto o se annullare la procedura. Nel caso di annullamento il processo termina, previo invio di una notifica alla società. Altrimenti, il cliente conferma il suo ordine e la società, ricevuta la conferma, invia le informazioni di pagamento al cliente e attende di ricevere i dati della carta di credito. Il cliente, ricevuti i dettagli di pagamento, può inoltre decidere di non procedere con l'acquisto e di cancellare l'ordine. Se i dati vengono inviati, la società esegue una procedura di verifica. Se i dati sono corretti, la società inoltra l'ordine e invia la fattura al cliente. Se i dati non sono corretti, un messaggio di errore viene inviato al cliente che può inviare di nuovo i dati di pagamento o cancellare l'operazione. L'eventuale cancellazione da parte del cliente deve essere gestita dalla società che deve terminare il suo processo.

Modellare sia la società che il cliente (Figura 7.13).

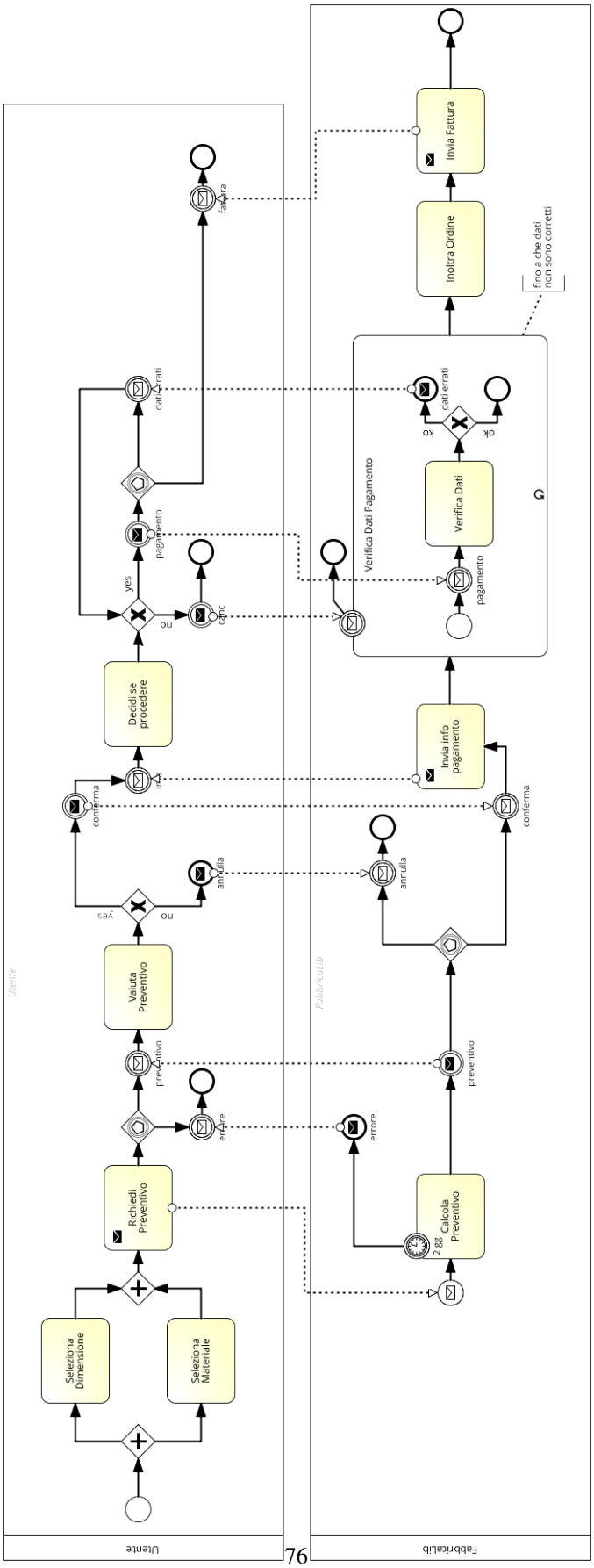


Figura 7.13: *FabbricaLib versione 2*

7.14 ACME

L'agenzia ACME organizza viaggi turistici, offerto ai clienti su un catalogo. I clienti interessati inviano all'agenzia una richiesta di informazioni per partecipare a un viaggio, indicando la destinazione, il periodo e il numero di partecipanti. L'agenzia verifica che il viaggio sia disponibile. Se il viaggio non è disponibile, viene inviata una notifica al cliente e il processo termina. Se c'è disponibilità, viene inviato un preventivo di offerta. Il cliente può accettare l'offerta. Se passano 7 giorni dall'invio del preventivo, l'agenzia fa scadere l'offerta e sia il processo dell'agenzia che quello del cliente terminano. Nel caso in cui il cliente accetti l'offerta, l'agenzia, in parallelo, prenota il viaggio e l'albergo; quando le prenotazioni sono entrambe completate, viene inviata una conferma al cliente e il processo di prenotazione termina (Figura 7.14).

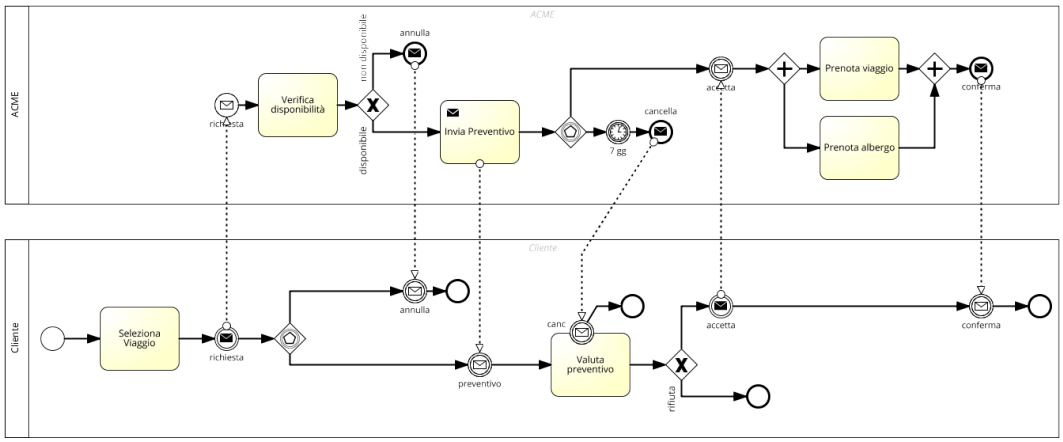


Figura 7.14: *ACME viaggi*

7.15 Gestione Rimborsi

Si chiede di rappresentare in BPMN il seguente processo, che coinvolge due enti, che saranno denominati Ufficio Ricezione e Ente di approvazione. I due enti sono distinti e seguono processi separati che interagiscono. Il primo ente raccoglie delle domande di rimborso da parte di associati tramite uno sportello. E' l'unico ad interagire direttamente con gli associati. L'Ufficio Ricezione verifica le domande presentate, se mancano informazioni invia una richiesta di chiarimento all'associato che ha presentato la domanda. In caso di mancata risposta dopo una richiesta di chiarimento, dopo 7 giorni la pratica viene archiviata e il processo termina. Quando la domanda è completa l'Ufficio Ricezione inserisce le informazioni relative all'abbonato in suo possesso e quindi invia la pratica all'ente, rimanendo in attesa di risposta.

L'Ente di approvazione riceve la pratica nell'ufficio Contabilità e, in parallelo, la Contabilità calcola l'importo dovuto mentre il settore Verifiche, ne valuta l'ammissibilità. Se ammissibile, vengono verificate le singole spese e viene calcolato il rimborso ammissibile. Completata questa attività, il settore verifiche invia la decisione alla Contabilità che invia quindi risposta all'Ufficio Ricezione (di rifiuto se non ammissibile o con la somma approvata se ammissibile). L'Ufficio ricezione risponde quindi all'associato.

Modellare i processi dell'Ufficio Ricezione e dell'Ente di approvazione (Figura 7.15).

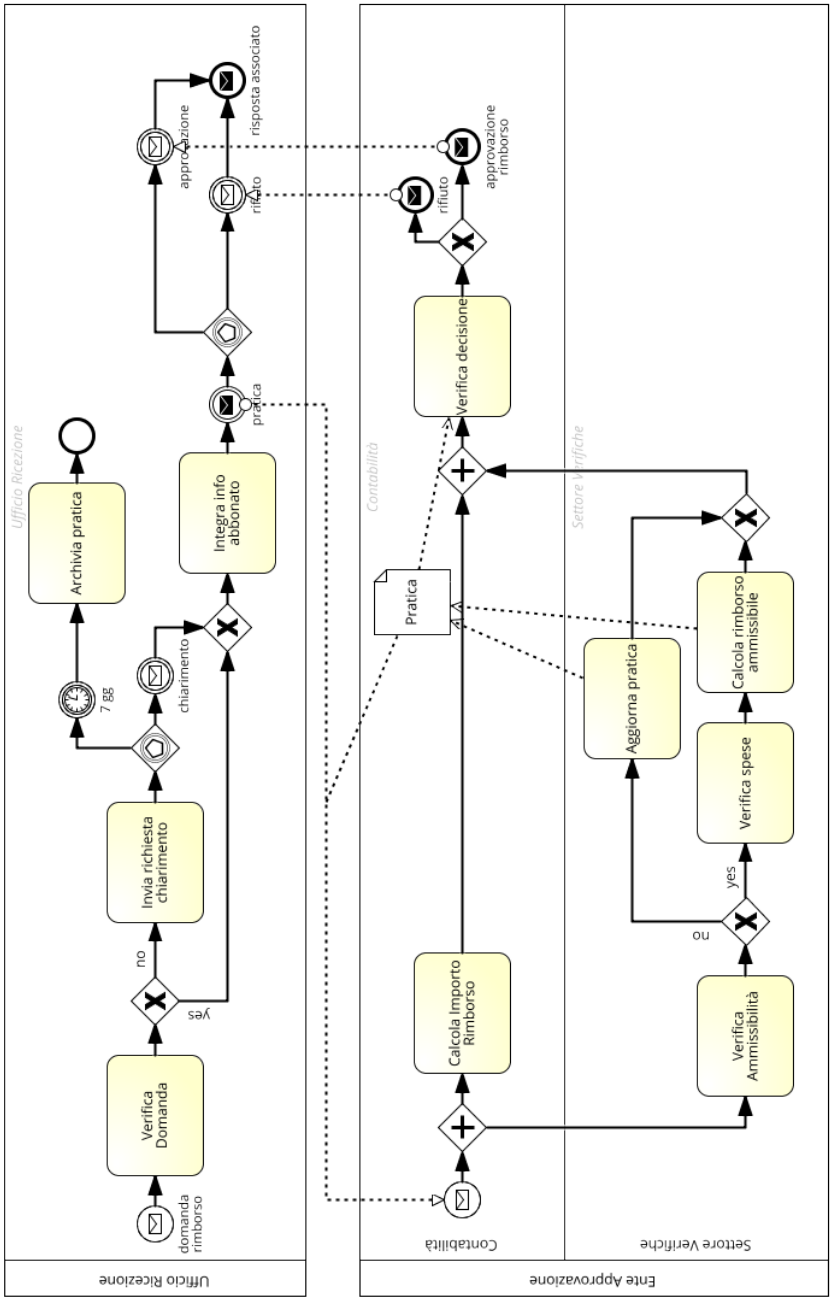


Figura 7.15: *Gestione Rimborsi*

7.16 Autoscuola

Si modelli utilizzando la notazione BPMN il seguente processo che descrive l'interazione di un cliente con il portale di un'autoscuola per la simulazione di un test teorico.

Il cliente invia una richiesta alla autoscuola specificando se vuole effettuare un test generico o un test su un argomento specifico. La autoscuola, ricevuta la richiesta, estrae in modo casuale 10 domande dal DB. Nel caso di test su un argomento, prima di eseguire questa attività è necessario specificare l'argomento di interesse del cliente. L'autoscuola invia quindi il set di domande al cliente. Ricevute le domande il cliente risponde e poi sottomette le risposte alla autoscuola che calcola il punteggio e invia al cliente l'esito del test. Il cliente quindi analizza i risultati e, nel caso in cui ritenesse una risposta scorretta, può inviare una segnalazione. Altrimenti conferma l'esito del test e il processo si conclude. La segnalazione di errore deve essere inviata entro 10 minuti dall'invio da parte dell'autoscuola del risultato. Altrimenti il test si ritiene approvato dal cliente e il processo dell'autoscuola e del cliente si conclude.

Nel caso di segnalazione, l'autoscuola analizza la richiesta pervenuta e invia un responso all'utente, terminando il processo (Figura 7.16).

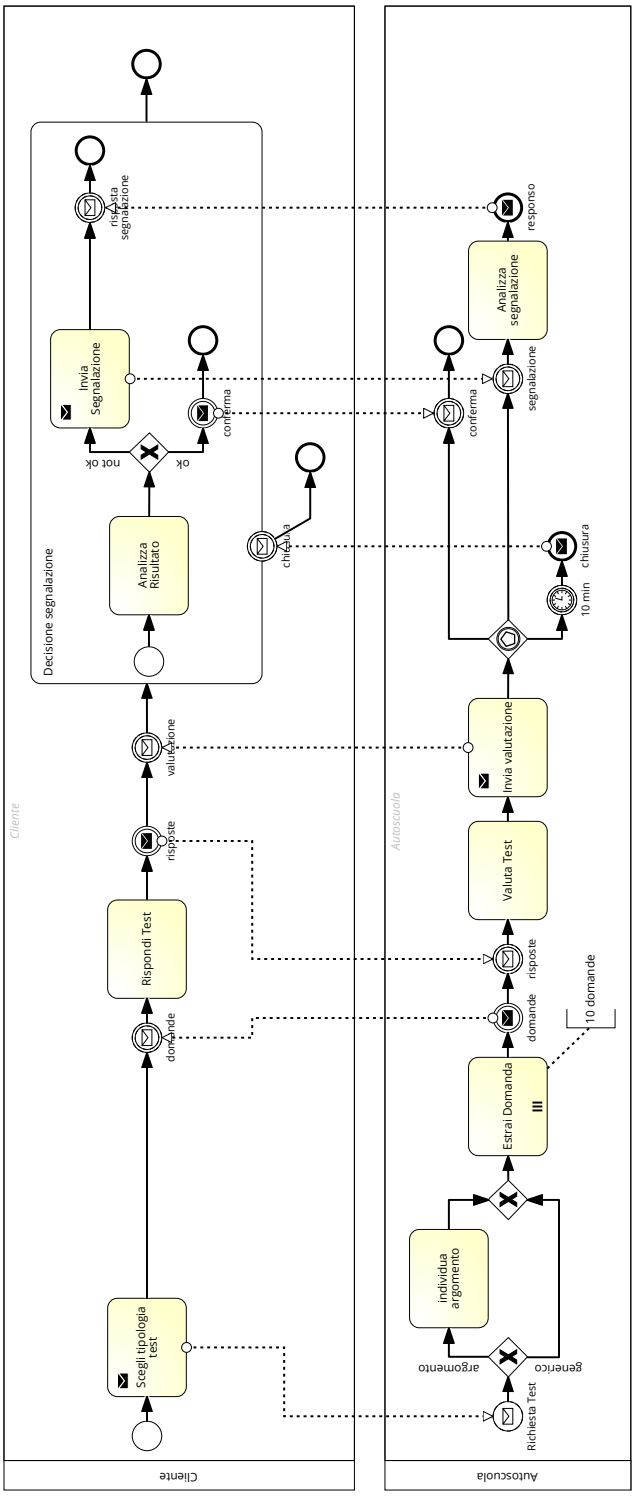


Figura 7.16: *Autoscuola*

7.17 Penna&Calamaio

Si modelli in BPMN il processo di vendita di Penna&Calamaio, una società di distribuzione all'ingrosso di articoli di cancelleria; il processo di vendita avviene tra Penna&Calamaio e i negozi che distribuiscono al dettaglio.

Il processo inizia con un ordine da parte di un negozio, in cui è indicata la merce desiderata. L'ufficio rivendite di Penna&Calamaio, dopo aver verificato la presenza dei beni a magazzino, inoltra l'ordine al magazzino che si occupa di impacchettarlo e inviarlo al corriere; il magazzino, inoltre, informa il negozio che la merce è in spedizione. Se la merce richiesta non è tutta a magazzino, l'ufficio rivendite stima il tempo di evasione. Se tale tempo è inferiore a 10 giorni allora l'ordine viene portato avanti, inoltrandolo al magazzino come sopra, altrimenti viene annullato completamente e il negozio viene informato.

Nel caso in cui il negozio non riceva l'ordine entro 5 giorni dalla notifica di spedizione, annulla l'ordine, informando l'ufficio rivendite. Una volta ricevuta la merce, il negozio ne controlla la correttezza e, nel caso tutto sia in ordine, effettua il pagamento altrimenti restituisce la merce e il processo si conclude. Quando Penna&Calamaio riceve conferma del pagamento, invia la fattura al negozio. La soluzione è illustrata in Figura 7.17.

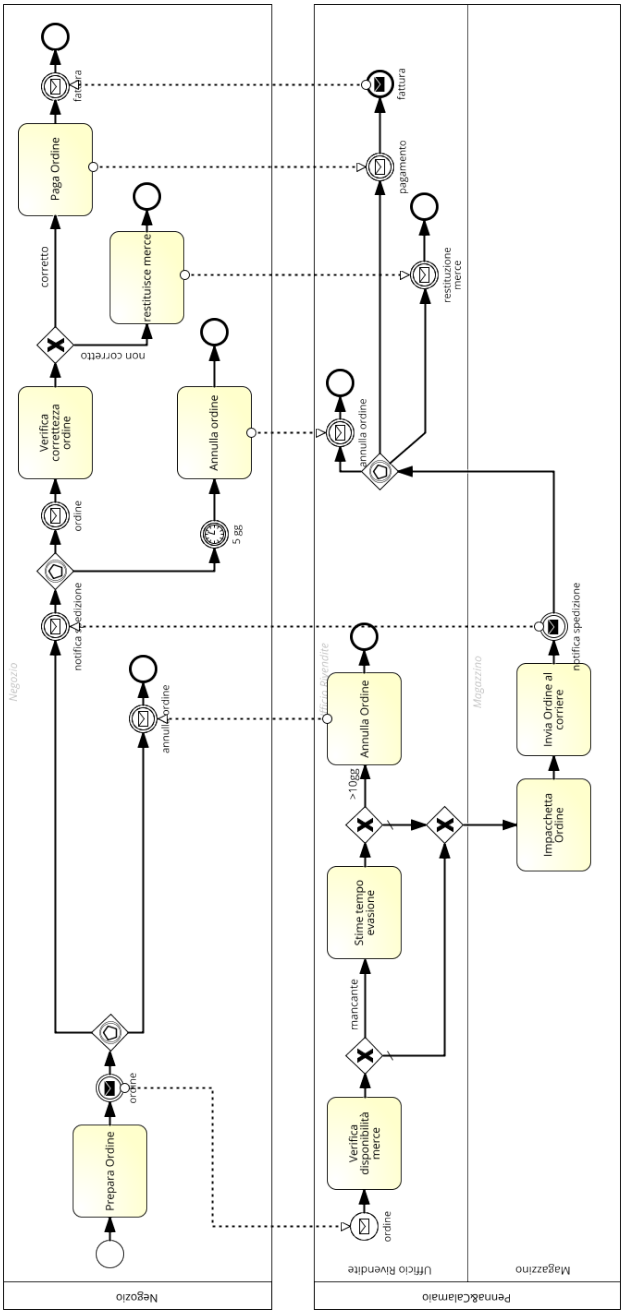


Figura 7.17: *Penna&Calamaio*

7.18 Ricorso ABF

Si modelli utilizzando BPMN il seguente processo per la gestione dei reclami dei clienti di una banca.

Al processo partecipano tre attori: il cliente, la banca, e l'ABF (un ente terzo che può intervenire nella risoluzione della controversia). Il processo inizia quando un cliente decide di effettuare un reclamo nei confronti di una banca. Alla ricezione del reclamo, la banca elabora la richiesta e, una volta presa una decisione, invia la risposta al cliente. Il cliente, trascorsi 30 giorni dall'invio del reclamo senza aver ricevuto risposta, può rivolgersi ad un ente esterno, l'ABF, per la risoluzione della controversia, inviando ad esso la documentazione relativa al reclamo. Ricevuta la documentazione, l'ABF verifica l'ammissibilità del ricorso. Nel caso in cui la richiesta non sia ammissibile, invia una notifica al cliente, a cui non rimane altro che aspettare la risposta della banca. Se invece il ricorso è ammesso, l'ABF invia una notifica alla banca.

Alla ricezione della notifica, la banca interrompe l'elaborazione della risposta al cliente e invia la documentazione relativa al ricorso all'ABF. Ricevuta la documentazione, l'ABF prepara l'istruttoria e notifica la sua decisione sia al cliente (che rimane in attesa della risposta della banca) che alla banca. La banca a questo punto deve adempiere alla decisione dell'ABF, e comunicare la risposta al cliente e all'ABF. Ricevuta la risposta, l'ABF termina il processo.

Se entro 30 giorni l'ABF non riceve notifica della risposta, pubblicizza l'inadempienza della banca (rimanendo comunque in attesa della risposta al reclamo). Soluzione in Figura 7.18.

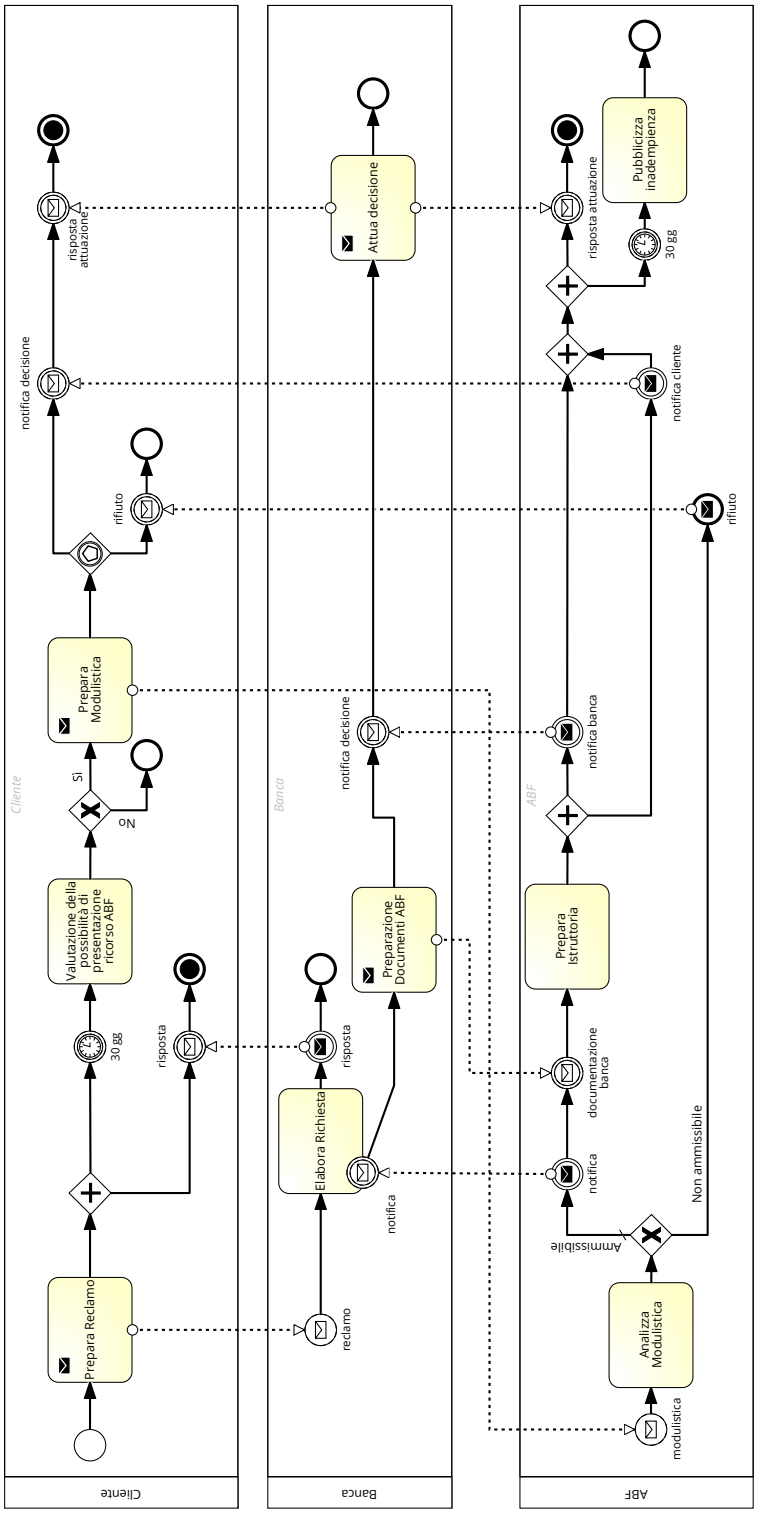


Figura 7.18: *Ricorso ABF*

7.19 U2Bike

Si vuole progettare il sistema informativo del produttore e rivenditore di biciclette U2Bike. Il sistema deve permettere agli utenti di ordinare una bicicletta.

Il processo che descrive la gestione degli ordini dell'azienda U2Bike è il seguente. Il processo inizia quando il cliente seleziona un modello dal catalogo e sceglie di acquistarlo. L'ordine viene inviato al reparto vendita di U2Bike che lo analizza, passa poi al reparto produzione che stima i tempi di produzione previsti. Il reparto vendita invia risposta al cliente con tale stima. Il cliente può quindi decidere se procedere con l'ordine o se annullare. Nel caso proceda, paga l'anticipo. Una volta che il reparto vendita ha ricevuto l'anticipo, il reparto produzione inizia il processo di produzione, terminato il quale il cliente riceve una notifica dal reparto vendita per ritirare la sua bicicletta ed entrambi i processi terminano. Qualora il processo di produzione sia in ritardo rispetto ai tempi previsti, la produzione stima i nuovi tempi di consegna e il reparto vendita notifica il ritardo al cliente. In tal caso, il cliente può decidere di rinunciare all'ordine, inviando una notifica di cancellazione all'azienda. Tale notifica interrompe la produzione e fa terminare il processo.

Realizzare il diagramma BPMN del processo di gestione di un nuovo ordine, modellando sia il cliente che l'azienda (Figura 7.19).

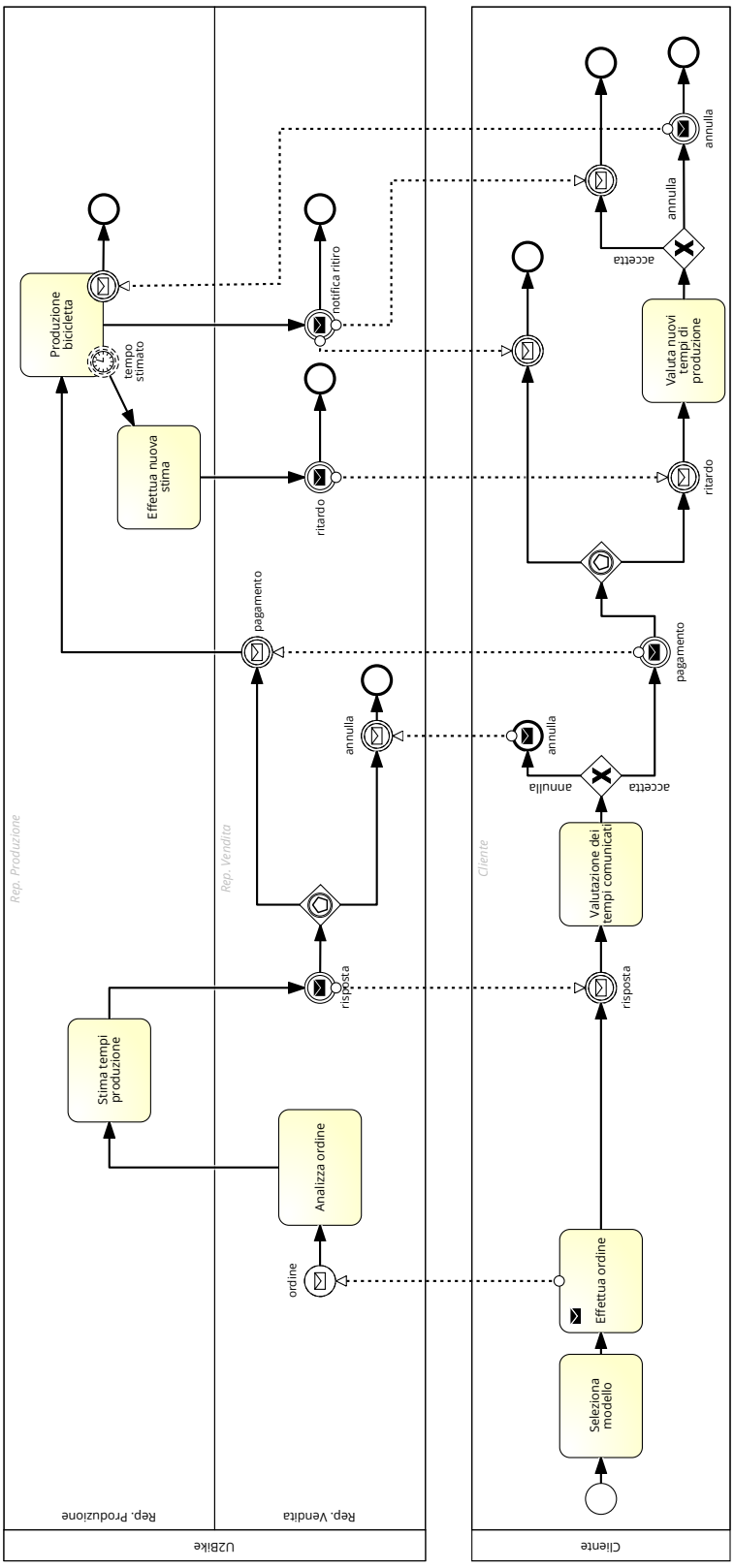


Figura 7.19: *U2Bike Ordine*

7.20 U2Bike - variazione

Si vuole progettare il sistema informativo del produttore e rivenditore di biciclette U2Bike. Il sistema deve permettere agli utenti di richiedere la riparazione di una bicicletta.

Il processo che descrive il processo di riparazione dell'azienda U2Bike è il seguente. Il processo inizia quando il cliente descrive il malfunzionamento della sua bicicletta e invia una richiesta di riparazione a U2Bike. La richiesta viene ricevuta dal servizio clienti che la analizza; passa poi all'officina che stima i costi di riparazione. Il servizio clienti invia risposta al cliente con tale stima. Il cliente può quindi decidere se procedere o annullare. Nel caso proceda, consegna la bicicletta. Una volta che il servizio clienti ha ricevuto la bicicletta, l'officina inizia il processo di riparazione. Per prima cosa, l'officina analizza lo stato del veicolo. A questo punto valuta se la stima di costo iniziale era corretta. In caso di aggiornamento invia un messaggio al cliente notificando il nuovo costo. Il cliente può quindi decidere se continuare o annullare la riparazione. Nel caso in cui invece il cliente accetta il nuovo preventivo o se la stima iniziale era corretta, l'officina esegue la riparazione, terminata la quale il cliente riceve una notifica dal servizio clienti per ritirare la sua bicicletta ed entrambi i processi terminano.

Realizzare il diagramma BPMN del processo di riparazione di una bicicletta, modellando sia il cliente che l'azienda (Figura 7.20).

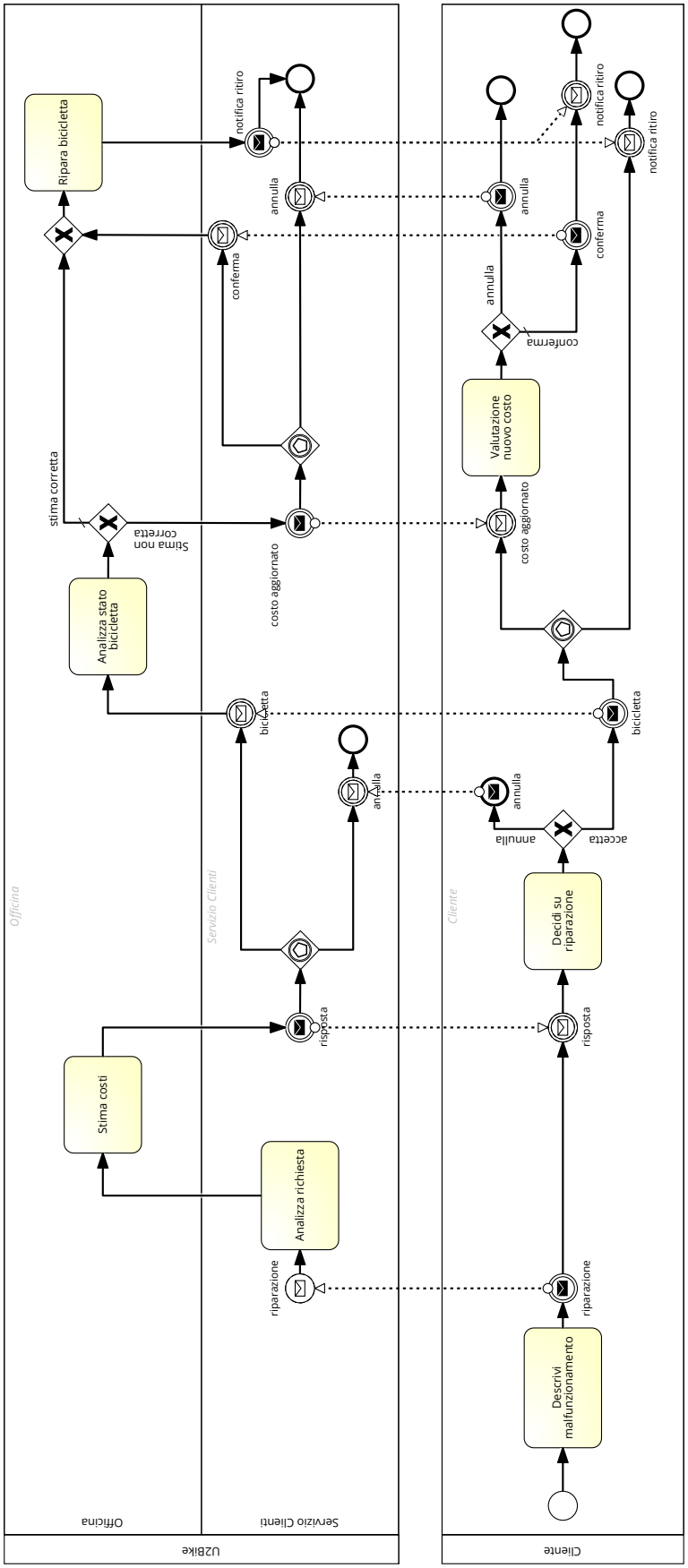


Figura 7.20: *U2Bike Riparazione*

7.21 CondominiumManagement

Si vuole progettare il sistema informativo della società CondominiumManagement che si occupa dell'amministrazione di condomini.

Descrivere il processo di pagamento di una rata usando la notazione BPMN. Il processo inizia quando l'amministratore inserisce una nuova rata da pagare nel sistema. La notifica di inserimento viene quindi inviata al singolo condomino che analizza la richiesta di pagamento.

Se il condòmino non riscontra errori procede direttamente con il pagamento. Nel caso notasse degli errori può inviare una richiesta all'amministratore descrivendo il problema riscontrato. L'amministratore, ricevuta la richiesta di verifica, esamina la stessa e re-invia la richiesta di pagamento eventualmente aggiornata (unica comunicazione sia in caso di modifica che in caso di conferma). Rimane quindi in attesa del pagamento del condomino che, ricevuta la nuova richiesta deve procedere al pagamento.

La richiesta di verifica può essere inviata entro due settimane, passate le quali l'amministratore ignora eventuali richieste e rimane solo in attesa del pagamento. Dal momento in cui viene inviata la richiesta di pagamento iniziale, il condomino ha due mesi per effettuare il pagamento. Se entro tale termine il pagamento non è stato effettuato, l'amministratore invia un sollecito. Alla ricezione del sollecito il condomino blocca ogni attività in corso (analisi della richiesta, pagamento o attesa di verifica) e deve procedere al pagamento con mora. Quando il pagamento con o senza mora viene effettuato la società invia una ricevuta al cliente e il processo termina (Figura 7.21).

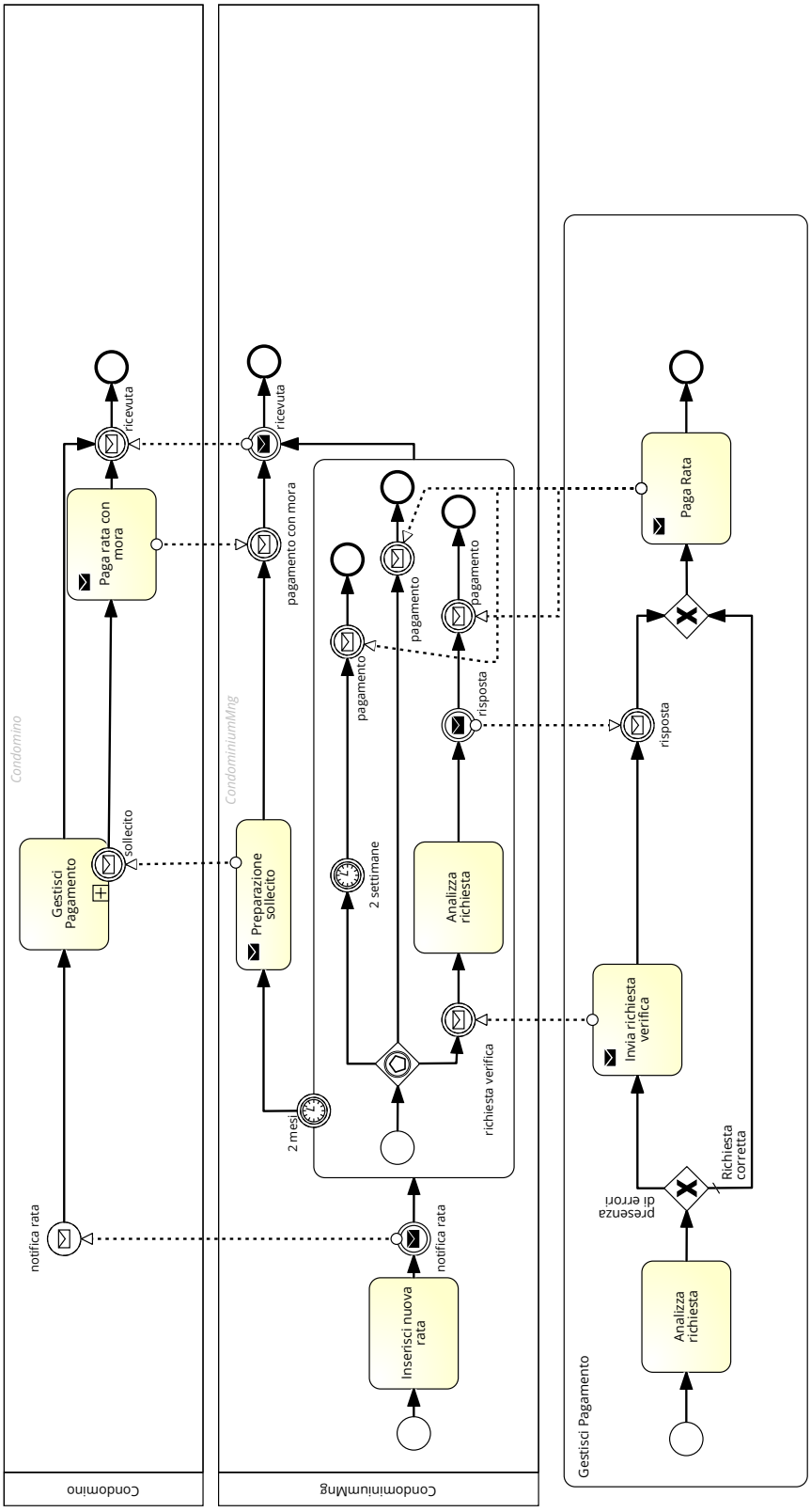


Figura 7.21: *CondominiumManagement*

7.22 Gestione di un ordine - Utilizzo di transazioni

Un'azienda manifatturiera vuole modellare con BPMN il processo di gestione di un ordine. Il processo inizia con una richiesta di ordine mandata dal cliente. L'ufficio amministrativo riceve l'ordine e deve gestirlo. Prima di tutto, questo ufficio chiede al Magazzino di prelevare gli articoli richiesti. Se ci sono articoli non disponibili il Magazzino deve mandare un ordine ai fornitori. Nel caso in cui gli articoli non arrivano entro 5 giorni l'ordine viene cancellato, una notifica di cancellazione viene mandata al cliente e gli articoli prelevati vengono resi nuovamente disponibili. Se invece tutto va bene, gli articoli insieme all'ordine sono spediti all'ufficio logistico che è responsabile di preparare il pacco e mandare le merci al cliente. Una volta che la merce è stata spedita, il cliente può restituire la merce entro sette giorni. Se succede, l'amministrazione prende nota del reso del cliente e manda indietro la merce al magazzino. Se il cliente non restituisce la merce, l'ufficio amministrativo chiude il processo. (Si chiede di non modellare il cliente). (Figura 7.22)

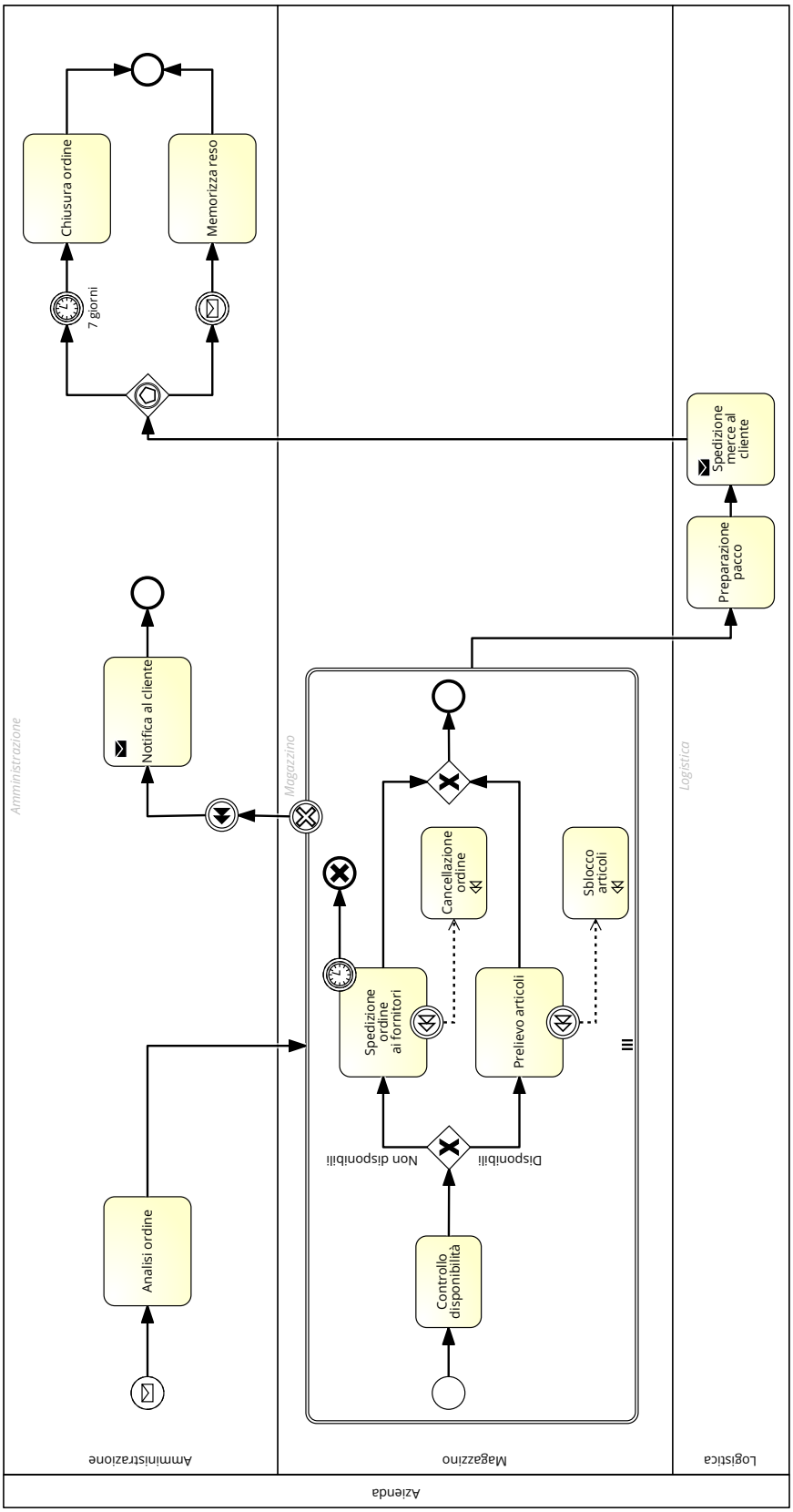


Figura 7.22: *Gestione di un ordine*

7.23 Rimborso spese - Utilizzo di transazioni

Un'azienda vuole modellare con BPMN il processo di rimborso delle spese di viaggio ai dipendenti. Quando un dipendende torna da un viaggio deve mandare all'ufficio del personale un modulo contenente la lista delle spese sostenute. Non appena l'ufficio del personale riceve il modulo controlla se l'impiegato esiste all'interno del sistema. Se non è presente, vengono inseriti i dati dell'impiegato nell'anagrafe dipendenti. A questo punto, la lista delle spese viene analizzata per l'approvazione. Se il totale delle spese è inferiore a 1000 euro, il rimborso è automaticamente approvato mentre se il totale supera i 1000 euro sono richieste ulteriori verifiche. Se si trovano delle irregolarità e il rimborso viene rifiutato, il dipendente riceve via mail una notifica. In caso di approvazione invece il rimborso viene depositato automaticamente sul conto del dipendente. In ogni momento durante la review, il dipendente può mandare all'ufficio una Richiesta di modifica dell'importo da rimborsare. In questo caso la richiesta viene registrata e il modulo deve essere rianalizzato. Inoltre, se il modulo non viene analizzato entro 30 giorni, il processo viene interrotto. Se questo succede quando il rimborso è già stato accreditato sul conto del dipendente, i soldi devono essere stornati dal conto. (Si chiede di non modellare il dipendente).(Figura 7.23)

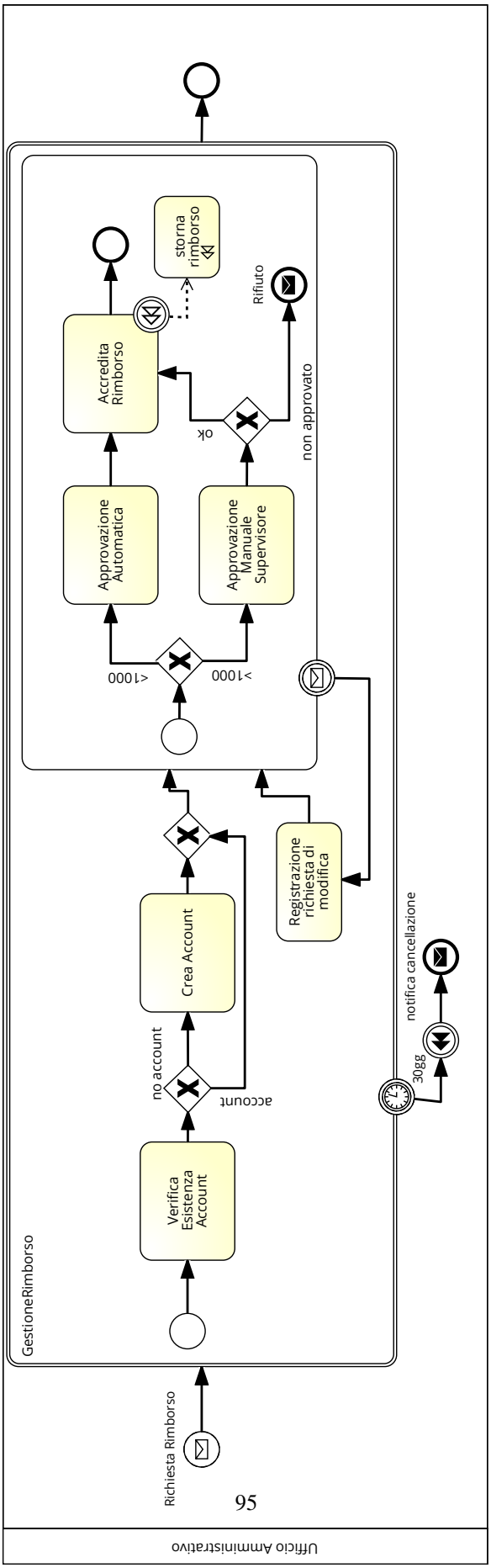


Figura 7.23: *Rimborso viaggi aziendali*

7.24 CRM - Utilizzo di transazioni

Ogni sabato mattina alle 03:00, un processo automatico si attiva e esegue delle operazioni di routine sul database dell'azienda manifatturiera Texile. Il processo esegue il backup intero del database e in particolare controlla sequenzialmente tutte le tuple della tabella "Fatture Non Pagate" (che contiene le fatture scadute e non pagate) in cerca di nuove tuple inserite o aggiornate dopo l'ultimo backup. Se non ci sono nuove tuple, il processo analizza il sistema CRM come descritto più avanti. Se la tabella contiene nuove tuple, i dati dei clienti che non hanno pagato le fatture sono inseriti in una nuova tabella e il sistema controlla se è la prima volta che il cliente non ha pagato una fattura in tempo. I clienti che non hanno pagato in tempo diverse volte sono memorizzati in una nuova tabella "Messaggi". Alla fine di questa procedura la tabella "Messaggi" (se è stata creata) viene mandata all'Ufficio Contabilità. Le attività descritte devono essere completate entro le 14:30, altrimenti, un avviso è mandato al supervisore senza interrompere il processo. Se l'analisi di nuove tuple finisce in modo eccezionale, la tabella "Messaggi" viene cancellata e il processo finisce. Una volta che tutte le attività descritte sono state completate, viene controllato il sistema CRM al fine di controllare se tutti i pagamenti relativi a fatture pagate in ritardo ricevuti nell'ultima settimana sono stati correttamente registrati. Se da questo controllo sono individuate delle inconsistenze i dati CRM sono aggiornati. L'intero processo deve essere completato in 13 ore, altrimenti una notifica viene mandata al Supervisore (sempre senza interrompere il processo).(Figura 7.24)

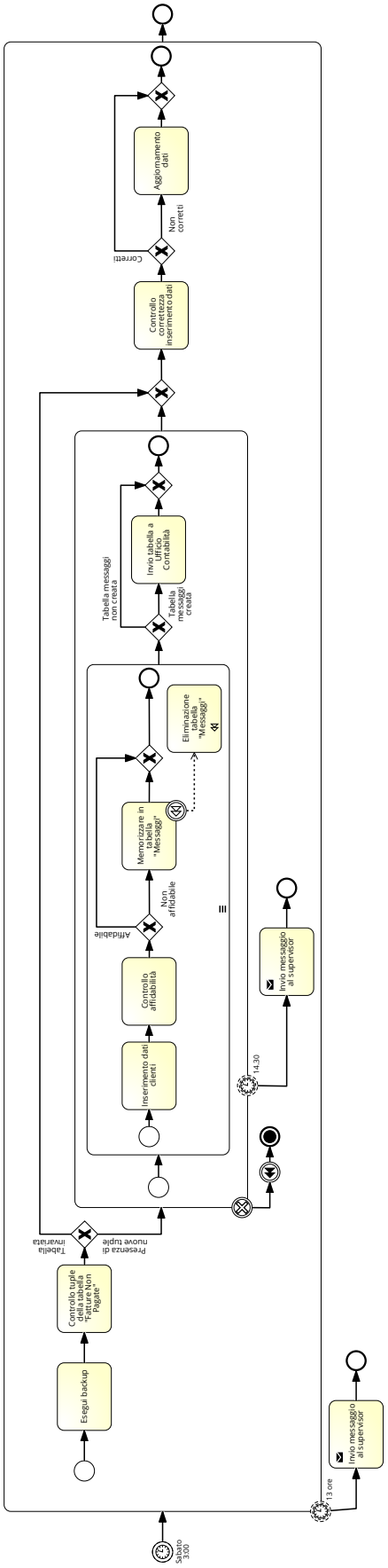


Figura 7.24: *Backup Fatture Non Pagate*