



# UNIVERSITÀ DI PISA

DIPARTIMENTO DI INFORMATICA  
Data Science and Business Informatics

Business Process Modeling

Progetto P35: Ricerca

Simone Di Luna 544322

Anno accademico 2022/2023

# Indice

<b>1</b>	<b>Introduzione</b>	<b>1</b>
<b>2</b>	<b>Business Process Identification</b>	<b>1</b>
<b>3</b>	<b>Business Process Modeling</b>	<b>2</b>
3.1	Coordinator . . . . .	2
3.2	Collaborators . . . . .	5
3.3	Evaluation process . . . . .	6
3.4	Collaborazione e interazione tra processi . . . . .	8
<b>4</b>	<b>Verification</b>	<b>11</b>
4.1	Verifica dei singoli processi . . . . .	11
4.2	Workflow system . . . . .	12

## 1 Introduzione

Lo scenario da modellare riguarda la sottomissione di un progetto di ricerca per l'ottenimento di un finanziamento. La traccia del problema è molto generica e lascia spazio a diverse interpretazioni. Per quanto possibile, si è cercato di rimanere fedele agli obiettivi illustrati, senza introdurre troppi elementi aggiuntivi.

In particolare, per capire il processo cognitivo che ha condotto alla produzione dei modelli che verranno presentati nelle prossime pagine, si ritiene utile chiarire l'interpretazione che si è data di alcuni passaggi del testo.

Ma prima una premessa: benché sia possibile ipotizzare che un unico processo di business possa modellare lo scenario in esame, è, probabilmente, più naturale considerare una collaborazione tra processi differenti. Per tale motivo, si è deciso di utilizzare il linguaggio diagrammatico BPMN in luogo di quello EPC.

## 2 Business Process Identification

Se accettano di collaborare, i responsabili di sede devono inviare al coordinatore le loro modifiche e integrazioni alla proposta. Il coordinatore prepara una nuova bozza che invia ai responsabili e il procedimento viene iterato fino a quando i responsabili non hanno più modifiche da richiedere oppure rifiutano di collaborare.

I due responsabili di sede sembrano agire congiuntamente, come un soggetto unico. Il testo, inoltre, pare sottintendere che entrambi i collaboratori debbano approvare la bozza di progetto finale affinché questa possa essere presentata ufficialmente. Pertanto, non si è ritenuto utile modellare i due collaboratori in *lane* o *pool* separati, dal momento che questi sarebbero assolutamente identici. Alla base di questa scelta, c'è anche la volontà di semplificare il modello finale. Infatti, se volessimo mantenere i due responsabili in *pool* separati, per ogni comunicazione diretta dal coordinatore ai responsabili, dovremmo

necessariamente utilizzare due attività identiche all'interno di un *parallel gateway*. Questo per due ragioni: la prima è che, nel linguaggio BPMN, a ciascuna attività può essere collegato al più un *message flow*; la seconda è che l'ordine con il quale i responsabili di sede vengono contattati non dovrebbe avere importanza. Pertanto, se si volesse seguire questa linea, si dovrebbe necessariamente semplificare l'*orchestration* dei singoli processi, portando inevitabilmente ad una perdita di espressività del modello.

Per quanto riguarda, invece, il processo di valutazione a cui deve essere sottoposta la bozza di progetto una volta presentata, la traccia fornisce pochissime indicazioni:

Se viene raggiunta una proposta condivisa, il coordinatore inserisce il testo della proposta e tutte le informazioni necessarie nel sistema elettronico di valutazione [...].

È, tuttavia, improbabile ipotizzare che una richiesta di finanziamento inerente un progetto di ricerca venga valutata in modo completamente automatizzato. Pertanto, si è ipotizzato l'esistenza di un processo di valutazione semplificato – a cui, talvolta, si farà riferimento con il termine “commissione” – che potesse sposarsi bene con lo scenario presentato.

In conclusione, il modello a cui si è giunti prevede la collaborazione fra tre partecipanti, rappresentati da altrettanti pool: *coordinator*, *collaborators*, *evaluation system*.

### 3 Business Process Modeling

Iniziamo con l'analizzare dapprima il modo con il quale le attività del processo del coordinatore vengono orchestrate. Successivamente si passa alla disamina dei processi dei collaboratori e del sistema di valutazione.

#### 3.1 Coordinator

Il processo inizia con l'intuizione da parte del coordinatore di una possibile idea di progetto. Prima della stesura della bozza, egli prende visione dei requisiti formali previsti per la sottomissione della stessa, dopodiché procede a redigere una proposta di collaborazione avente come allegato la suddetta bozza.

Prima di inoltrare la richiesta ai responsabili di sede, il coordinatore inizializza la procedura di sottomissione del progetto. Questa attività è necessaria per istanziare il processo *evaluation system* e fare in modo che venga attivato anche quando i richiedenti non raggiungono mai una proposta condivisa. Come si può vedere dalla figura 1, queste attività sono eseguite in modo sequenziale.

Dopo aver inviato una richiesta formale di collaborazione, il coordinatore resta in attesa di ricevere una risposta da parte dei responsabili di sede (vedi figura 2). Nel modello, questa attesa viene rappresentata utilizzando un *event based gateway* che è

collegato a due *catch event* distinti: se la proposta di collaborazione viene rifiutata, il processo si interrompe con un *message end event* che permette all'istanza del sistema di valutazione di giungere a corretto compimento; se invece la collaborazione viene accettata, il coordinatore attende che i responsabili di sede comunichino le loro impressioni sulla bozza. L'attesa qui viene interrotta dal primo dei tre *catch event* che si verifica: *changes or additions proposed, draft accepted, draft rejected*.

Se i collaboratori richiedono modifiche e/o integrazioni alla proposta, il coordinatore dovrà valutarle e decidere se integrarle all'interno della bozza attuale oppure se rifiutarle (informando i responsabili di sede). Se accetta le modifiche presentategli, il coordinatore le integra all'interno di una nuova bozza che verrà nuovamente sottoposta al giudizio dei collaboratori. Queste tre attività – *evaluate changes, create a new draft, propose new draft* – sono all'interno di un loop che viene eseguito zero o più volte e costituiscono una vera e propria trattativa tra coordinatore e collaboratori che avviene prima della sottomissione del progetto.

Il *catch event* “*draft rejected*” può verificarsi solo se il loop in questione viene eseguito almeno una volta. A prescindere che la trattativa fallisca per volontà del coordinatore o dei collaboratori, il processo va a concludersi con le stesse modalità previste per il rifiuto della collaborazione. In entrambi i casi, infatti, il *sequence flow* si ricongiunge con uno *XOR JOIN* a quel ramo del diagramma.

Infine, se i collaboratori accettano la bozza, il coordinatore procede ad inserire la proposta e tutte le informazioni necessarie nel sistema elettronico di valutazione.

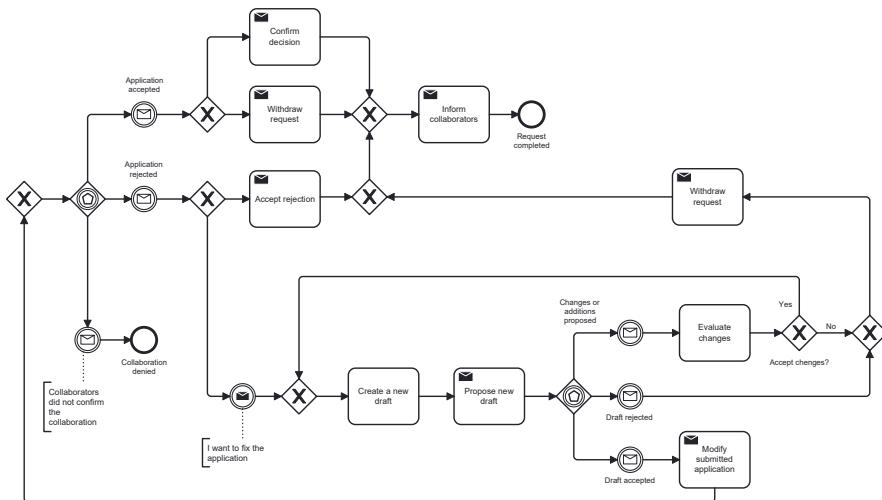
Una volta sottomessa la proposta, il coordinatore attende l'esito della commissione (vedi figura 3). Ancora una volta è un *event based gateway* a modellare l'attesa. Gli esiti possibili sono tre: *application accepted, application rejected, collaboration not confirmed*.

Se la proposta viene accettata, il coordinatore deve comunicare se intende confermare la sua richiesta oppure se ritirarla (*XOR SPLIT*). Questa opzione è stata prevista per meglio aderire ai requisiti della traccia, la quale prescrive che:

In ogni momento il coordinatore può decidere di ritirare la proposta.

In questo modo, infatti, garantiamo che il coordinatore, oltre ad avere l'ultima parola prima della sottomissione del progetto, ce l'abbia anche per quanto concerne l'effettiva accettazione in caso di valutazione positiva.

In caso di domanda rigettata, invece, il coordinatore dovrà comunicare se intende correggerla ovvero se accettare la decisione della commissione. Questa scelta viene modellata attraverso uno *XOR SPLIT*. Se il coordinatore sceglie di provare a correggere la proposta, lo scenario che viene a configurarsi è lo stesso che si è presentato all'inizio della descrizione di questo processo di business. Il coordinatore crea una (nuova) bozza

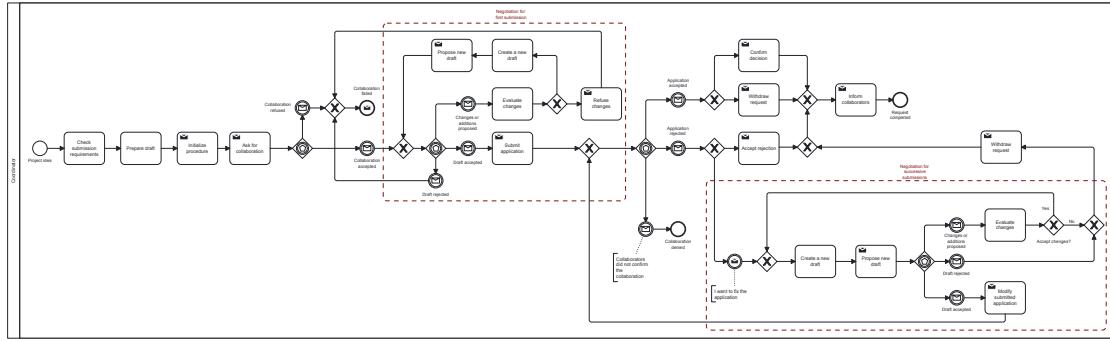


**Figura 3.** Coordinator: esito richiesta e successive sottomissioni.

di progetto che invia ai collaboratori e resta in attesa di risposta. Se questi propongono delle modifiche, il coordinatore decide se accettarle o rifiutarle. Se è d'accordo, le integra all'interno di una nuova bozza da inviare ai responsabili di sede, che se accettata, porta il coordinatore a immettere nel sistema informatico la nuova domanda. Anche in questo caso ci troviamo davanti ad uno ad un loop che può essere eseguito zero o più volte. Tuttavia, ci sono due importanti differenze rispetto al precedente. La prima è che in caso di fallimento della trattativa, per far sì che l'*evaluation process* giunga correttamente conclusione, il coordinatore deve ritirare esplicitamente la richiesta di finanziamento. La seconda è che, in questo caso, c'è anche un altro loop più grande che ingloba il precedente. Infatti, quando i richiedenti giungono ad una proposta condivisa, il *sequential flow* in uscita dall'attività *modify submitted application* si ricollega al flusso uscente da *submit application*. Questo ciclo, quindi, oltre alle attività presenti nell'altro loop, comprende anche il task *modify submitted application*.

Prima, quando si presentato l'*event based gateway* posto subito dopo l'attività *submit application*, si è volutamente tralasciato di menzionare uno dei tre *catch event*. Questo si è fatto perché si vuole sottolineare un fatto importante: esiste la possibilità che i collaboratori, quando contattati dal sistema elettronico, non confermino l'esistenza della collaborazione. Ciò, però, può accadere solo in seguito alla prima sottomissione da parte del coordinatore; per le modifiche successive, infatti, il sistema non chiede ai responsabili di sede di confermare nuovamente la collaborazione. In definitiva, quindi, anche se l'evento *collaboration not confirmed* si trova all'interno del loop più grande, esso può verificarsi solamente una volta.

Per concludere, in tutti gli scenari illustrati – successivi all'attività di sottomissione del progetto – è previsto che prima della conclusione del processo, il coordinatore comunichi l'esito finale della richiesta ai collaboratori. La figura 4 mostra l'intero processo del coordinatore.

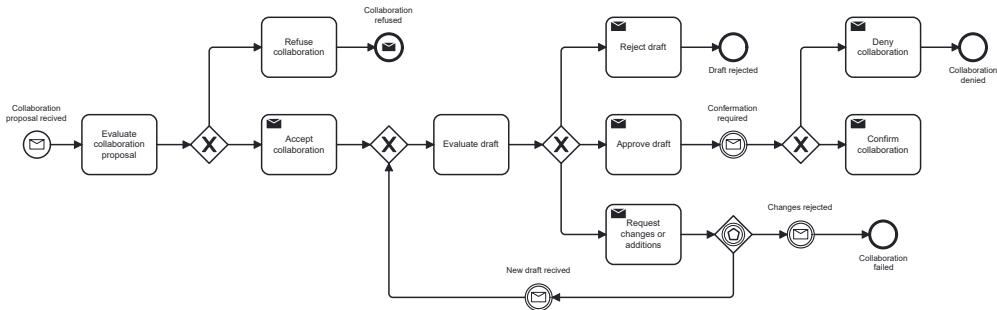


**Figura 4.** Coordinatore: processo completo.

### 3.2 Collaborators

Il processo dei responsabili di sede inizia con la ricezione della proposta di collaborazione. Come prima cosa, i collaboratori valutano la proposta e decidono se rifiutarla o accettarla, trasmettendo la relativa comunicazione al coordinatore. Come si vede dalla figura 5, questa scelta viene implementata con uno *XOR JOIN*. Nel primo caso, il processo giunge subito a conclusione.

Se, invece, accettano di collaborare, i responsabili di sede, dopo aver valutato attentamente la bozza, sono chiamati a decidere se approvarla così com'è ovvero se apportarvi delle modifiche e/o integrazioni. L'*exclusive gateway* prende in considerazione anche una terza alternativa che conduce alla terminazione del processo: il rifiuto della bozza. È molto improbabile che i responsabili di sede rifiutino la bozza subito dopo aver confermato la collaborazione. Questa opzione verrà intrapresa, verosimilmente, quando giungono nuove bozze. Infatti, l'attività *request changes or additions* può innescare una trattativa con il coordinatore; questa è rappresentata tramite un loop composto da due attività *evaluate draft*, *request changes or additions*. Affinché ciò possa accadere, è necessario che il coordinatore accetti i suggerimenti dei collaboratori, inviando una nuova bozza modificata; in caso contrario, il processo giunge a conclusione. Anche in questo caso, è un *event based gateway* a modellare l'attesa.



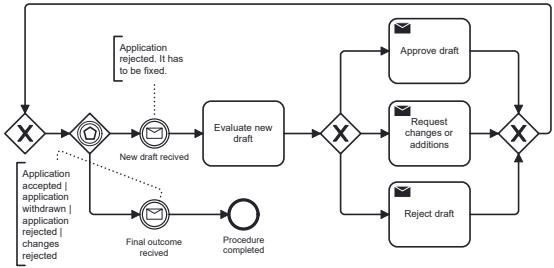
**Figura 5.** Collaboratori: prima parte del business process.

Se i collaboratori accettano la bozza, il sistema di valutazione invia loro una richiesta di conferma di accettazione della collaborazione: se la negano, il processo si conclude; se, invece, confermano, restano in attesa di una comunicazione da parte del coordinatore che li aggiorni sullo stato della pratica (vedi figura 6).

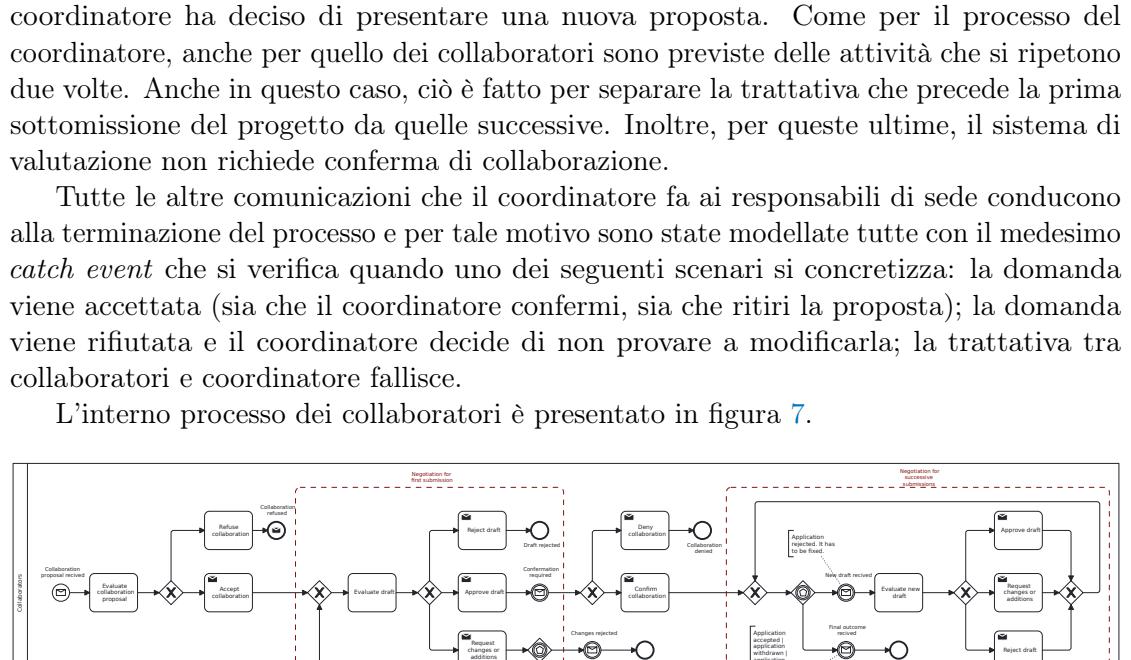
Se i responsabili di sede ricevono una nuova bozza, è implicito che la commissione non ha accolto la domanda e che il coordinatore ha deciso di presentare una nuova proposta. Come per il processo del coordinatore, anche per quello dei collaboratori sono previste delle attività che si ripetono due volte. Anche in questo caso, ciò è fatto per separare la trattativa che precede la prima sottomissione del progetto da quelle successive. Inoltre, per queste ultime, il sistema di valutazione non richiede conferma di collaborazione.

Tutte le altre comunicazioni che il coordinatore fa ai responsabili di sede conducono alla terminazione del processo e per tale motivo sono state modellate tutte con il medesimo *catch event* che si verifica quando uno dei seguenti scenari si concretizza: la domanda viene accettata (sia che il coordinatore confermi, sia che ritiri la proposta); la domanda viene rifiutata e il coordinatore decide di non provare a modificarla; la trattativa tra collaboratori e coordinatore fallisce.

L'interno processo dei collaboratori è presentato in figura 7.



**Figura 6.** Collaborators: seconda parte.



**Figura 7.** Collaborators: processo completo.

### 3.3 Evaluation process

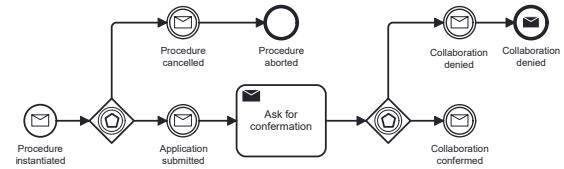
Il processo viene azionato da un *message start event* che si concretizza quando il coordinatore avvia la procedura di sottomissione della domanda di finanziamento (figura 8). Come già spiegato in precedenza, ciò serve a sincronizzare i tre processi. Tuttavia, a seguito di questo evento, nessuna attività viene posta in essere da parte del valutatore, che infatti rimane in attesa di una delle due possibili azioni che il coordinatore può compiere: presentazione effettiva della domanda con il progetto in allegato, ovvero, richiesta di annullamento della procedura, la quale porta alla conclusione prematura del processo.

Una volta che il progetto è stato presentato, il sistema contatta i responsabili di sede per richiedere di confermare la collaborazione con il sottoscrittore. In caso di risposta negativa, la procedura viene abortita e il processo termina con il sistema che provvede ad informare il coordinatore dell'accaduto. Se invece la collaborazione viene confermata, la proposta viene sottoposta al giudizio della commissione (vedi figura 9). L'attività di valutazione potrebbe essere composta da diversi task, tuttavia per semplificare il modello, si è deciso di non utilizzare un sottoprocesso.

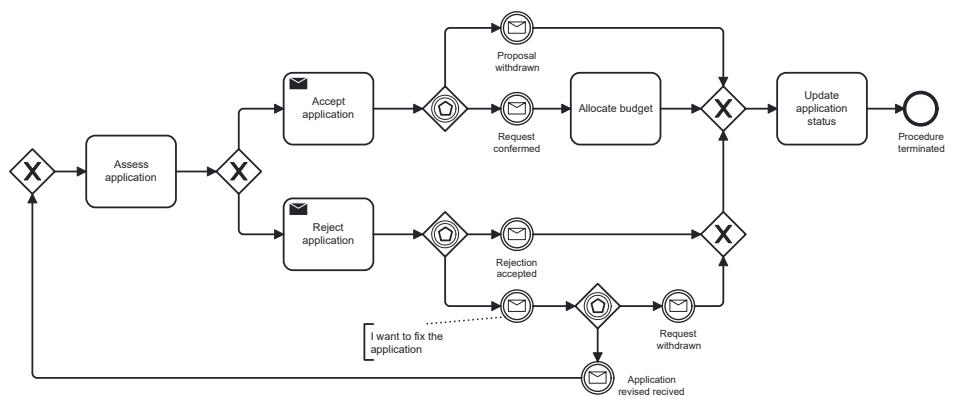
L'esito della valutazione, quando disponibile, viene comunicato al coordinatore del progetto, dopodiché, il sistema resta in attesa di ricevere una risposta da parte di quest'ultimo. In caso di rifiuto, se il coordinatore decide di non procedere con la sottoscrizione di un nuovo progetto, la pratica viene archiviata; se, invece, il coordinatore esprime la sua volontà di presentare una nuova proposta, il sistema resta in attesa di ricevere il progetto revisionato oppure una richiesta di annullamento. In quest'ultimo caso, il *sequential flow* si ricongiunge con uno *XOR JOIN* al ramo che viene intrapreso in seguito all'accettazione del rifiuto da parte del coordinatore. Altrimenti, la nuova proposta aziona un loop composto da due attività *assess application* e *reject application*, la prima viene eseguita almeno una volta, la seconda zero o più.

Quando la domanda viene accolta, il sistema attende la conferma da parte del coordinatore. Quando questa arriva, il sistema provvede ad allocare il budget per finanziare il progetto. In ogni caso, prima di chiudere definitivamente la pratica, viene aggiornato il suo stato.

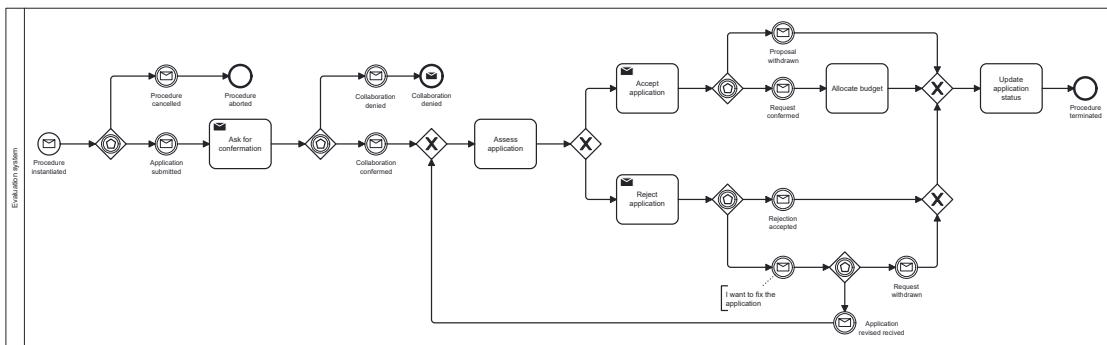
La figura 10 mostra il processo di valutazione nella sua interezza.



**Figura 8.** Evaluation system: procedura inizializzata.



**Figura 9.** Evaluation system: valutazione del progetto.



**Figura 10.** Evaluation system: processo completo.

### 3.4 Collaborazione e interazione tra processi

I processi presentati nelle pagine precedenti sono stati costruiti in modo da produrre moduli strutturalmente compatibili. Per quanto concerne il linguaggio BPMN, questo significa che per ogni messaggio che un’attività/evento di un pool invia, c’è esattamente un’altra attività/evento all’interno di un altro pool che lo riceve. Il *collaboration diagram* rappresentato nella figura 11 permette di visualizzare non solo il modo col quale i singoli processi orchestrano le loro attività, ma anche l’interazione fra i tre partecipanti attraverso lo scambio di messaggi.

Inoltre, BPMN 2.0 ha introdotto una nuova tipologia di diagramma chiamata *choreography diagram*. Il suo punto di forza è che, utilizzando un unico diagramma, permette di focalizzare l’attenzione solamente sull’interazione fra i partecipanti, mettendo da parte l’*orchestration* che avviene all’interno dei singoli processi. Si è pertanto deciso di adottare anche questa prospettiva, in modo da poter ripercorrere facilmente ed in modo unificato, tutti possibili scenari che possono verificarsi quando i tre partecipanti interagiscono tra di loro (12).

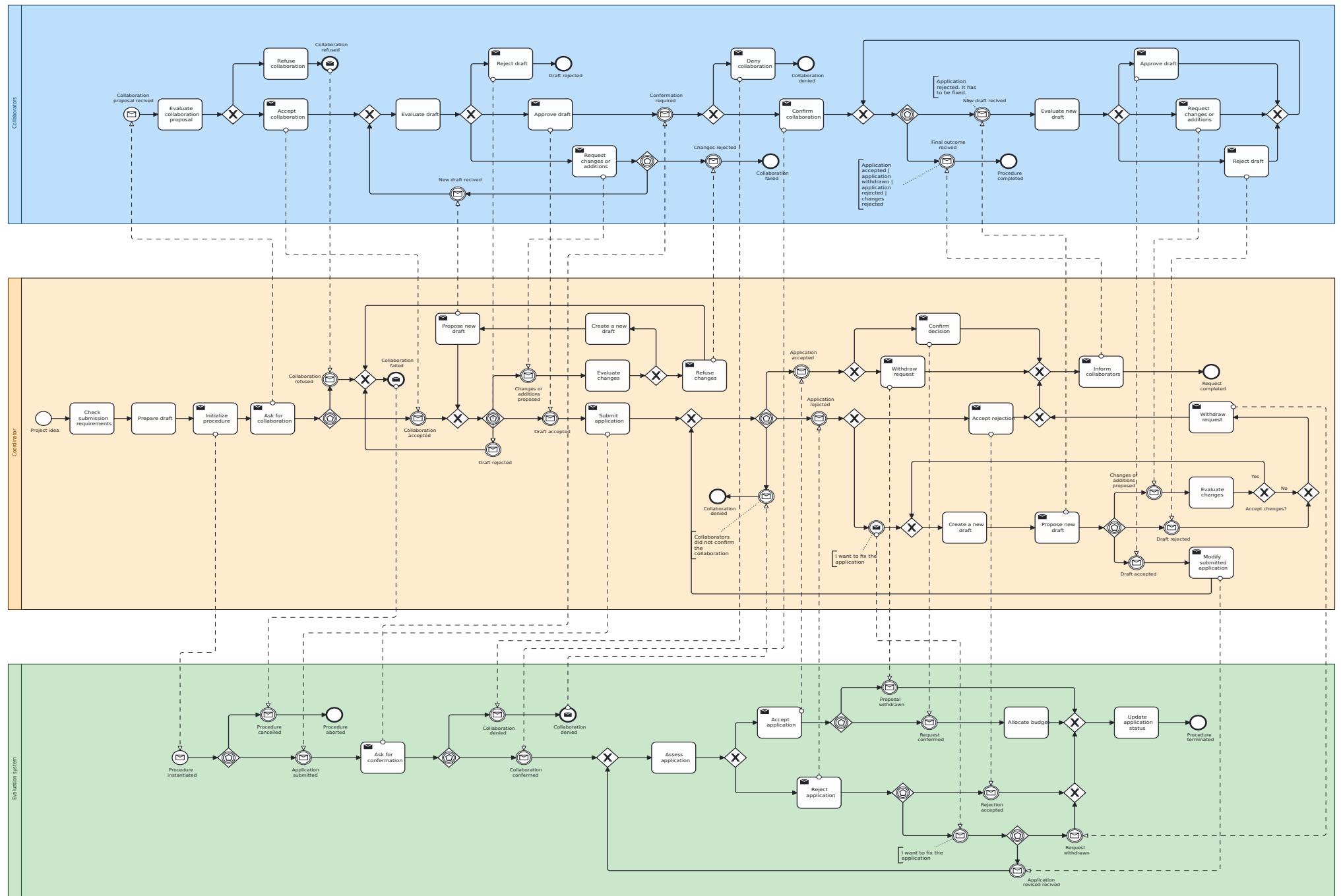
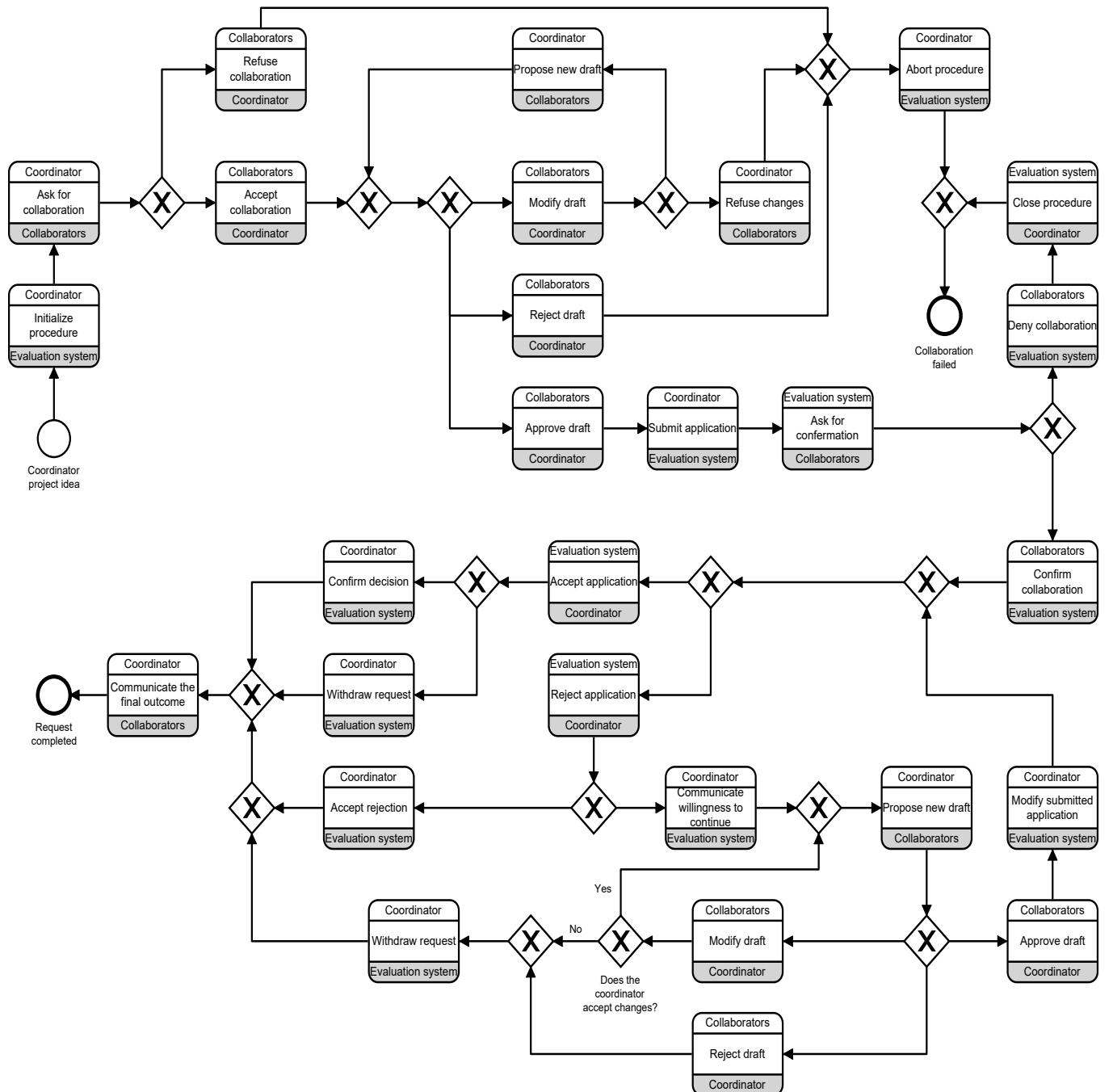


Figura 11. Collaboration diagram



**Figura 12.** Choreography diagram

## 4 Verification

Tutti i modelli di business presentati nelle pagine precedenti sono stati trasformati in petri net ed analizzati con l'ausilio di vari software e tool al fine di verificare la correttezza della loro esecuzione. In particolare, gli strumenti utilizzati sono stati: *WoPeD*, *Woflan* e la libreria *pm4py* di python.

La trasformazione dei diagrammi BPMN è avvenuta in modo tale da poter ricavare delle workflow net. In particolare:

- i *sequential flow* sono diventati delle piazze;
- le attività e gli eventi sono stati trasformati in transizioni;
- per gli XOR split/join si è adottata una conversione 1:1 utilizzando la sintassi *sugared* così da mantenere la rete più compatta;
- gli *event based gateway* e le piazze ad esso direttamente collegate (che, come detto, nel diagramma BPMN costituiscono gli archi) sono stati invece sostituiti da un'unica piazza collegata direttamente alle transizioni relative ai *catch event*;
- per ciascuna net, è stata aggiunta un'unica piazza iniziale con archi uscenti diretti verso gli *start event* (nel nostro caso tutti i diagrammi hanno un unico evento iniziale) e un'unica piazza finale avente come preset le transizioni corrispondenti agli *end event*.

Infine, come si vedrà in seguito, per il workflow system che va a tradurre il *collaboration diagram*, i *message flow* sono stati trasformati in piazze e sono state anche aggiunte la *start transition* – avente come postset le piazze iniziali delle singole reti – e la *end transition* – avente come preset le piazze finali delle singole reti – con i relativi *initial* e *final place*.

### 4.1 Verifica dei singoli processi

Le workflow net relative ai singoli processi, prese singolarmente, presentano tutte le stesse identiche proprietà strutturali e comportamentali. Per tale motivo, l'analisi che segue si applica a tutte e tre le reti.

Come si può vedere dall'immagine 13, le workflow net sono *free-choice*. Questo perché, all'interno di ciascuna di esse, comunque prendiamo due transizioni, il loro preset risulta essere uguale o disgiunto. In particolare, si tratta di *S-net* perché ogni transizione ha esattamente un unico *input* e un unico *output place*. Per tale motivo, ciascuna workflow net è dotata di un unico *S-component* che ingloba tutti i nodi della rete e che perciò la rende anche *S-coverable*. Se inserissimo la *reset transition*, che per ciascuna rete collega la piazza finale con quella iniziale, potremmo notare che non sono presenti né *PT-handles*, né *TP-handles*, pertanto le workflow net risultano essere *wellstructured*.

Siccome si tratta di *S-net*, i singoli sistemi (e quindi i relativi *short-circuited system*) sono *1-bounded* e perciò *safe*. Le reti non hanno una dimensione eccessiva, pertanto è possibile analizzarle direttamente attraverso il loro *reachability graph* (che non verrà

**Tabella 1.** Confronto tra workflow net.

	Coordinator	Collaborators	Evaluation system
N° places	48	34	24
N° transitions	58	41	29
N° arcs	116	82	58
N° S-components	1	1	1
N° Places not covered	0	0	0
Free-choice	✓	✓	✓
Wellstructuredness	✓	✓	✓
Boundedness	✓	✓	✓
Liveness	✓	✓	✓
Soundness	✓	✓	✓

mostrato per questioni di spazio). Tutte le transizioni appaiono all'interno dei rispettivi grafi (no *dead task*) e da qualsiasi nodo partiamo, possiamo sempre raggiungerne un altro che abilita la transizione che desideriamo eseguire (a patto che inseriamo anche la *reset transition*). Di conseguenza possiamo concludere che i sistemi sono anche *live*. Dal momento che gli *short-circuited system* sono sia *live* che *bounded*, per il *main theorem* possiamo concludere che le workflow net sono *sound*.

La tabella 1 sintetizza e mette a confronto le proprietà delle tre reti, presentando, per ciascuna di esse, anche alcune statistiche.

## 4.2 Workflow system

Come già accennato nel paragrafo 3.4, i processi dei tre partecipanti sono stati costruiti in modo da essere compatibili. Questo significa che, per ogni workflow net, possiamo ricavarci i relativi moduli, aggiungendo per ogni *message flow* presente nel diagramma BPMN corrispondente, una piazza con un arco in entrata o in uscita, a seconda che si tratti, rispettivamente, di un messaggio inviato o ricevuto, proveniente dalla/entrante nell'attività/evento coinvolto. Per ogni messaggio inviato da una transizione in un modulo, esiste esattamente un'altra transizione in un altro modulo pronta a riceverlo e viceversa. Per fare in modo, però, che la petri net risultante da questo processo sia una workflow net, è necessario aggiungere una *start* e una *end transition* ed un unico *input* e *output place* ad esse collegate. Il *workflow system* risultante (vedi figura 14) può così essere analizzato come è stato fatto per le singole workflow net. A causa della complessità della rete, *WoPeD* non riesce a completare l'analisi, pertanto sono stati usati *woflan* e la libreria di python *pm4py*.

Per quanto concerne le proprietà strutturali, c'è subito da rilevare che la rete non è *free-choice*. Questo è causato dagli *event based gateway*. Infatti, come è stato già detto, questo tipo di *flow object*, nelle petri net viene trasformato in una piazza con gli archi in uscita che entrano all'interno delle transizioni che rappresentano gli eventi esterni. Questi ultimi, quindi, hanno esattamente una piazza in comune nel loro preset (quella dello XOR

**Tabella 2.** Workflow system: tabella di sintesi.

Workflow system	
N° places	135
N° transitions	129
N° arcs	318
N° S-components	186
N° Places not covered	0
Free-choice	✗
Wellstructuredness	✗
Boundedness	✓
Liveness	✓
Soundness	✓

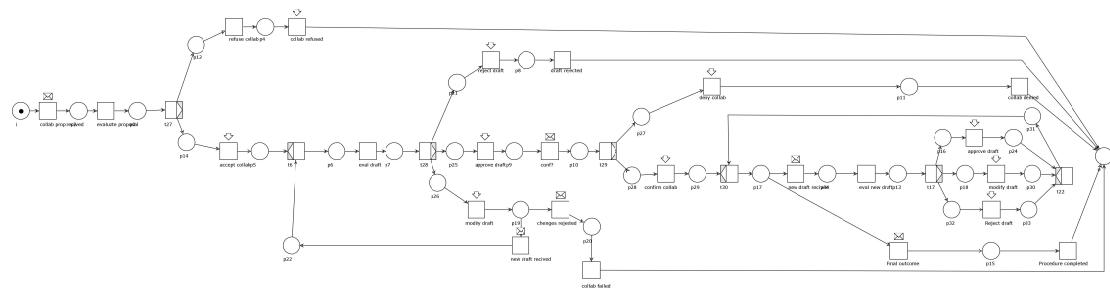
split隐式). Quando però aggiungiamo le piazze dell’interfaccia, ognuna di queste transizioni viene collegata al suo *place* di competenza su di essa e ciò fa sì che i preset di queste transizioni non siano né uguali, né disgiunti. In generale, si avrà una *free-choice violation* ogni coppia di *catch event* collegata a un *event based gateway*. Siccome, nel *collaboration diagram* in esame sono presenti 11 gateway di questo tipo, di cui 8 con due eventi esterni collegati e i restanti con 3, in totale dovremmo registrare  $8 \cdot \binom{2}{2} + 3 \cdot \binom{3}{2} = 17$  violazioni ed infatti, questo è proprio il numero esatto di *free-choice violation* che si è registrato nel *workflow system*.

Questo tipo di gateway è anche la causa per cui la workflow net non risulta essere *wellstructured*. In totale, infatti, sulla *short-circuited net* si sono registrati ben 257 *PT/TP-handles*. In ognuno di essi, sono coinvolte sempre almeno una piazza o una transizione che si sono venute a formare dopo la trasformazione dell’*event based gateway*.

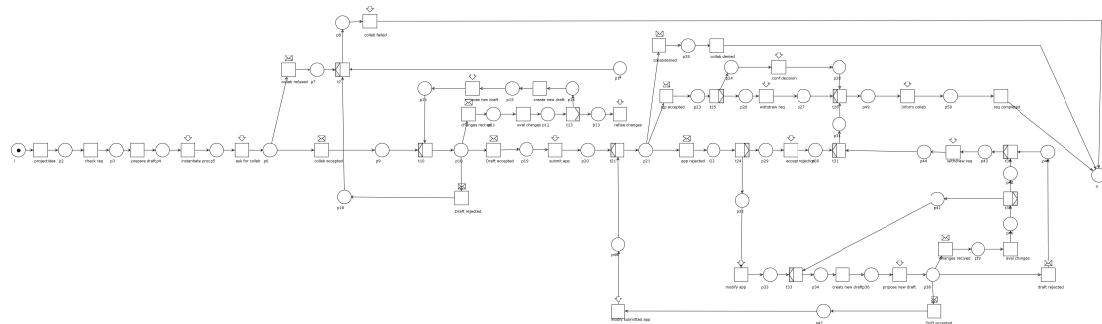
In totale, *Woflan* registra un insieme di ben 186 *S-components*, i quali coprono tutte le piazze della rete; per tale motivo la workflow net è *S-coverable*.

Passando invece alle proprietà dinamiche, il *coverability graph* è finito e pertanto coincide con il *reachability graph*, tuttavia a causa della *state explosion* risulta impossibile da analizzare.

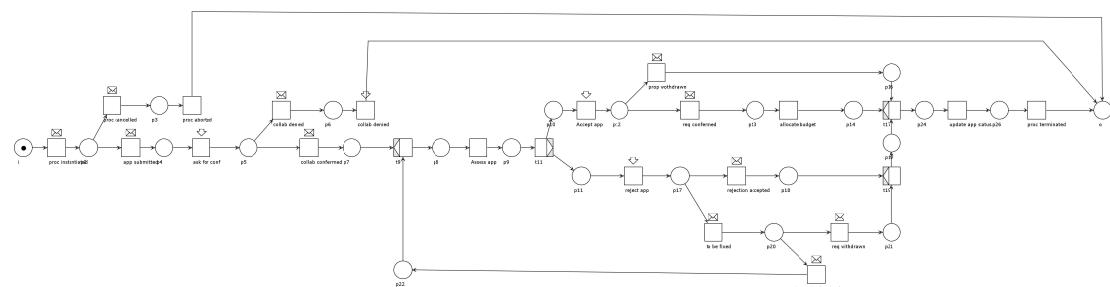
Nonostante la presenza di numerosi *PT/TP-handles* e la non *wellstructuredness*, il workflow system risulta essere comunque privo di *dead task*, *bounded* e *live* ( $N^*$ ) e di conseguenza anche *sound*. In conclusione, quindi, possiamo affermare che i diagrammi prodotti permettono una corretta interazione fra i tre partecipanti. La tabella 2 sintetizza le proprietà del workflow system appena discusso e presenta alcune statistiche.



**(a)** Collaborators.

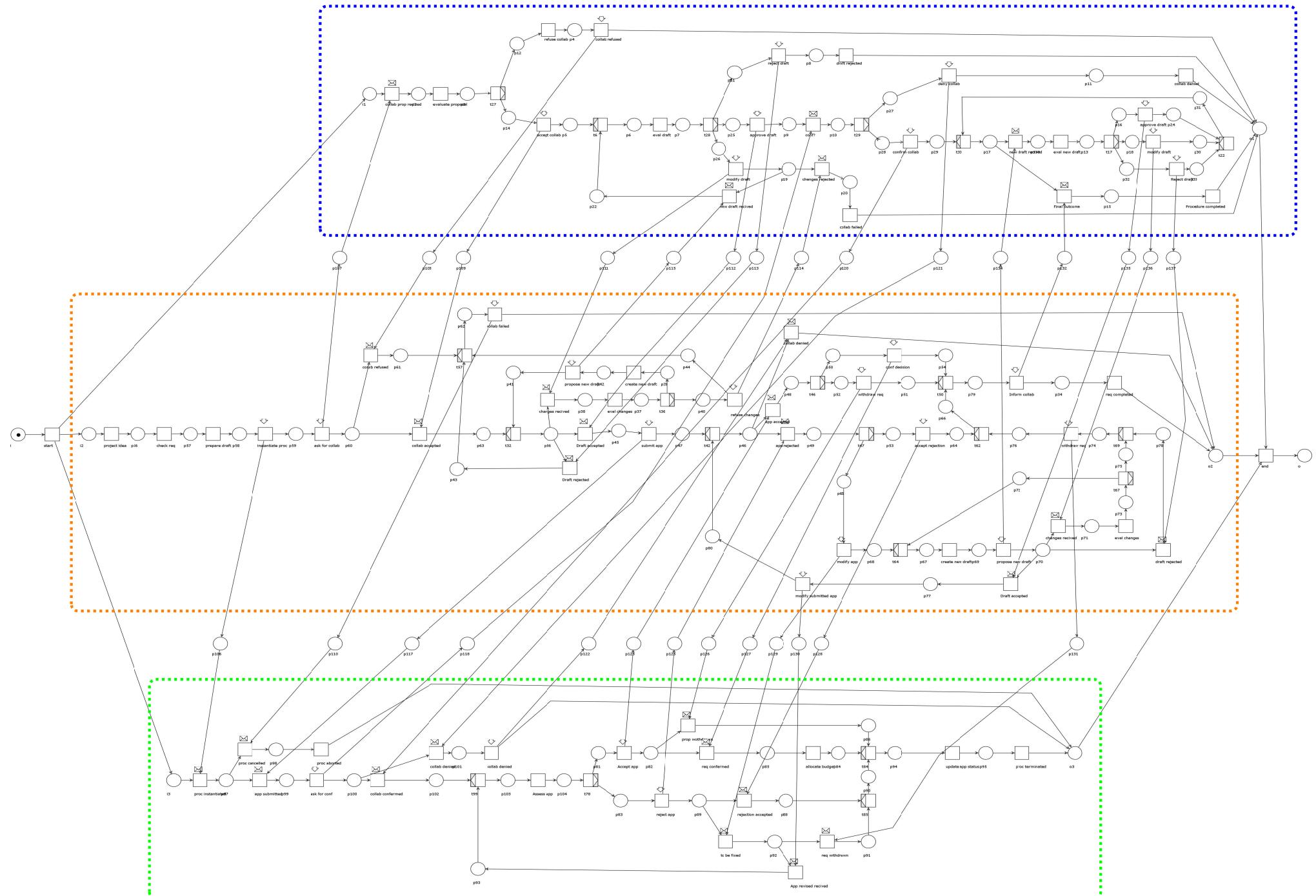


**(b)** Coordinator.



**(c)** Evaluation system.

**Figura 13.** Workflow net dei singoli processi.



**Figura 14.** Workflow system: in blu la workflow net dei collaboratori; in arancione quella del coordinatore; in verde quella del processo di valutazione.