Universitat Politècnica de Catalunya

Generació de descripcions de processos en llenguatge natural

Memòria

Francisco Anselmo López Cuenca

dirigit per
Lluís Padró i Josep Carmona

17 de desembre de 2017

Índex de continguts

Índex de taules i figures

Resum [CAT]

Aquest projecte desenvolupa una solució per generar textos en llenguatge natural a partir de models de processos de negoci escrits en l'estàndard BPMN. L'objectiu és generar una descripció del procés de negoci que sigui escrita de manera similar a les redactades per persones. Per aconseguir-ho, s'utilitzen tècniques del camp de l'anàlisi i la generació lingüística, a més d'estructures de dades i algorismes específics per a que el resultat sigui el més natural per les persones possible.

Abstract [ENG]

This project develops a solution to generate text in natural language from business process models written in BPMN standard. The goal is to generate descriptions of these business processes that are similar to the human-made ones. Linguistic analysis and generation techniques are used in addition to dedicated algorithms and data structures, that help at making more natural texts to the people.

Glossari [CAT]

Procès de negoci -. Conjunt de condicions, tasques i esdeveniments relacionats entre sí en un determinat ordre que produeixen un servei o un producte.

BPMN -. Notació per Models de Processos de Negocis. Estàndard de notació basat en XML i utilitzat per representar processos de negoci tant de manera visual com en forma d'estructura de dades.

NLP -. Processament del Llenguatge Natural. Camp de la computació centrat en l'obtenció, l'anàlisi i la generació de llenguatge humà o similar.

RPST -. Arbre Estructurat del Programa Refinat. Estructura de dades arbrària que representa la organització d'un programa d'ordinador.

DSynT -. Arbre Sintàctic Profund. Estructura de dades arbrària basada en la Teoria del Sentit-Text de Noam Chomsky.

Glossary [ENG]

Business Process -. Set of conditions, tasks and events related on a defined order that produces a product or a service.

BPMN -. Business Process Model Notation. Standard based on XML used to represent business processes visually and as a data structure.

NLP -. Natural Language Processing. Field of computer science dedicated to the study of machine-human interaction and the development of techniques for obtaining, analyzing and generating written or oral human language.

RPST -. Refined Program Structure Tree. Tree data structure that represents the organization of the computer program,.

DSynT -. Deep Syntactic Tree. Tree data structure based on the Meaning-Text Theory of Noam Chomsky.

1. Introducció

1.1. Contextualització

Aquest projecte desenvolupa una solució que genera descripcions en llenguatge natural a partir de models de processos de negoci. Està basat en estudis previs tant del camp dels models de processos com de l'anàlisi i la generació del llenguatge natural.

Els treballs realitzats dins d'aquests camps se centren normalment en l'idioma anglès i les seves propostes sovint obliden que, amb certes modificacions, es poden aplicar també a altres idiomes. Aquest serà un dels objectius d'aquest treball.

El projecte es troba al voltant de dos àrees importants: el Processament del Llenguatge Natural i els Models de Processos.

1.1.1. Processament del Llenguatge Natural

El processament de llenguatge natural és actualment un dels camps amb més importància dins la informàtica. Forma part de ciències de la computació, intel·ligència artificial i lingüística computacional. Estudia la interacció entre persones i màquines, sigui per via oral o escrita. La intenció és crear codis o aplicacions que permetin passar del llenguatge humà a conceptes lògics comprensibles pels ordinadors, i viceversa.[1]

Dintre d'aquest camp trobem tasques que normalment depenen unes de les altres: stemming, lematització, segmentació d'oracions, traducció per màquina, generació de llenguatge natural, OCR[1], reconeixement de parla, etc... La generació de llenguatge natural juntament amb l'etiquetat de categoria gramatical (POS[2]-tagger) i l'anàlisi sintàctic-morfològic seran les eines a utilitzar en el projecte.

Moltes empreses estan interessades en aquest tipus de tecnologia perquè permet automatitzar tasques relacionades amb la comunicació. Per exemple, es poden realitzar informes sobre el temps actual només depenent de la informació dels sensors meteorològics o generar resums sobre textos llargs.

El procès comença quan l'usuari rep l'informe de crèdit. Es realitza l'aprovació. Llavors, una de les següents opcions s'executa:

- Si és aprovat, no es fa res.
- Si no, l'usuari inclou l'historial de transaccions.

Un cop s'ha executat una de les branques, l'usuari inclou el text estàndard. Finalment, el procés s'acaba.

Text 1: Descripció del model de la Figura 1

En el cas del projecte actual, s'utilitzarà la llibreria FreeLing[2], que realitza tasques d'anàlisi sintàctic, anàlisi morfològic i etiquetat POS. Per la part final del procés, la realització, s'utilitzaran les llibreries RealPro[3], de CoGenTex, i SimpleNLG[4].

1 *Optical Character Recognition*, Reconeixement Òptic de Caràcters

2 *Part of Speech*

1.1.2. Models de processos de negoci

Els processos de negoci són conjunts de tasques i condicions relacionats lògicament que produeixen un servei o un producte. La gestió d'aquests processos és un camp que s'estudia per tal de trobar millorar les estructures d'organització i producció d'una empresa o institució.

Els models de processos de negoci són les representacions visuals i de dades d'aquests processos. L'estàndard de facto dels models és BPMN (Business Process Model and Notation). És un format per la descripció i visualització de processos de negoci basat en UML, mantingut actualment pel Object Management Group. Els arxius BPMN tenen com a representació textual el format XML. Els principals usuaris d'aquesta especificació són els analistes de negoci, gerents i desenvolupadors.

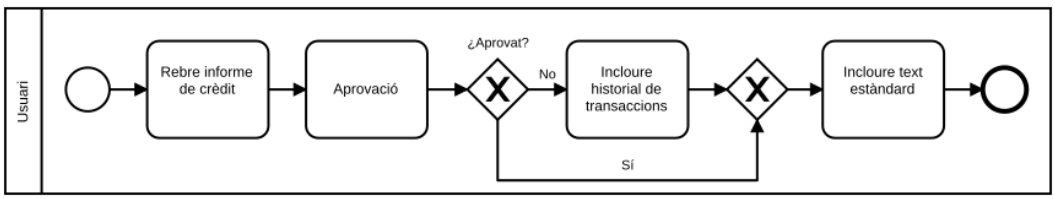


Figura 1: Visualització d'un model BPMN

Els elements més importants que conformen l'estàndard són quatre: els elements de flux, els elements de connexió, els carrils de natació i els artefactes. A la categoria d'elements de flux trobem les activitats, els esdeveniments i les portes.

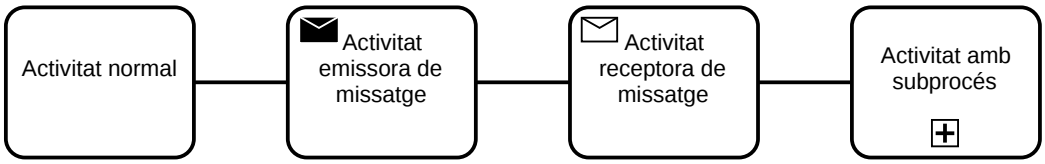


Figura 2: Tipus d'activitats en un diagrama BPMN

Les activitats representen les tasques a realitzar dins un procés. Es simbolitzen amb rectangles amb cantonades arrodonides. A part de les activitats normals, podem trobar-ne d'altres tipus. Per exemple, les activitats amb subprocés, que contenen, com el seu nom indica, un procés que pot estar o no especificat també al mateix model.

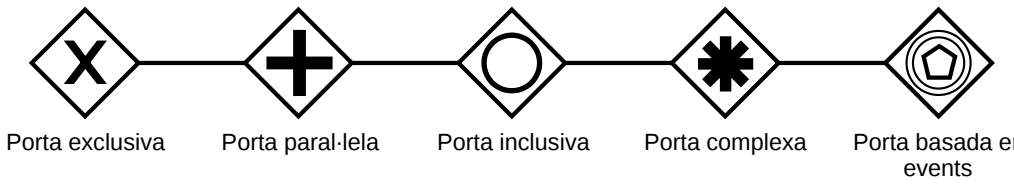


Figura 3: Tipus d'activitats en un diagrama BPMN

Les portes representen la separació i/o fusió de camins depenent de condicions o missatges, Es simbolitzen amb rombes i el símbol interior ens indica el tipus concret. Els tipus de portes BPMN són els següents:

- **Portes XOR o exclusives:** En aquest tipus de portes, només es pot escollir una de les branques sortints.

- **Portes AND o paral·leles:** Expressen l'inici i final de branques que s'executen al mateix temps.

- **Portes OR o inclusives:** Indiquen que una branca o més d'una poden ser executades, depenent de la condició i cadascuna de les seves opcions.

- **Portes complexes:** Representen condicions que no poden ser expressades amb les portes anteriors. Es recomana provar altres alteratives com combinacions d'altres portes abans d'utilitzar aquesta. És necessari indicar com s'ha d'escollir la o les branques a la seva etiqueta.

- **Portes basades en esdeveniments:** Aquestes portes es comporten com una porta exclusiva però en comptes de tenir una condició textual, requereixen d'un esdeveniment per decidir el camí. Es pot dir que quan s'arriba a una d'aquestes portes, esperem fins que l'esdeveniment succeeixi.

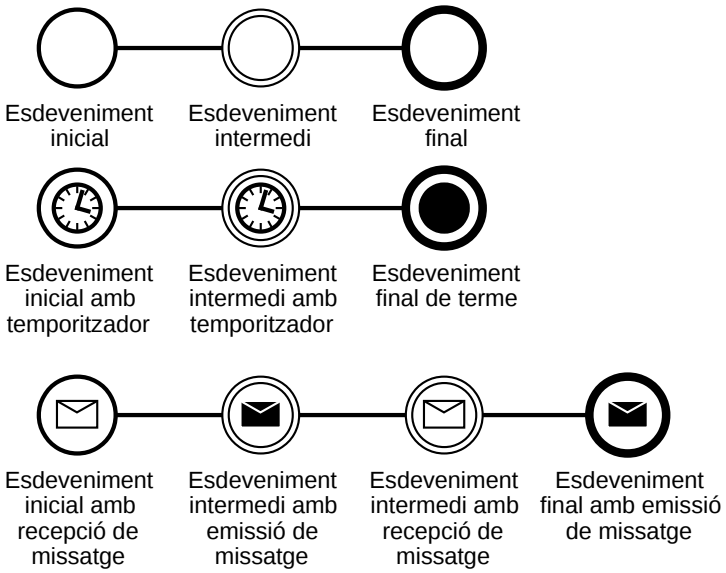


Figura 4: Tipus d'esdeveniments BPMN

Els esdeveniments són allò que ocorre durant el procés. Es simbolitzen amb cercles, on la seva vora indica en quin moment del procés ocorre i el símbol, el tipus. Els moments on els esdeveniments poden trobar-se són a l'inici (circumferència simple), en mig del procés (circumferència doble) i al final (circumferència gruixuda). Si contenen un símbol, aquest representa l'acció que esperem que succeeixi. A la Figura 4 podem veure alguns dels tipus d'esdeveniments que trobem a l'especificació de BPMN.

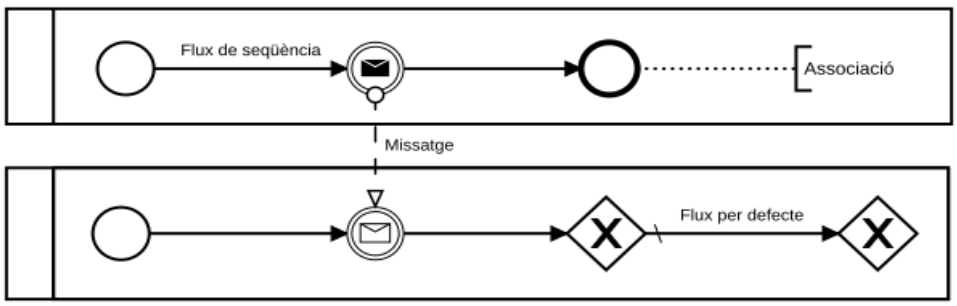


Figura 5: Tipus d'elements de connexió BPMN

Dins els elements de connexió trobem els fluxos de connexió, els missatges i les associacions. Els fluxos de connexió simplement connecten diferents elements de flux seguint la seqüència del procés. Els missatges són informació que es passa entre diferents punts del procés. Les associacions uneixen artefactes o text als elements BPMN.

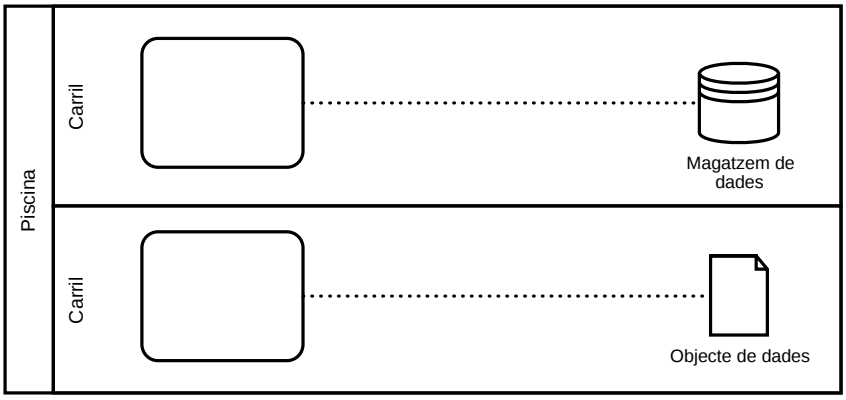


Figura 6: Piscina, carrils i artefactes BPMN

Els carrils de natació són la forma de representar què o qui realitza les accions d'ells i es poden agrupar en piscines. Per últim, els artefactes són informació addicional que s'afegeix al diagrama com anotacions, grups d'activitats o objectes. Gràcies a aquests elements, es pot dissenyar qualsevol tipus de procés de negoci.

1.2. Actors implicats

En aquesta secció es mostren les persones implicades en el projecte:

1.2.1. Desenvolupador

El rol de desenvolupador l'assumeixo jo mateix, l'Anselmo López Cuenca. A banda d'analitzar les taques, dissenyar les solucions, desenvolupar les implementacions i incorporar-les en el codi, entre les meves tasques s'inclouen també la gestió d'aquest projecte, l'escriptura de la memòria i la realització de les proves adients.

1.2.2. Directors del projecte

Els directors del projecte són Josep Carmona i Lluís Padró, professors del Departament de CS (Computer Science) de la Universitat Politècnica de Catalunya. El seu paper és el de supervisar el projecte i donar informació complementària sobre el camp en el que es treballa, en cas que sigui necessari.

1.2.3. Actors beneficiats

Les grans empreses i institucions realitzen milers de tasques al dia. Tenir un control de l'eficiència d'aquestes tasques és vital per obtenir bons resultats. Aquestes tasques, l'ordre entre elles i les seves condicions formen els coneguts com a processos de negoci.

Els usuaris de la solució podran obtenir com a resultat textos on es descriuen les accions que el procés realitza. D'aquesta manera podran entendre millor el funcionament dels processos que es duen a terme i es podran detectar millors possibles punts febles.

Una altra aplicació, que es comentarà en profunditat més endavant, és la generació de 'converses' a partir de models en format BPMN. Aquests diàlegs mostren pas a pas el procés i les seves tasques. A més, es poden construir respostes a preguntes que pugui fer l'usuari sobre alguna tasca del procés.

2. Problema i objectius

2.1. Formulació del problema

Les empreses realitzen cada dia milers de tasques a la mateixa vegada. Aquestes tasques normalment tenen un ordre i vàries condicions d'execució. Aquestes agrupacions s'anomenen processos, que poden ser estudiats, avaluats i/o actualitzats segons convingui. Per tal de poder definir-los, existeix el format estàndard BPMN[3]. Gràcies a aquesta notació, tenim una descripció visual de les tasques amb una notació idèntica a qualsevol empresa. BPMN permet als experts revisar els processos i millorar-los independentment de l'empresa i de les activitats que aquesta pugui dur a terme. A partir d'aquests estudis, es detecten aquelles tasques o esdeveniments que alenten el procès sencer i es poden eliminar o tractar de manera adequada.

De la mateixa manera que un format tècnic i estàndard és útil pels entesos i els experts, les persones que prenen les decisions o les han de dur a terme no tenen perquè saber com interpretar aquests diagrames. Per aquesta raó, és de gran utilitat traduir l'esquema BPMN a llenguatge natural que pugui ser llegit. Així, tant els caps com els experts en BPMN podran discutir sobre eficiència, eficàcia o altres característiques dels processos en qüestió.

El text resultant ha de permetre als lectors entendre millor el funcionament de les tasques i processos descrits. La generació d'aquest text ha de contemplar la planificació i organització lògica de les paraules per ser intel·ligible i comprensible pel lector. Per tant, la voluntat d'aquest projecte és resoldre aquesta problemàtica de manera eficient, adaptable a idiomes diferents de l'anglès i comprensible sense gaire esforç.

2.2. Objectius del projecte

Un cop detectat el problema a resoldre pel projecte, podem definir els objectius i requeriments que s'han de complir i assolir.

3 *Business Process Model Notation*, Notació per Models de Processos de Negoci

En general, l'objectiu de la transformació d'un diagrama BPMN a llenguatge natural és la validació del procés per part de les persones amb poder de decisió. El llenguatge natural en el nostre projecte serà el text resultant del programa, que ha d'estar redactat com si l'esquema fos descrit per una persona real.

El projecte engloba diversos objectius concrets a ser assolits per tal de produir el resultat esperat:

- **Investigar i fer recerca:** Llegir articles i informes del camp de la generació i l'anàlisi de llenguatge natural per tal de trobar els millors mètodes que es puguin entregar al projecte.

- **Desenvolupar un programa eficient i organitzat:** Implementar un codi que resolgui el problema formulat abans i ho faci de manera eficient amb la possibilitat de ser posteriorment modificat sense complicacions.

- **Facilitar l'adaptació a altres idiomes:** Permetre que altres desenvolupadors puguin adaptar el programa per BPMN amb etiquetes d'altres idiomes.

- **Cercar exemples de diferent dificultat:** Per provar correctament el programa durant i després de la implementació, s'han de buscar exemples que incorporin diferents elements de l'estàndard BPMN.

- **Facilitar interfícies entre persona i màquina:** Crear o modificar eines per construir interfícies com converses per chatbots.

Els punts descrits han sigut escollits perquè el programa pugui tant resoldre la problemàtica descrita prèviament com per ser una base per futures modificacions i treballs.

3. Estat de l'art

El camps tant dels processos de negoci com del processament de llenguatge natural tenen moltes branques de recerca. En el cas concret d'aquest projecte, volem utilitzar la generació de llenguatge natural per tal d'aconseguir transformar els diagrames BPMN en text.

Existeixen estudis en aquest camp que ens serviran de fonts de coneixement. Aquests tracten tant sobre generació de llenguatge sobre models com d'anàlisi o algorísmia. A continuació, resumeixo alguns d'ells:

3.1. Models BPMN a text

La transformació d'un diagrama BPMN a text ja s'ha estudiat prèviament, com es mostra al document «Supporting Model Process Validation through Natural Language Generation»[5]. Aquest treball s'utilitzarà com a base del projecte.

En el document s'explica un mètode per la generació d'aquest textos seguint els següents passos:

- **Extracció de la informació lingüística:** Es recorren tots els elements del BPMN i s'extreu la informació lingüística de les seves etiquetes gràcies als anàlisis sintàctics, morfològics, ontològics i a l'etiquetat POS. Aquestes dades s'utilitzaran posteriorment per afavorir la construcció d'oracions ben formades.[6]

- **Transformació del diagrama en RPST[4]:** Construïm l'arbre RPST segons l'algorisme descrit o una de les seves alternatives per tal d'obtenir l'estructura del procés i l'ordre dels nodes.[7], [8]

4 Refined Program Structure Tree

- **Planificació del text i generació d'estructures DSynS[5]:** Aquí és on s'ordena el text segons l'ordre del RPST i segons el flux del procés. A més, es generen les estructures sintàctiques a partir del RPST i de les dades lingüístiques que s'han extret abans. Decisions com posar més paràgrafs o fer llistes de punts, o indentar més o menys les parts es prenen en aquest punt.

- **Agrupació d'oracions i frases:** Aquesta part elimina duplicitats en oracions consecutives per tal que les frases s'assemblin més a les que diria una persona. Revisa, per exemple, si dues oracions tenen el mateix subjecte o realitzen la mateixa acció.

- **Realització del text:** S'executa l'algoritme de realització que agafa les DSynS i els converteix en text llegible.[9], [10]

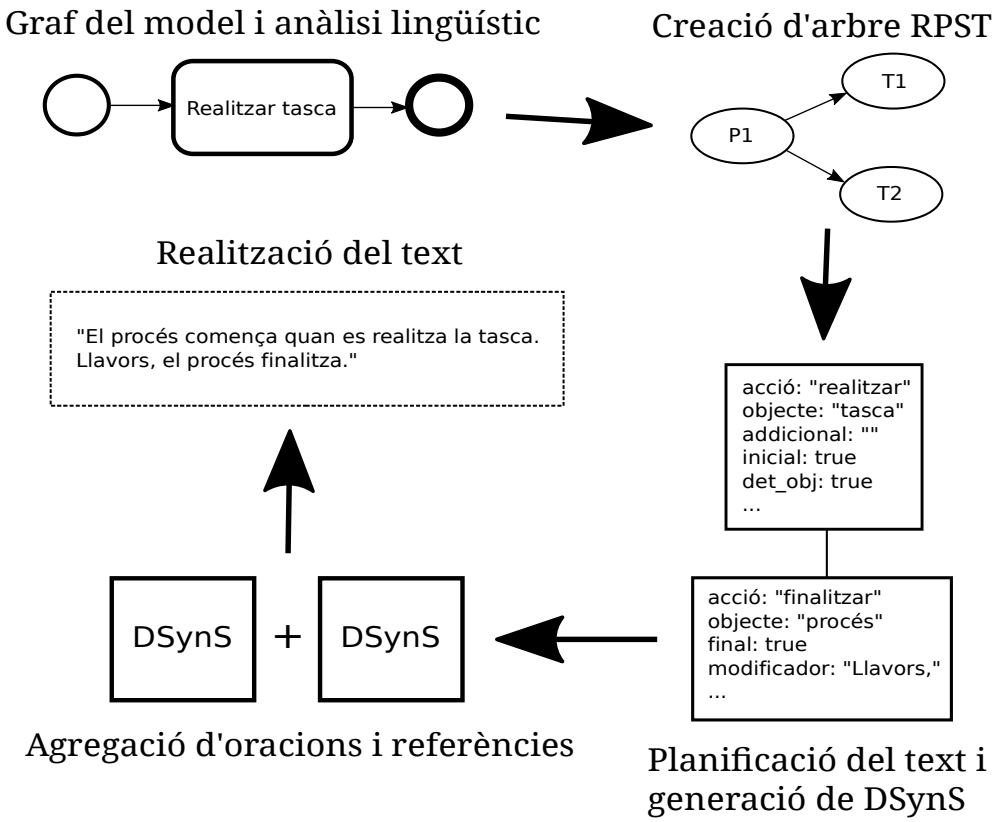


Figura 7: Procés de generació de descripcions de models de processos

5 Deep Syntactic Structures, Estructures de Sintaxi Profunda

El algoritmes i certes parts del treball estan fets per funcionar només en llengua anglesa, és a dir, s'aprofita de característiques pròpies de la llengua i les utilitza per simplificar el mètode explicat.

3.2. Estructurant models de processos

Els models de processos es representen normalment com a grafs dirigits que segueixen uns requisits, com que per cada porta d'obertura de qualsevol tipus li correspon una porta de tancament del mateix tipus. Normalment, per desconeixement o per altres motius, aquestes regles s'incompleixen i poden donar lloc a regions del graf que s'anomenen rígids.

El treball «Automated Discovery of Structured Process Models: Discover Structured vs. Discover and Structure»[11] [7] mostra com reestructurar els rígids un cop ja s'ha construït l'arbre RPST. Aquests canvis ens permetran generar textos que segueixin un ordre lògic pel lector sense generar llistes amb desviacions per cada branca o camí possible dins el rígid.

Per aconseguir-ho se segueixen els següents passos, resumits:

- **Generació del RPST:** El primer objectiu és generar l'arbre RPST amb l'algorisme que desitgem. A l'article s'utilitza el «Heuristics Miner».

- **Revisió dels rígids:** Un cop tenim generat l'arbre, hem de cercar totes aquelles regions etiquetades com a rígid. Per cada un d'ells, s'ha d'avaluar quins tipus de portes el formen i si està mínimament estructurat, Depenent de les condicions, s'utilitzarà BPStruct o Extended Oulsnam.

- **Aplicació dels mètodes**

  - **BPStruct:** Aquest mètode s'aplica quan el rígid és solid i, o bé està format només per portes AND o bé conté diversos tipus de portes. Consisteix en generar una xarxa de Petri a partir de la part del graf corresponent al rígid. A partir d'aquesta xarxa es dupliquen les parts adequades amb el «Proper Complete Prefix Unfolding». Es crea un graf de relacions d'ordre entre nodes i, si no ha aparegut cap error, el model del procés estarà ben estructurat.

- **Extended Oulsnam:** Aquest mètode s'aplica en els casos restants. Consisteix en detectar els punts anomenats injeccions i ejeccions, i aplicar els operadors push-down i pull-*up*, respectivament. Les injeccions i les ejeccions es mostren a la figura 2, on els rombes representen les portes de qualsevol tipus i els quadrats, una seqüència ininterrompuda de tasques. Els operadors s'explicaran més endavant, ja que seran utilitzats a la implementació del programa.

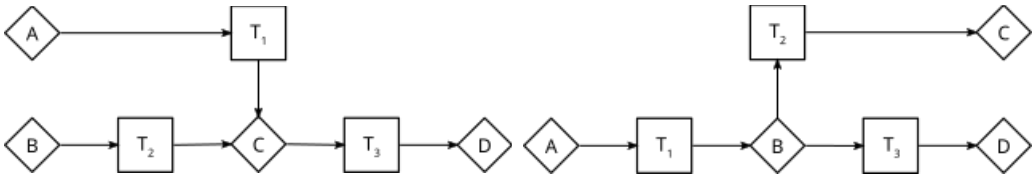


Figura 8: Exemples: a) injecció i b) ejecció

- **Nova generació del RPST amb els canvis realitzats:** En el moment en que s'han aplicat els operadors corresponent, s'ha de recalcular la estructura d'arbre del RPST per tal de ser coherents amb els canvis.

Gràcies als resultats d'aquest treball, la generació de text aplicada als rígids millora ja que aquests acaben transformat en un conjunt de condicions i portes ben estructurades.

4. Definició de l'abast

4.1. Abast

Per començar, s'ha d'entendre el format BPMN. És necessari comprendre la seva visualització, la representació interna i com es pot traspassar tota aquesta informació al programa. Un arxiu BPMN normalment consta de dues parts: les definicions dels elements i la descripció de la visualització. El programa ha de llegir la part de les definicions tenint en compte que els elements poden no ser definits dins una piscina o un carril.

Després, analitzem les etiquetes dels elements utilitzant FreeLing, una llibreria que es dedica a l'anàlisi del text i permetran obtenir informació sobre ell, per exemple, si és verb o nom, si és un plural o a quina categoria pertany (humà, inanimat, ...).

El següent pas és la creació de diferents arbres i estructures de dades per tal d'organitzar el text i crear frases de manera efectiva. El RPST i els DSynS són les escollides per representar l'estructura del procés i la sintaxis requerida, respectivament.

A continuació, es realitza l'agregació de frases, la generació de referències i, per últim, la realització. Es posaran paràmetres per a que els textos puguin tenir més o menys llistes, major o menor profunditat als nivells de referència, etc...

Per acabar, un altre aspecte que es tindrà en compte serà la possibilitat d'ampliació per altres idiomes. Per aconseguir-ho es revisarà l'estructura del codi i es posaran en pràctica patrons de disseny de software, a més de incorporar noves eines, si escau.

4.2. Possibles obstacles

Els estudis fets estan centrats exclusivament en diagrames BPMN en llengua anglesa. Per tant, la major part del codi tracta les regles i excepcions lingüístiques d'aquest idioma. Pot portar temps reorganitzar el codi actual, fet per l'anglès, per a que sigui senzill de reutilitzar en altres idiomes.

S'utilitza software fet per altres persones, per tant, poden sorgir problemes que no es podrien solucionar al primer moment i que necessiten d'alternatives. La seva cerca pot comportar temps perdut i que es podria utilitzar en altres parts del projecte.

Malgrat que el format BPMN és un estàndard, es pot ampliar mitjançant extensions i algunes d'elles només tenen sentit en software de tercers. Per tant, poden haver elements que no s'hagin tingut en compte i això podria interferir en l'execució i/o en el resultat.

Els errors en el codi suposen molt temps a dedicar. Poden haver errors que surten en condicions molt específiques que, malgrat disposar del debugador, són difícils de trobar. A més, s'han de pensar de solucions que no canviïn en excés el codi desenvolupat.

El temps i una mala planificació poden convertir-se en un gran problema. Si les fites no es compleixen, el projecte es pot endarrerir causant la eliminació d'objectius principals.

5. Metodologia del projecte

La metodologia del projecte tindrà components de les metodologies AGILE, en especial del SCRUM. Aquestes tècniques permetran desenvolupar més ràpidament durant el curt període de temps que dura el TFG. En el cas del projecte actual, s'apliquen tant les reunions setmanals i els esprints.

5.1. Seguiment i validació

Per avaluar l'avenç del projecte i el seu estat, cada setmana hi haurà una reunió amb els directors de projecte per analitzar el progrés, planejar els pròxims passos i debatre solucions als obstacles que es puguin presentar.

En cas d'estancament o d'alguna problemàtica, es valorarien diverses alternatives o solucions debatent a les mateixes reunions setmanals. Les propostes poden ser des d'una reestructuració de les taques prèviament definides fins a una cerca de propostes per solucionar un problema concret.

Caldrà executar proves al final de cada iteració per tal de comprovar si s'ha de reescriure el codi o es pot passar a la següent part. Les proves han de ser exhaustives i han de contenir la major part de la especificació BPMN. D'aquesta manera, ens assegurem de la eficàcia del programa.

Per validar els resultats finals, utilitzarem mètodes similars als exposats en treballs anteriors[5] com comparar textos prèviament escrits per persones amb els resultats de l'execució del programa. Aquests textos i els seus respectius diagrames provenen de col·leccions de processos de negoci pertanyents a universitats o concursos.

No s'han produït variacions en quant a la metodologia establerta en el projecte des de la planificació inicial.

6. Planificació temporal

6.1. Programa

El temps estimat per la duració del projecte és de 704 hores, comprèn un període aproximat de 4 mesos, des del 12 de setembre fins al 12 de gener. Si s'inclou el treball previ, revisant els articles de referència i codi, el temps es pot incrementar en 1 mes extra.

Aquesta estimació pot variar depenent de les circumstàncies i de les decisions preses a les reunions setmanals amb els directors. En aquests cassos, es debatran les eventualitats i les possibles alternatives.

6.2. Pla de projecte

El projecte s'ha dividit en diferents etapes depenent de la interconnexió entre les tasques de cadascuna i la seva dificultat. Són les següents:

- Fita inicial
- Anàlisi, disseny i preparatius del projecte
- BPMN i anàlisi lingüístic
- Arbre RPST
- Planificació i estructures DSynS
- Agregació d'oracions i refinació
- Realització de text
- Fita final (Redacció de memòria, annex i documentació)

6.2.1. Fita inicial

La primera fase del projecte és la fita inicial i és en la que ens trobem actualment i consta de aquestes parts:

- Definició de l'abast
- Planificació temporal
- Gestió econòmica i sostenibilitat
- Estat de l'art
- Contextualització i bibliografia

S'haurà de lliurar la documentació amb aquestes parts respectant les rúbriques, juntament amb una presentació preliminar, que servirà d'entrenament per la presentació final.

6.2.2. Anàlisi i definició del projecte

La següent part és l'anàlisi i definició del projecte. Durant aquestes setmanes, s'hauran de decidir objectius, requisits i funcionalitats a complir. A més, també s'analitzaran les competències i les tecnologies per tal de descriure l'estat de l'art en la matèria.

També, s'ha de tenir en compte el disseny. El programa serà inclòs a la plataforma «nlp4bpm», on es troben diferents eines relacionades amb els models de negoci, i un dels objectius a complir és la possibilitat d'adaptació a altres idiomes. Per tant, és necessari pensar en l'arquitectura de software i programar tenint tot això en compte.

6.2.3. BPMN i anàlisi lingüístic

En aquesta etapa, la prioritat estarà en llegir correctament els models BPMN, traduir-los en estructures de dades i analitzar les etiquetes dels nodes del model. La notació BPMN està basada en XML, el qual és un format molt estès, i resulta fàcil trobar parsers d'aquest llenguatge de marques.

Un cop obtenim les estructures de dades necessàries, utilitzarem la llibreria FreeLing per realitzar els anàlisis pertinents. La seva missió és trobar l'acció i l'objecte del verb, a més d'altres característiques com conjuncions o partícules. La informació resultant de l'anàlisi per cada node serà afegit a cadascun d'ells amb l'objectiu d'utilitzar-la durant la creació dels DsynT.

6.2.4. Arbre RPST

La creació correcta de l'arbre RPST serà un element clau en el projecte. Cada node de l'arbre representa un conjunt d'arcs amb els seus nodes del model. Aquests conjunts són: trivials, bonds, polígons i rígids.

Els rígids són els conjunts que no s'han pogut agrupar en les altres categories i, per tant, són més complicats d'abstreure per crear sentències. Per sort, hi ha estudis que donen solucions per reestructurar els rígids i obtenir nodes de tipus més simple. S'hauran d'implementar els algorismes proposats i adaptar-los al projecte per tal que siguin el més eficient possible.

6.2.5. Planificació i estructures DSynS

La següent etapa és la planificació del text i la creació dels missatges d'arbre sintàctic profund. L'arbre RPST resultant de la etapa anterior ens dona una estructura arbrària amb nodes ordenats que ens permetrà crear sentències i oracions en un ordre lògic.

Ara bé, l'ordre lògic del model no implica un ordre que sigui lògic pel lector. El planificador organitzarà i decidirà en quin ordre s'aniran creant les estructures DSynS a partir de factors com la profunditat del node en l'arbre o la quantitat de tasques.

Les DSynS són estructures de dades en forma d'arbre basat en la teoria Sentit-Text de Noam Chomsky, on es diferencia la forma final de les frases (Surface, SSynS[6]) del sentit d'elles (Deep, DSynS[7]). Els DSynS formen part de les estructures que representen el significat abstracte de les sentències.

6 Surface Syntactic Structures
7 Deep Syntactic Structures

6.2.6. Agregació d'oracions i refinació

Aquí es repassaran els missatges creats. Es revisarà si, per exemple dos missatges consecutius tenen el mateix subjecte per tal de unir-les o traslladar una oració llarga a una altre espai i referenciar-la en la frase original.

6.2.7. Realització del text

L'altra etapa clau del projecte és la realització del text. Existeixen diverses maneres de generar text i cadascuna utilitzar diferents dades per fer-ho. Algunes necessiten de l'estadística o d'un gran corpus de paraules. En aquest projecte ens decantarem per aquells que utilitzin les DSynS, com RealPro o Support Vector Machines, o que permetin la adaptació per altres idiomes, com SimpleNLG.

6.2.8. Fita final

Per a la fita final, s'afegirà el codi resultant del projecte, documentació útil i altra informació relacionada. La inclusió del projecte a la plana web comportarà temps per tal d'acomodar-ho correctament. Un cop complert, es farà una revisió general del treball i serà el torn de crear i pensar la presentació pel tribunal de TFG.

6.3. Duració estimada

Tasca	Hores
Fita inicial	240h
Anàlisi, disseny i preparatius del projecte	80h
BPMN i Anàlisi lingüístic	40h
Arbre RPST	72h
Planificació i generació d'estructure DSynS	56h
Agregació i refinació de missatges	48h
Realització de text	72h
Fita final	104h
TOTAL	704h

Taula 1: Duració estimada de cada fase

6.4. Recursos

En aquest apartat, es descriuran els recursos necessaris per la realització del pla del projecte.

6.4.1. Recursos humans

Per poder desenvolupar i portar el projecte, fa falta persones amb rols específics. En aquest cas, els rols necessaris seran: el cap de projecte, el desenvolupador i el beta-tester.

6.4.2. Hardware

En quant al hardware, es requereix un ordinador per el desenvolupament del codi, concretament s'utilitza un MSI GP60 2PE, i un servidor on penjar el codi per tal d'executar-lo quan es necessiti a través del web. El servidor pel projecte serà un dels pertanyents al departament de Computer Science de la UPC.

6.4.3. Software

Pel que fa al software, s'utilitzarà el següent: el sistema operatiu Linux Mint 18.04, l'entorn de desenvolupament Eclipse en la versió per Java EE, les llibreries FreeLing, RealPro i SimpleNLG, i els programes ProjectLibre, TexMaker i LibreOffice.

6.5. Valoració d'alternatives

Al anar avançant el projecte, poden succeir diverses dificultats o eventualitats. Totes aquestes desviacions sobre el pla seran discutides a les reunions setmanals amb els directors. Així es podran analitzar i debatre diferents propostes sense perdre temps de dedicació. A les reunions, aquestes propostes són avaluades en funció dels possibles costos que generin i del temps estimat de realització. Sempre s'escull la opció que comporti el mínim nombre de canvis possibles a la planificació.

Les dates, i sobretot les hores, són orientatives i poden variar del que es disposa al diagrama de Gantt. Per altra banda, algunes tasques que s'ha planejat fer per separat es podrien combinar i fer en paral·lel sense complicacions pel projecte.

Per tal de que els imprevistos no afectin a l'estimació econòmica del projecte, una de les partides del pressupost estarà destinada a contingències i imprevistos. Aquest cost serà d'un 15% del pressupost estimat sense aquesta partida.

6.6. Estat del projecte respecte la planificació

En el moment actual, el projecte es troba finalitzat, un cop acabada la fase de documentació i revisió. Les tasques que estaven previstes s'han anat resolent satisfactòriament.

Pel que fa a la configuració de l'entorn i preparatius inicials, es van realitzar sense problemes segons la planificació feta.

La fase d'anàlisi d'etiquetes del model funciona correctament utilitzant FreeLing com a llibreria base. S'utilitza un analitzador d'etiquetes ràpid creat pel grup NLP4BPM de la UPC.

La creació del graf del model i del seu arbre RPST estan en una fase molt avançada. La construcció del arbre RPST original presentava algunes deficiències i s'està programant una versió millorada que en refina el resultat. S'utilitzen tècniques trobades durant la investigació inicial i alguns articles proporcionats pels directors.

La creació de les DSynS i la refinació d'aquest missatges es troben acabats per llengua anglesa. L'adaptació per altres llengües dependrà de la disponibilitat de l'analitzador d'etiquetes per altres idiomes. Ara bé, aquesta adaptació només requereix substituir les cadenes de text predefinides en anglès per les dels altres idiomes.

La realització del text es genera gràcies als realitzadors RealPro i SimpleNLG per llengua anglesa. Degut a que RealPro només genera textos en anglès i que el seu codi no és obert, s'està desenvolupant una versió del SimpleNLG per llengua catalana basat en el projecte SimpleNLG-ES, de la Universitat de Santiago de Compostel·la. D'aquesta manera, es pot integrar un nou idioma al sistema i modificar parts que ho necessitin amb total llibertat.

7. Implementació del projecte

En aquest apartat, s'explica el desenvolupament i la implementació del projecte incloent també comentaris sobre cada aspecte.

7.1. Lectura i creació dels models

El primer pas és seleccionar el fitxer amb les dades dels models en format BPMN. L'arxiu ha de complir amb l'estàndard BPMN per tal que el convertidor el pugui analitzar bé. El convertidor utilitzat pertany a la llibreria Activiti.

Malgrat tot, és necessari implementar un filtre previ. El convertidor falla directament en el moment en que troba marques que pertanyen a extensions de l'estàndard que desconeix. Per tant, el filtre ha de ser capaç de guardar la informació a netejar i eliminar-ho per convertidor. La informació guardada pot ser necessària en fases posteriors del procés.

Un cop obtenim les dades i les organitzem dins de les estructures auxiliars, hem de crear el model en forma de graf per tal de poder saber quins element precedeixen o segueixen a un altre element dins del model. Utilitzem una classes convertidora per llegir tots els elements del model i creem els objectes corresponents però sense tenir una subclasse per cada subtipus d'element, és a dir, no hi haurà una classe per les portes exclusives i una altra per les paral·leles sinó que només n'hi haurà una amb un atribut que indicarà el tipus.

En aquest punt podem detectar si el model té dos o més terminals (nodes d'entrada i/o nodes de sortida). En cas de tenir més de d'un terminal d'entrada o sortida, hem de convertir el graf per tal que en tingui només dos. La millor forma d'aconseguir-ho és creant a l'inici i/o al final una porta XOR i un esdeveniment corresponent, i substituir els terminals múltiples per la porta comuna que els pertoqui.

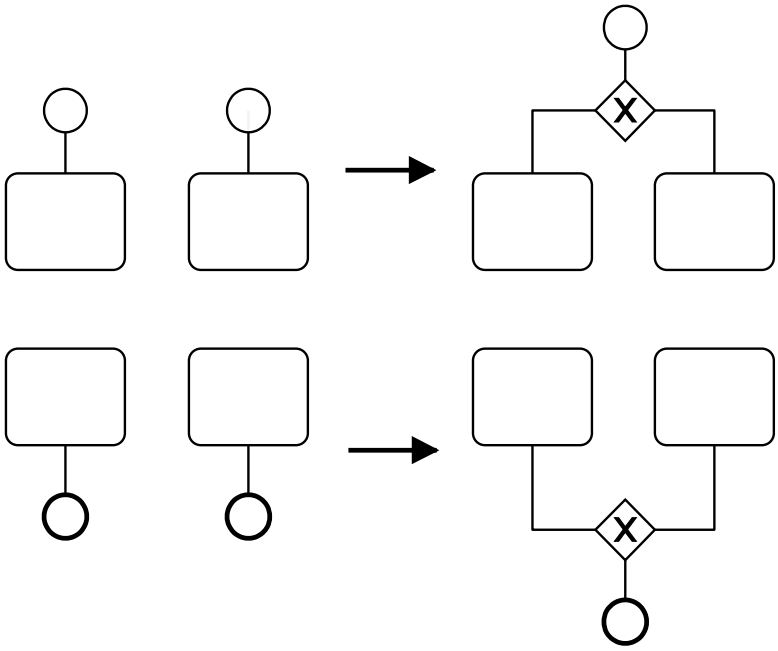


Figura 9: Exemple de transformació de múltiples terminals inicials i/o finals en un únic terminal

Per altra banda, el model del procés pot estar conformat per més d'una piscina i, per tant, hi haurà més d'un node inicial. Per als següents passos caldrà separar les piscines i tractar-les com a models diferents.

7.2. Anotació del model

Obtenir correctament la informació lingüística de les etiquetes del model és vital si volem que el text resultat tingui sentit i sigui comprensible. Podem definir que un text és comprensible si segueix les regles morfològiques, sintàctiques, gramaticals i ortogràfiques, entre d'altres, d'un idioma.

Aquestes dades s'obtenen aplicant les tècniques del processament del llenguatge natural com el POS-tagger o etiquetador de categoria gramatical. En el projecte, aquest anàlisi es fa gràcies a l'analitzador d'etiquetes de l'equip NLP4BPM de la Universitat Politècnica de Catalunya, que realitza aquesta tasca amb gran velocitat. L'analitzador ens permet obtenir la categoria gramatical d'un mot, juntament amb altres dades com poden ser el gènere, el nombre, la persona, etc.

La necessitat d'un analitzador especial per les etiquetes prové de comprovar que les eines típiques esperen analitzar una frase o un text sencer. Les etiquetes, en canvi, segueixen patrons diferents. L'analitzador utilitza les eines disponibles de FreeLing per extreure la informació inicial d'aquestes sentències. Normalment, s'obtindria només una categoria per cada paraula, però per l'analitzador s'obtenen les tres millors.

El següent pas és traslladar tota aquesta informació al parser basat en regles. Tenint en compte la categoria gramatical de cada paraula, el parser ens retornarà tots els sintagmes que formen part de l'arbre trobat. Si no, es basarà en les categories trobades prèviament per donar un resultat. Aquest resultat seran les paraules amb la informació de l'anàlisi i els índexs per cada part de l'etiqueta (acció, objecte i addició).

L'analitzador necessita una llista de les etiquetes i per tant, recorrem tots els elements del graf agafant les teves etiquetes. Després, s'estudien en conjunt totes les etiquetes retornant una llista d'estructures on podem trobar la categoria, el lema i altres característiques de la paraula. El context que ens ofereix el conjunt de les etiquetes ajudarà a obtenir millors resultats.

Amb totes aquestes dades, el que volem és detectar quin és l'objecte i quina l'acció de l'element. Juntament amb el subjecte, que el podem trobar com a nom del carril o de la piscina, i altres dades addicionals, formem les anotacions que ens serviran posteriorment a la fase de planificació del text.

Conèixer la funció de cada paraula permetrà al planificador construir correctament les dades sintàctiques per cada node i, al realitzador, generar les frases respectant les regles lingüístiques que estiguin implementades. Això augmentarà la comprensió del text als lectors de la descripció del model.

7.3. Creació de l'arbre RPST

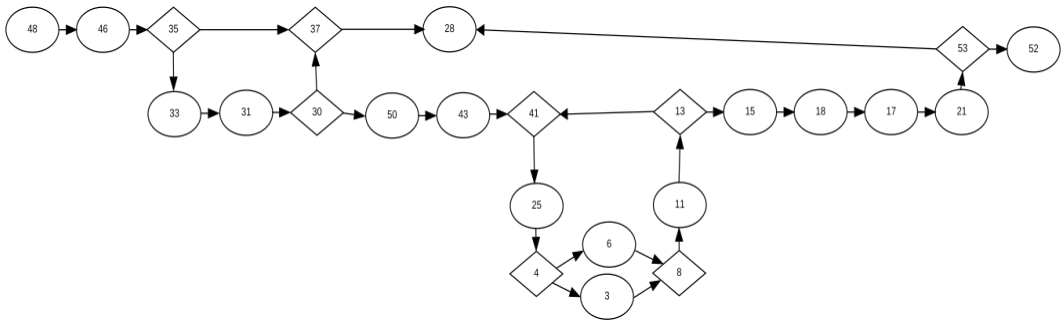


Figura 10: Graf del model abans de reestructurar

La següent tasca a realitzar és la generació de l'arbre RPST del model. Per tal efecte, s'utilitza l'algorisme de Artem Polyvyanyy et al., explicat a «Simplified Computation and Generalization of the Refined Process Structure Tree»[7], que està incorporat a la llibreria JBPT.

A mode de resum, els passos que segueix l'algorisme són els següents:

- Normalitzar el graf: es considera que el graf d'un model de processos amb únicament dos terminals, anomenats TTG[8], està normalitzat quan tot node té com a màxim un eix entrant o com a màxim un eix sortint.

- Generació del TCT[9] a partir del graf normalitzat.

- Assignació dels tipus: depenent de les característiques que es compleixen, a cada fragment se li assigna el seu tipus corresponent.

- Eliminació de les redundàncies i els fragments trivials que només existeixen al graf normalitzat i no a l'original.

- Creació final del RPST: si tot va bé el TCT construït prèviament coincidirà amb el RPST resultat del graf.

8 Two-Terminal Graph, Graf amb Dos Terminals
9 Triconnected Components Tree, Arbre de Components Triconnexos

Aquest RPST ens permetrà distingir entre diferents fragments i regions especials del model per planificar més endavant l'ordre dels camins. Ara bé, poden existir rígids dins el RPST. Aquestes regions són les més complicades de tractar ja que estan mal estructurades i dificulten una correcta generació de text.

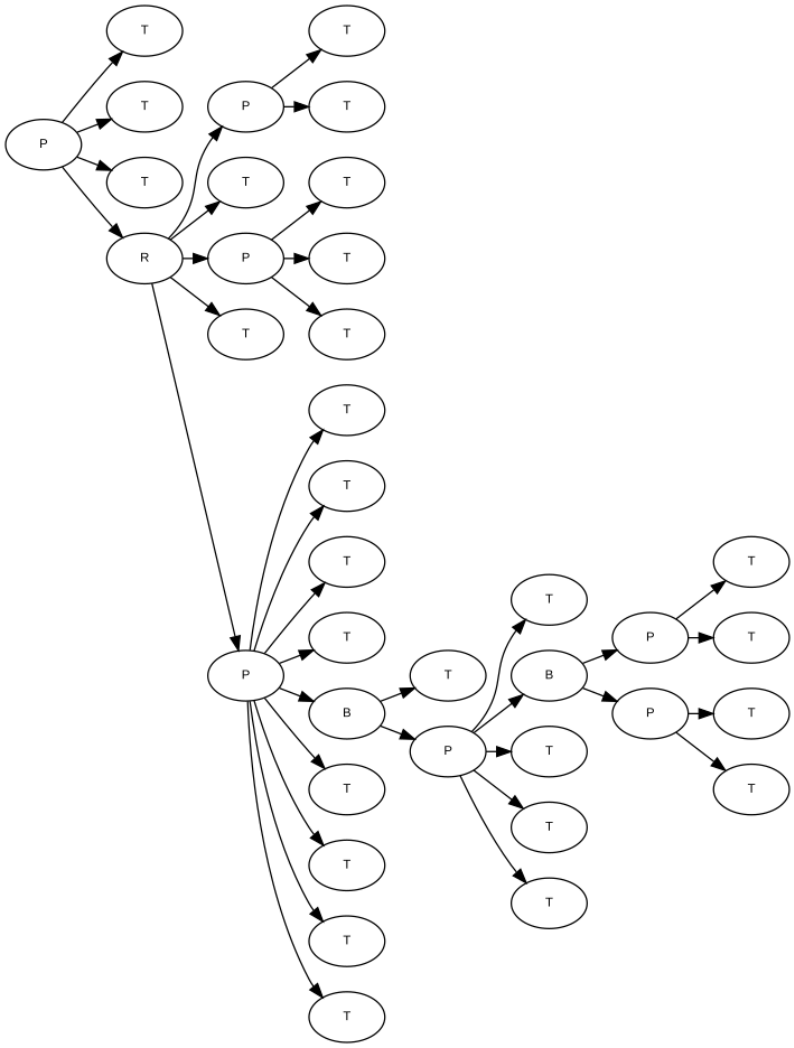


Figura 11: Representació de l'arbre RPST resultant

Per arreglar-ho, aplicarem part de les tècniques del treball «Automated Discovery of Structured Process Models: Discover Structured vs. Discover and Structure»[8]. En el document s'explica un algorisme per reestructurar rígids i segons les seves característiques un mètode, BPStruct, o un altre, Extended Oulsnam. Dins d'aquest últim mètode, trobem dos operadors a aplicar dins el graf i que permeten que un graf torni a ser ben estructurat.

Si volem aplicar aquests operadors, abans de tot hem de trobar les anomenades injeccions i ejeccions, és a dir, els conjunts de 4 portes connectades pel flux del model amb una de les portes com a punt central que . En el cas de les injeccions, trobem dues portes precedint el punt central i una a la sortida. Per el cas de les ejeccions, hi ha una porta precedint el centre i dues a la sortida.

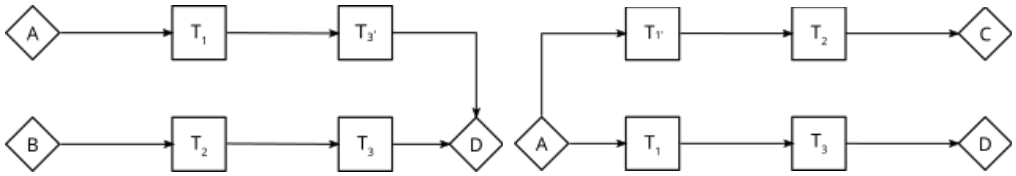


Figura 12: Operadors aplicats: a) push-down i b) pull-up

Els operadors, anomenats push-down per les injeccions i pull-up per les ejeccions, normalment s'apliquen fins que no quedin injeccions o ejeccions però en el projecte actual es realitzaran un nombre determinat de canvis. El resultat de la seva aplicació és que els elements en comú dins de la injecció o ejecció queden duplicades i les portes centrals s'esborren, com es veu a la figura 4.

En cas de quedar regions sense estructurar s'aplicarà una variant del mètode BPStruct per crear els camins correctes dins el rígid i generar textos en un ordre comprensible.

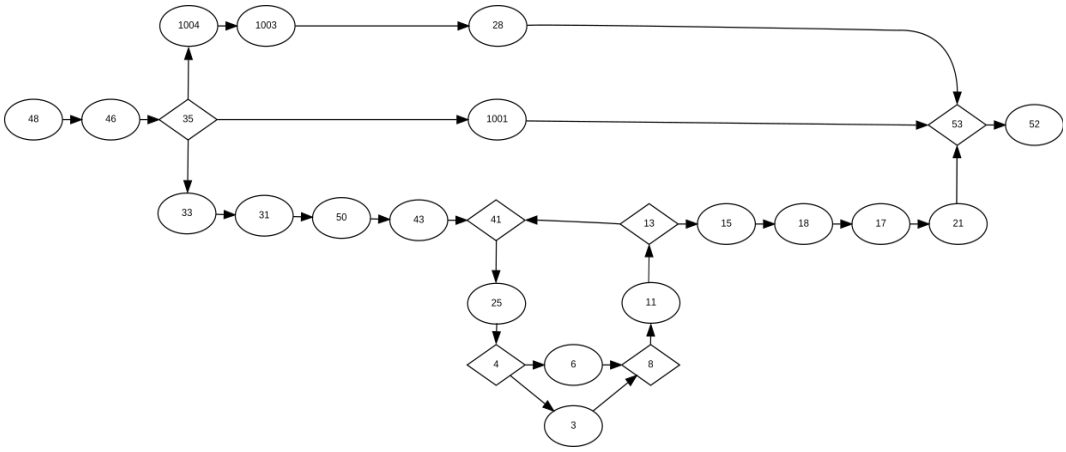


Figura 13: Graf del model després de reestructurar

7.4. Planificació del text i creació de DSynS

Amb el RPST creat i millor estructurat, el següent pas consisteix en recórrer els nodes i generar les estructures de sintaxis profunda, o DSynS corresponents. Aquesta fase és una de les més importants ja que aquí és on s'agafa tota la informació obtinguda anteriorment i es converteix en les estructures de sintaxis profunda necessàries pel realitzador de text.

Primer de tot, s'ha de generar un ordre pels nodes basant-se en l'arbre RPST i el graf del model. Llavors, anem recorrent el llistat de nodes. Segons el tipus d'element BPMN, de node RPST i de nodes adjacents, s'executaran diferents codis que crearan les DSynS adequades.

El cas més simple és el de les tasques. Les anotacions obtingudes anteriorment a partir de les etiquetes contenen informació com accions i objectes. El propi node també conté informació com a quin carril o piscina es troba, per tal de saber qui o què realitza l'acció de la tasca. La estructura sintàctica que es crea és senzilla. Primer, s'introdueixen l'acció, l'objecte, el subjecte i les dades addicionals. Després, es comproven característiques com si l'objecte és un paraula en plural o ja incorpora un determinant, per exemple. Per últim, si escau, s'afegeixen modificadors a l'inici o al final de la frase. Un cop tot està dins l'estructura, s'afegeix la DSynS a la llista que conforma el pla de frases.

Per les portes, és a dir, les condicions dins el procés, l'estructura es realitza de forma distinta. El primer de tot és obtenir el tipus de la porta. Per les portes AND, que precedeixen zones paral·leles, es crea un text depenent del número de branques sortints de la porta. Per les de tipus XOR, si només tenen dues branques i la respostes són Si/No, el text resultant seguirà la següent estructura: «IF + condició de l'etiqueta». En canvi, si el número de branques és major a dos o si no hi ha etiqueta, s'indica que només una de les branques serà executada.

Un cas especial és el dels rígids perquè no estan ben estructurats i no es pot simplement escriure cada opció. Per tant, el primer que cal fer és descobrir quins camins possibles hi ha dins aquesta regió. La base per fer-ho serà el mètode BPStruct. Primer, es transforma la regió en una xarxa de Petri. Després, calculem els recorreguts concurrents que hi ha des del node inicial i agafem el més llarg. Aquest recorregut serà el camí principal del rígid, mentre que els altres recorreguts seran les desviacions del mateix.

7.5. Realització del text final

La realització final del text és un tema vital pel projecte. El realitzador escollit ha de transformar tota la informació que hem recopilat i, com a resultat, ens retorna les frases que volem.

La realització de textos és un àrea d'estudi dins el camp de la Generació de Llenguatge Natural l'objectiu del qual és convertir les dades que se li transmeten en oracions que compleixin les regles sintàctiques, morfològiques i ortogràfiques de cada idioma. Aquestes dades a transmetre són normalment abstraccions de les oracions que desitgem. Cada tipus de realitzador segueix un esquema de passos diferent en el seu procés, però la majoria necessiten d'un lèxic amb una gran quantitat de paraules del idioma i les seves característiques. Gràcies a aquest lèxic, el realitzador pot reconèixer les paraules del text i les transforma per a què sigui coherent amb els mots adjacents.

En el projecte, hem utilitzat dos realitzadors: RealPro i SimpleNLG. El primer, RealPro, pertany a la companyia CoGenTex, dedica a la generació del llenguatge natural. Al treball d'Henrik Leopold és aquest el realitzador escollit per generar el text i, per tant, va ser la primera opció per incorporar-la al projecte. Aquest realitzador escrit en Java utilitza com a base la Teoria del Sentit-Text, que explica entre moltes definicions la separació entre la sintaxis profunda i la sintaxis superficial.

Es descriu la sintaxis profunda com aquells atributs i relacions sintàctiques que s'utilitzen per igual a tots els idiomes, de manera universal; en canvi, la sintaxis superficial conté dades més específiques per a un idioma en concret. Com a exemple, es posa el cas en què a una estructura de sintaxis profunda li poden correspondre diferents estructures de sintaxis superficial, depenent de les regles que s'hagin aplicat sobre l'estructura original.

RealPro prové d'una empresa establerta i disposa d'una llicència acadèmica que ens utilitzar-lo com a part del projecte durant un cert temps. Ara bé, el realitzador només funciona amb llengua anglesa i el seu codi és com una caixa negra, no podem veure el seu funcionament intern. Per tant, impossibilita la capacitat de millora per aquesta banda.

En aquest moment és quan comencem a buscar alternatives a aquest producte. El camp dels realitzadors és bastant ampli però cadascun d'ells utilitza sistemes diferents per realitzar un text. Alguns requereixen de sistemes complexos basats en programació funcional, d'altres utilitzen estadístiques i lèxics, etc. En el nostre cas, vam trobar que l'alternativa més adequada era SimpleNLG.

SimpleNLG és un projecte originàriament desenvolupat per Ehud Reiter, professor de la universitat d'Aberdeen i cofundador de la companyia ARRIA NLG. Actualment, es un projecte de codi obert amb llicència MPL allotjat a GitHub.

Una de les grans virtuts de SimpleNLG és la quantitat d'idiomes que pot abastir, gràcies a la possibilitat de modificar el seu codi. Existeixen projectes que han adaptat, modificat o afegit idiomes com per exemple: SimpleNLG-EnFr, de la Universitat de Montreal; SimpleNLG-IT; o SimpleNLG-ES, de la Universitat de Santiago de Compostel·la.

Per començar, vam optar per agafar la versió original de SimpleNLG per començar a treballar amb ella. Donat que SimpleNLG no accepta directament les estructures sintàctiques que RealPro si rebia, cal crear una classe que transformi les dades de les estructures en trucades a l'API de SimpleNLG.

El realitzador funciona adequadament dins el projecte i genera un text similar al que retornava el RealPro, amb l'avantatge de que podem millorar aquests resultats retocant el codi de la llibreria *op*en-so*urce.*

Per altra banda, atès que no hi ha una versió de SimpleNLG per llengua catalana, s'ha començat un fork del SimpleNLG-ES per abastir aquest idioma. El SimpleNLG-CAT, com s'ha denominat, segueix en desenvolupament però es pot utilitzar actualment per casos senzills.

El desenvolupament d'aquesta versió va començar amb el fork a la versió espanyola i amb la generació del lexicon, el conjunt de paraules i característiques que s'utilitza a SimpleNLG. Per a la versió catalana, s'ha generat el lexicon a partir de les dades lingüístiques que conté Freeling per cada idioma.

Un altre detall que s'ha incorporat va ser el mòdul de morfofonologia. El català aplica en moltes ocasions les elisions fonètiques («l'» o «d'»), cosa que a l'espanyol no n'hi ha. Per poder aplicar aquestes transformacions respectant l'estructura del codi actual, es va cercar en les versions d'altres llengües romàniques per veure com ho implementaven. Finalment, s'ha agafat com a base per aquest mòdul la implementació de la versió italiana, SimpleNLG-IT.

7.6. Suport multilingüe

Un dels objectius del projecte és proporcionar un sistema que funcioni amb models de processos de diferents idiomes i es pugui generar textos seguint les regles del idioma. Per tal d'aconseguir-ho, s'han dut a terme diverses modificacions i desenvolupaments.

La primera decisió presa va ser el canvi de la llibreria utilitzada per l'anàlisi de les etiquetes. El treball de l'Henrik Leopold utilitza una llibreria (JWNL) que funciona com a interfície a WordNet. La llibreria permetia conèixer la categoria gramatical o el lema, entre d'altres característiques, de les paraules. El problema és que WordNet, i per tant també la seva llibreria, només disposen de l'idioma anglès.

Amb l'objectiu de donar suport a altres llenguatges, era necessari cercar una alternativa similar a WordNet i la llibreria JWNL. Dintre de les opcions disponibles, la escollida va ser FreeLing, la llibreria de processament del llenguatge natural creada pel centre de recerca TALP de la UPC. FreeLing inclou dades lingüístiques de diversos idiomes, des de l'anglès i l'espanyol fins el gal·lès o l'eslovè, que permeten utilitzar totes les seves eines per aquestes llengües.

Els avantatges d'utilitzar FreeLing són molt amplis. Disposa d'eines com l'anàlisi morfològic, l'etiquetadora de categoria gramatical o parsers que permeten obtenir els detalls concrets que vulguem de les paraules. Gràcies a què el projecte és de codi obert i està ben documentat, és possible localitzar les eines que necessitem a cada moment i veure el seu funcionament intern, a més d'explicar com introduir nous idiomes dintre de la llibreria.

Una altra part indispensable és adaptar l'organització de les classes Java per tal d'incorporar els nous idiomes, si escau. La primera versió del codi només generava text pels models amb etiquetes en anglès. Les classes, per la seva banda, estaven concretament fetes per aquest idioma, per exemple, hi havia llistes predefinides amb totes les preposicions de l'idioma o s'utilitzava característiques úniques de la llengua per accelerar una comprovació.

La millor manera de solucionar-ho és creant classes abstractes amb les funcions que comparteixen o poden compartir totes les subclasses, una per idioma. Això permet que incloure nous idiomes en el futur sigui una tasca més senzilla pels desenvolupadors. Es podrien aplicar més patrons de disseny per tal d'acomodar-ho millor en posteriors versions.

Tal com s'ha comentat a l'apartat anterior (Realització del text), RealPro va ser el primer realitzador utilitzat atès que era l'utilitzat al treball original. Un cop ens vam assegurar que no es podien afegir altres idiomes fàcilment, es va escollir SimpleNLG com a substitut.

SimpleNLG ens permet afegir idiomes i provar les característiques multillenguatge de les que disposa el programa. Durant les primeres reunions amb els directors, es va plantejar la possibilitat d'afegir més idiomes comprensibles pel programa a banda de l'anglès. Finalment, s'ha decidit incorporar el català i l'espanyol.

8. Resultats

Al final de tot, quan el projecte s'ha acabat de desenvolupar o almenys ha arribat a una fase molt avançada, ens trobem en el moment d'analitzar els resultats obtinguts pel projecte.

8.1. Comparativa amb altres projectes

Una manera de veure quines millores aporta la solució implementada al projecte actual és mitjançant comparacions amb els resultats d'altres projectes similars.

Per una banda, tenim el projecte fet per un antic alumne sobre aquest mateix tema. El seu projecte també tenia la intenció de generar descripcions a partir de models però es basava només en generar l'arbre RPST i crear text a partir de patrons. Per reestructurar els possibles rígids, es transformava la regió en una xarxa de Petri i s'aplicava un mètode similar al BPStruct.

Un altre mètode de generació de descripcions disponible és el codi escrit pel un dels autors del treball «Supporting Model Process Validation through Natural Language Generation», Henrik Leopold. Ell utilitza les eines que s'esmenten a l'article, com JWNL (llibreria interfície de WordNet, el lèxic en anglès), o RealPro (el realitzador). Aquest mètode només reconeix models de processos amb etiquetes en anglès, per tant la comparació només es pot realitzar en aquest idioma.

En els següents extractes es poden veure les diferències entre cada versió:

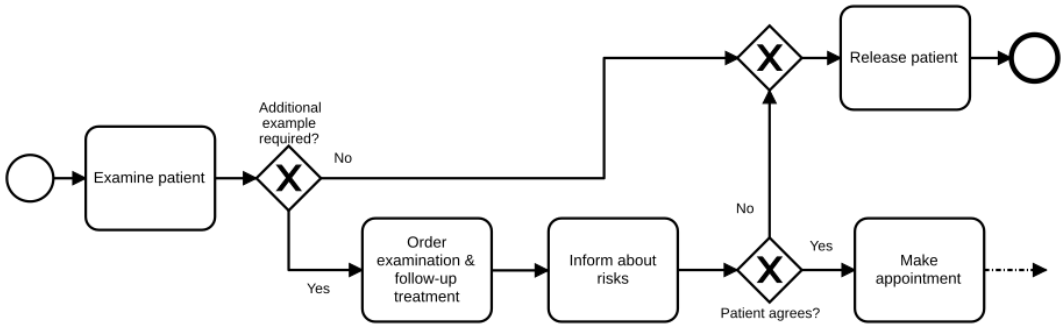


Figura 14: Extracte del model BPMN Hospital

The user hospital does the next things. First, hospital validate sample state, and then there are 2 things to be done at the same time: - On the group 1, starting with hospital send sample. [...]	The process begins when the Hospital examines a patient. Then, one of the following branches is executed: - The Hospital conducts the release patient. [...]

Text 2: Descripció corresponent a l'extracte de la Fig. 13.
Versió antic alumne (esquerra) i versió projecte actual (dreta).

Al Text 2, podem veure la descripció genera per a l'inici del procés Hospital (Figura 13). Al text de l'esquerra, la versió de l'antic alumne, podem veure que, per alguna raó, l'activitat «Examine patient» no es troba. En canvi, mostra la frase «hospital validate sample state», que és incorrecta perquè la forma del verb no es correspon amb la tercera persona del singular en anglès. Una possible raó és que la seva generació del RPST hagi fallat i l'ordre dels nodes no es correspongui amb el model. En el nostre cas, detectem la primera activitat i li afegim el text previ «The process begins when...».

A partir de la porta exclusiva amb l'etiqueta «Additional example required?» podem veure que hi ha un rígid. Es pot detectar si ens fixem en que la porta «Patient agrees?» connecta amb la mateixa porta que també és adjacent a la primera de totes. Aquest és un exemple de ejecció. A la nostra versió, primer s'aplicaran els operadors de Oulsnam (pull-up, en aquest cas concret) per desfer-nos del rígid i després es generaran totes les opcions disponibles. En canvi, a l'altre text veiem que no s'ha detectat el rígid correctament i, per tant, no mostra els camins que pertoquen dins d'aquesta regió.

The process begins when the Hospital examines a patient. Then, the process contains an a region which allows for different execution paths. One option from start to end is the following. - The Hospital makes the appointment. [...]	The process begins when the Hospital examines a patient. Then, one of the following branches is executed: - The Hospital conducts the release patient. [...]

Text 3: Descripció corresponent a l'extracte de la Fig. 13.
Versió Henrik Leopold (esquerra) i versió projecte actual (dreta).

Al text 3, podem veure la descripció que resulta del codi d'Henrik Leopold, un dels autors del treball que serveix de base en aquest. L'inici d'ambdós textos reflecteix la existència de la primera activitat. Es pot veure també que s'ha detectat el rígid ja que es mostra un text predefinit que ens ho indica. Ara bé. Per tal de trobar els camins dins el rígid, el codi transforma el rígid en una xarxa de Petri, com en el cas anterior, i cerca un camí principal i les seves desviacions. Ara bé, l'algorisme a vegades falla com en el cas exposat, on veiem que una de les opcions comença amb l'activitat «Makes appointment» i aquesta no és adjacent a la primera porta del rígid.

8.2. Test d'estrès

Per provar l'eficiència del projecte, s'han realitzat execucions sobre conjunts de processos. El conjunt de processos escollits és el de «Admissions d'estudiants a universitats».

Nom del procés	**Nº d'elements**	**Temps d'execució**
Münster	88	13864 ms
Hohenheim	55	13831 ms
IIS Erlangen	62	10562 ms
Frankfurt	40	10164 ms

Taula 2: Temps d'execució i nombre d'elements per procés del conjunt

Com es pot veure en els temps d'execució, la mitja del temps ronda al voltant dels 10 segons. La major quantitat del temps és ocupat per l'anotació, malgrat que l'actual analitzador d'etiquetes millora la duració en comparació amb la primera versió.

9. Aplicació: Generació automàtica de converses per un chatbot

9.1. Introducció

La generació de descripcions de processos en llenguatge natural ens proporciona funcionalitats que poden ser aplicades en altres situacions o en altres problemes.

Una d'aquestes aplicacions és generar una arbre de diàleg per un chatbot. L'objectiu és que mitjançant aquest diàleg l'usuari pot seguir pas a pas un procés o fer preguntes relacionades amb algun element. Serveix com una ajuda per les persones que volen realitzar algun tràmit com, per exemple, demanar dies lliures dins la intraweb de l'empresa.

Durant la realització del TFG, se'm va oferir col·laborar en aquest projecte i el vaig acceptar. La millor part de tot és que s'aplicarien la gran majoria de conceptes apresos i de les funcionalitats que estaven fetes pel generador de descripcions. Això permetia que al millorar els algorismes d'una banda, també estava treballant en les converses de chatbot.

9.2. Desenvolupament del projecte

Tal com s'ha dit, les fases inicials de la generació de l'arbre són iguals o similars a les de la generació de les descripcions. La lectura de l'arxiu BPMN amb la informació del model, l'anotació dels elements del model i la posterior construcció del seu graf són idèntics entre dos. Els següents passos a partir d'aquest és on els processos de generació divergeixen.

Amb l'anàlisi lingüístic realitzat i el graf ja construït, comencem amb la fase de generació de l'arbre de diàleg. Per poder generar un arbre de diàleg adequat, és necessari fer diverses passades sobre el graf del model i sobre l'arbre de diàleg generat.

La primera passada consisteix en generar els nodes de diàleg seguint el flux del graf i complint certes restriccions. Si generéssim un node de diàleg per cada node del graf, els nous nodes que corresponen a elements sense etiqueta o sense cap significat útil serien confusos per una persona i no tindrien cap sentit. Per exemple, una porta AND que uneix branques ens indica el fi d'una zona paral·lela però per a un usuari que no coneix la terminologia BPMN i que només vol seguir un pas a pas, això no li dóna cap informació important. Per tant, les normes d'aquesta primera passada ignoren o agrupen nodes del graf.

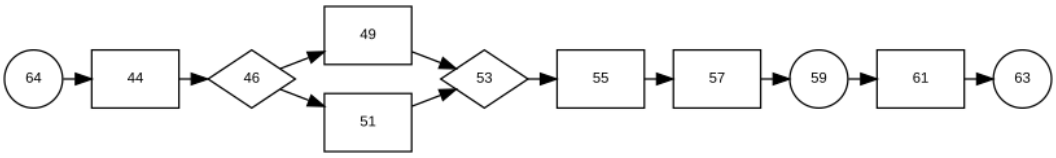


Figura 7: Exemple de graf d'un model

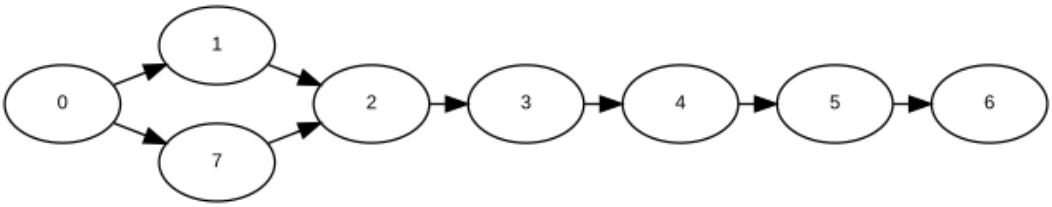


Figura 8: Graf amb els nodes de diàleg creat a partir de la Figura 7

Un cop creats tots els nodes i assignats cadascun amb un identificador numèric, procedim a repassar els eixos del graf del model i connectar els nodes del diàleg entre ells segons aquests eixos. Cada node creat conté una llista amb el elements del graf relacionats amb ells. Això ens permet connectar els nodes de manera similar al graf, mantenint l'ordre.

Després, generem per cada node del diàleg les estructures sintàctiques (DSynS) que li pertoquen. Utilitzem les dades dels anàlisis a les etiquetes per definir el subjecte, l'acció i l'objecte, entre d'altres. S'intenta que per cada element del graf relacionat amb el node es creï una DSynS. A més, segons els elements que el precedeixen o el succeeixen, s'afegeixen modificadors o frases abans i/o després de la frase principal.

El següent pas és la detecció i re-estructuració de les zones paral·leles. Denomino com a zones paral·leles les regions delimitades per portes AND i que estan ben estructurades. Donat que el usuaris no tenen perquè conèixer els conceptes de branques i per tal que no hagin de tornar cap a l'inici de la zona moltes vegades, reorganitzem les branques amb la intenció de tenir totes les combinacions de branques ja calculades o, si hi ha masses branques, almenys una d'elles.

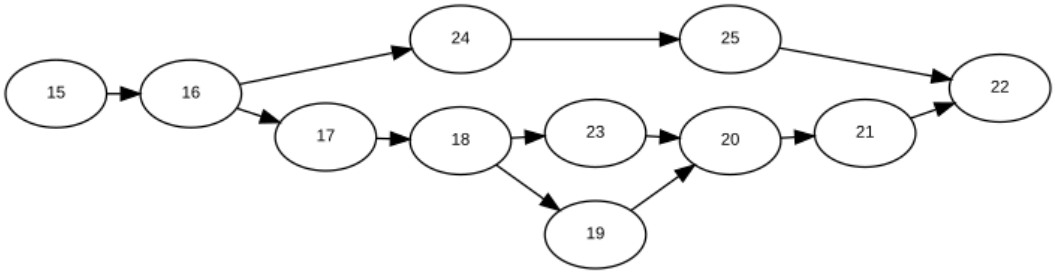


Figura 9: Exemple de graf de diàleg previ a les zones paral·leles

El procediment per fer-ho és començar detectant les zones paral·leles i les seves branques dins l'arbre de diàleg. Ara, calculem les combinacions de branques possibles si la seva quantitat és menor a un número determinat. La raó és que crear un número excessiu de branques pot ser encara pitjor que la situació original. Finalment, dupliquem i ajuntem les branques de cada combinació. Aquestes operacions es fan recursivament des de les zones més petites fins les més grans.

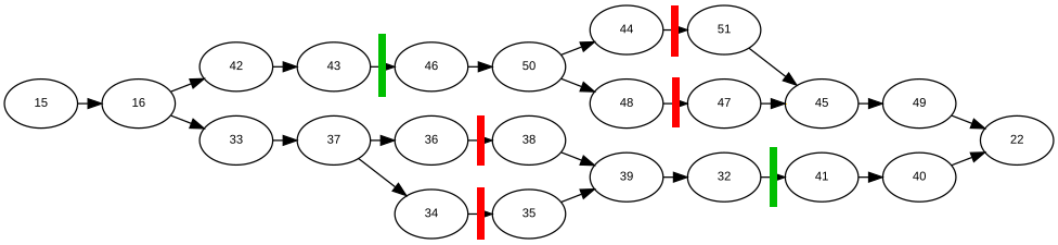


Figura 10: Exemple d'un graf (Fig. 7) transformat

A partir d'aquest moment, ja no modifiquem més l'estructura de l'arbre de diàleg. El pas a realitzar ara és la generació del text. Cada node de conversa té una llista amb el DSynS a realitzar. En cas de que no n'hi hagi cap, s'assigna un text predefinit o buit, depenent del context. El realitzador escollit és, al igual que en treball original, el SimpleNLG.

Pels eixos, sobretot aquells que tenen com a origen una porta, s'agafa l'etiqueta que tenen i s'assigna directament com a opció del node de conversa origen. Si no tenen text, s'escriu per defecte «Next». Al acabar aquest pas tindrem transformat tot el graf del model en nodes de converses amb les seves referències.

Un altre tipus de preguntes que necessitem són les que pregunten sobre els elements del model, com per exemple, si un usuari pregunta qui és el responsable d'una tasca. El generador crea nodes de conversa amb les respostes a preguntes possibles juntament amb una llista de frases clau com la pregunta en si. Juntament amb això, el node conté una referència al node del qual es fa la pregunta.

Per últim, tots els nodes creats i les seves referències passen cap a l'exportador. L'exportador que s'ha implementat té dos formats destí: XML i JSON. En aquesta fase simplement recorrem els nodes de conversa creats juntament amb les referències i, per cada un d'ells, agafem la seva informació per construeix l'objecte en el format desitjat.

```
<node id="38" text="Starting here, tasks groups are executed at the same time. Which one do you want to start with?">
    <option ref="34" text="Start with the ZooClub department waits for the payment."/>
    <option ref="36" text="Start with the Billing department sends the payment request."/>
  </node>
```

Text 4: Exemple de node de conversa XML

```
{   "id": "38", "text": "Starting here, tasks groups are executed at the same time. Which one do you want to start with?",
    "option": [{
        "ref": 36, "text": "Start with the Billing department sends the payment request."
    }, {
        "ref": 34, "text": "Start with the ZooClub department waits for the payment."
    }]
}
```

Text 5: Exemple de node conversa JSON

10. Gestió econòmica

10.1. Consideracions i comentaris

La realització d'un projecte, per tal de que aquest arribi a bon port, requereix tenir en compte aspectes com la previsió dels costos. En aquest apartat, es mostrarà l'estimació de costos de les tasques descrites abans, dels recursos necessaris i altres costos indirectes que es poden derivar de les tasques.

Durant l'execució del projecte, s'aniran actualitzant les xifres dels preus i els costos, si escau. Aquestes dades s'estaran controlant per tal de no sobrepassar les previsions. Els elements que surten als pressupostos sorgeixen o estan relacionats amb les tasques exposades al diagrama de Gantt de la planificació.

10.2. Recursos humans

El projecte serà desenvolupat només per una persona, la qual haurà d'assumir els 3 rols existents dins del projecte: cap de projecte, desenvolupador i *b*eta-tester. A la següent taula es mostren les hores previstes per dur a terme les tasques i els preus estimats del temps a dedicar en el projecte que ha de gastar l'empresa. Els preus bruts han sigut extrets dels estudis de remuneració 2017 de la consultora Michael Page:

Rol	Hores	Preu per hora	Total
Cap de projecte	172h	50€/h	8600€
Programador	460h	35€/h	16100€
Beta-tester	72h	30€/h	2160€
TOTAL	704h		26860€

Taula 3: Preus per cada rol i cost total

10.3. Hardware

Amb l'objectiu de implementar i desenvolupar, cal complir amb uns requeriments de hardware per tal de programar, documentar i fer proves en condicions òptimes. En cas de necessitar reparacions o altres modificacions, encara que els preus puguin ser molt variats no s'hauria de superar els 300 euros, com a molt.

Producte	Preu	Unitat	Vida útil	Amortització
Ordinador portàtil	750€	1	4 anys	63€
Servidor web	480€	1	6.5 anys	40.32€
Reparacions	300€			
TOTAL	1530€			

Taula 4: Preus del hardware utilitzat

El servidor web mostrat a la taula representa el lloguer mitjà anual d'un servidor web com a *PaaS*[10], que seria la opció més econòmica en cas d'allotjar el servei pel nostre compte. En el cas concret del nostre projecte, s'utilitzaran els servidors de la universitat per allotjar el programa, per tant aquest cost es redueix. Les amortitzacions s'han calculat utilitzant el percentatge donat per l'Agència Tributària pels equips informàtics, un 20%.

10.4. Software

El *software* és una part fonamental del projecte i totes aquestes eines seran utilitzades al llarg d'ell.

10 Platform as a Service (Plataforma com a servei)

Producte	Preu	Unitat	Vida útil
TexMaker	0€	1	3 anys
Eclipse JEE IDE	0€	1	3 anys
FreeLing	0€	1	3 anys
ProjectLibre	0€	1	3 anys
Linux Mint 18.0	0€	1	3 anys
RealPro	0€	1	3 anys
TOTAL	0€		

Taula 5: Cost del software utilitzat

Com es pot veure, la totalitat del programari escollit es gratuït. La majoria tenen llicències que permeten el seu ús sense costos (CPAL, Affero GNU GPL, ...), d'altres tenen llicència acadèmica, que dóna accés gratis al programa sempre que sigui per ús educatiu.

10.5. Despeses indirectes

En un projecte informàtic, la despesa més important és la d'energia elèctrica. Com també s'ha d'escriure una memòria, el cost del paper també entra en aquest apartat.

Producte	Preu/Unitat	Unitats	Cost
Electricitat	0.15€/(kW/h)	63.18kW/h	10.14
Paper	0.01€	100 fulls	1€
TOTAL			11.14€

Taula 6: Despeses indirectes

10.6. Control d'imprevistos

Durant el projecte, hi ha la possibilitat de que apareguin incidències o problemes de diferent gravetat que poden afectar al pressupost estimat. Per tal d'evitar grans desviacions o canvis en els costos, s'utilitzarà el diagrama de Gantt per revisar el temps de les tasques i reorganitzar el calendari adequadament. Les reunions amb els directors de projecte serviran també com a espai per decidir les solucions o actuacions pertinents a cada cas.

10.7. Pressupost total

Finalment, ajuntant totes les parts descrites podem veure el pressupost total estimat del projecte.

Concepte	Cost aproximadament
Recursos humans	26860€
Hardware	1530€
Software	0€
Costos indirectes	11€
Imprevistos	4215€
TOTAL	32616€

Taula 7: Pressupost total

Es poden restar els 480 euros del servidor cloud web ja que s'utilitzarien els serveis de la universitat. S'ha decidit que la partida per imprevistos sigui d'un 15% del total original, ja que és el percentatge més recomanat per aquests casos. Si tot va bé, els diners destinats a imprevistos no seran gastats i es descomptaran del pressupost final. Per tant, en un cas òptim, el cost estimat del projecte seria inferior als 28000 euros.

11. Sostenibilitat

Sostenible?	Econòmic	Social	Ambiental
Planificació	7	6	8

Taula 8: Puntuació de sostenibilitat

11.1. Àrea econòmica

S'ha dissenyat un pressupost estimat pel projecte tenint en compte els recursos que s'utilitzaran o que poden ser necessaris. Dins l'apartat de hardware s'han reservat 300€ per possibles reparacions en l'equipament. En el cas de presentar el projecte davant una empresa, per exemple, jo penso que no seria tant competitiu com m'agradaria. Hi ha etapes que es podrien reduir i cedir temps, com la fita inicial. Si es fes un canvi d'aquest tipus, el pressupost podria baixar ja que les hores del cap de projecte passarien a ser del desenvolupador o del beta-tester. S'ha intentat distribuir els temps de les etapes i les seves tasques tenint en compte els punts claus com l'arbre RPST o la realització del text.

11.2. Àrea social

Aquest projecte s'engloba dins del sector de la gestió de processos de negoci. En l'actualitat, saber com funcionen aquests processos i els punts a millorar dins ells, és un coneixement que les empreses i institucions valoren molt. Conèixer els punts febles d'un procés permet solucionar, reforçar o canviar les tasques i condicions en què es treballa per tal d'augmentar l'eficiència del treball.

Barcelona i la seva corona metropolitana és la seu de múltiples empreses i institucions, tant de nivell local com de nivell continental. Totes elles podran gaudir dels beneficis del projecte un cop aquest finalitzi. Donat que normalment els diagrames BPMN són difícils d'entendre a primera vista, la generació d'un text en llenguatge natural que descrigui el diagrama, pot aportar major comprensió sobre els propis processos als directius, caps amb poder de decisió o, simplement, gent sense coneixement del tema.

L'existència del projecte no suposa cap perjudici a cap sector de població o d'indústria. Tal com ja s'ha dit, per les organitzacions i empreses resulta beneficiòs tenir una altra manera de veure els processos a part dels diagrames BPMN.

11.3. Àrea ambiental

Degut a que és un projecte de software, la incidència mediambiental és molt baixa. Els recursos de hardware i les despeses indirectes són les que generen més impacte.

Les peces que componen l'ordinador de desenvolupament i el servidor són elements manufacturats prèviament. En el cas de l'ordinador, podem saber més o menys d'on procedeixen i quina ètica té l'empresa que els fa. Ara bé, normalment el servidor web serà propietat de tercers i, per tant, no es pot conèixer com s'han obtingut els materials, en quines condicions de treball s'ha fet o si gasta molts recursos energètics.

Pel que fa a les despeses indirectes, tenim l'electricitat i el paper. Són els que més petjada ecològica poden generar. El paper només s'utilitzarà per la versió física de la memòria. El programa en si no necessita paper pels seus resultats. El consum i la generació de l'electricitat seria l'altre punt a tenir en compte. Les condicions de fabricació d'aquests elements no les podem conèixer directament i és aquí on hi pot haver més incidència mediambiental.

De totes formes, en conjunt l'afectació no és gens alta. El codi del TFG es podria reutilitzar en projectes futurs i sense el TFG l'impacte ambiental seria fins i tot menor ja que el paper és innecessari.

12. Integració de coneixements

12.1. Coneixements previs

La utilització de grafs de processos de negoci, arbres RPST i dels missatges DSynT requereixen coneixements d'estructures arbràries i de grafs, cosa que s'aprèn a l'assignatura de Algorísmia. Un cop obtenim la informació dels arxius BPMN, necessitarem estructurar-la en un o alguns grafs que representaran aquests processos. Per muntar l'arbre RPST, recorrerem el graf, anirem creant els nodes de l'arbre corresponent i afegirem la informació lingüística de les etiquetes gràcies als algorismes de cerca en arbres. Els missatges DsynT també són arbres, per tant, ens serviran algorismes similars.

Per tal d'obtenir la informació dels processos de negoci, haurem de llegir arxius BPMN, basats en XML, i fer parsing de les etiquetes dels elements del procés. Podré aprofitar coneixements de les assignatures de Llenguatges de Programació i Compiladors. S'aplicarà un parser sobre els arxius i els seus resultats s'hauran de anar transformant gràcies a les tècniques de compiladors fins a obtenir les estructures de dades necessàries per seguir endavant.

12.2. Adequació a l'especialitat

La especialitat de Computació es defineix com aquella que capacita els estudiants per dissenyar sistemes informàtics complexos tenint en compte criteris d'eficiència, seguretat i fiabilitat, i els prepara per saber escollir els llenguatges de programació, els algorismes i les eines adequades a cada ocasió.

El projecte és adequat per la especialitat de Computació perquè, a més de estar dins el camp del processament de llenguatge natural, utilitza estructures de dades complexes que requereixen algorismes adaptats a ells. La forma de implementar-los i la eficiència d'ells són algunes de les bases de la especialitat.

Per exemple, per a la creació de l'arbre RPST s'haurà de recórrer el procès de negoci convertit en graf detectant quin tipus d'element és cada node i quines relacions té amb els elements previs i posteriors. A més, s'haurà de reestructurar prèviament certes parts del graf, tot seguint les pautes de l'algorisme Heuristics Miner.

La refinació i la realització del text final requereixen l'aplicació de tècniques de generació de llenguatge natural com els DSynS o la generació de text basant-nos en la teoria del llenguatge de Noam Chomsky. Aquestes tècniques depenen d'algorismes sobre estructures d'arbres i lingüística.

12.3. Competències tècniques

Les següents competències estan incloses al projecte:

Avaluar la complexitat computacional d'un problema, conèixer estratègies algorísmiques que puguin dur a la seva resolució, i recomanar, desenvolupar i implementar la que garanteixi el millor rendiment d'acord amb els requisits establerts (CCO1.1). El nivell és de bastant, ja que s'ha de fer recerca per trobar diferents algorismes que puguin aplicar solucions a tots els objectius parcials i també el final, buscant una eficiència raonable en la seva implementació, a més de la seva eficàcia.

Definir, avaluar i seleccionar plataformes de desenvolupament i producció hardware i software per al desenvolupament d'aplicacions i serveis informàtics de diversa complexitat (CCO1.3). S'han de buscar diferents formes fiables per a poder programar sense tenir gaires problemes o imprevistos siguin per part del codi, de l'entorn de desenvolupament o dels ordinadors. A més, s'ha de incloure tot el codi dins el servidor WildFly amb PrimeFaces. Per tot això, el nivell és de bastant.

Demostrar coneixement dels fonaments, dels paradigmes i de les tècniques pròpies dels sistemes intel·ligents, i analitzar, dissenyar i construir sistemes, serveis i aplicacions informàtiques que utilitzin aquestes tècniques en qualsevol àmbit d'aplicació (CCO2.1). Aquesta competència té un nivell de bastant. Durant el projecte, analitzarem el llenguatge humà escrit en les etiquetes, l'interpretarem, n'extraurem les dades i generarem respostes en format de text. Aquestes tècniques estan incloses dins el camp de la Intel·ligència Artificial i del Processament del Llenguatge Natural. Durant la recerca trobarem diferents algorismes per aplicar i, per aquest cas, molts d'ells entren dins d'aquesta àrea, com els support vector machines.

Capacitat per a adquirir, obtenir, formalitzar i representar el coneixement humà d'una forma computable per a la resolució de problemes mitjançant un sistema informàtic en qualsevol àmbit d'aplicació (CCO2.2). En profunditat, ja que és un requisit indispensable per tal de poder convertir en dades tot el coneixement humà necessari, en major part la informació del RPST i dels missatges DSynT. Per tal de passar del coneixement humà a informació computable s'utilitzen els diferents algorismes explicats prèviament.

13. Lleis i regulacions

13.1. Aplicades al context del projecte

Actualment, ni els processos de negocis ni el processament del llenguatge natural estan regulats per cap llei específica a Espanya.

El més pròxim a una regulació que tenen els processos de negoci és l'estàndard BPMN. Ara bé aquest només especifica els tipus de dades dins el procés i la seva visualització. Si existeixen altres regulacions vers els processos, probablement siguin d'ús intern a la empresa o institució i, per tant, estan fora del nostre abast.

13.2. Aplicades al desenvolupament del projecte

Tal com s'ha mostrat en l'apartat de Software, dins el pressupost, s'utilitza en gran mesura software lliure. Per aquesta raó, s'han de tenir molt en compte les llicències que poden tenir les llibreries utilitzades i la llicència que tindrà el projecte finalment per tal que es respectin les de les llibreries.

Referències

[1] D. Jurafsky i J. H. Martin, *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, 2009.

[2] L. Padró i E. Stanilovsky, «FreeLing 3.0: Towards Wider Multilinguality», *LREC2012*, 2012.

[3] B. Lavoie i O. Rainbow, «A Fast and Portable Realizer for Text Generation Systems».

[4] A. Gatt i E. Reiter, «SimpleNLG: a realisation engine for practical applications», *Proceedings of the 12th European Workshop on Natural Language Generation*. Association for Computational Linguistics, p. 90-93, 2009.

[5] H. Leopold, J. Mendling, i A. Polyvyanyy, «Supporting Process Model Validation through Natural Language Generation», *IEEE Trans. Softw. Eng.*, vol. 40, núm. 8, p. 818-840, ago. 2014.

[6] H. Leopold, S. Smirnov, i J. Mendling, «On the refactoring of activity labels in business process models», *Inf. Syst.*, vol. 37, núm. 5, p. 443-459, jul. 2012.

[7] A. Polyvyanyy, J. Vanhatalo, i H. Völzer, «Simplified Computation and Generalization of the Refined Process Structure Tree», Springer, Berlin, Heidelberg, 2011, p. 25-41.

[8] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, i G. Bruno, «Automated Discovery of Structured Process Models: Discover Structured vs. Discover and Structure», Springer, Cham, 2016, p. 313-329.

[9] M. Ballesteros, B. Bohnet, S. Mille, i L. Wanner, «Data-Driven Deep-Syntactic Dependency Parsing», *Nat. Lang. Eng.*, vol. 1, núm. 1, p. 1-38, 2015.

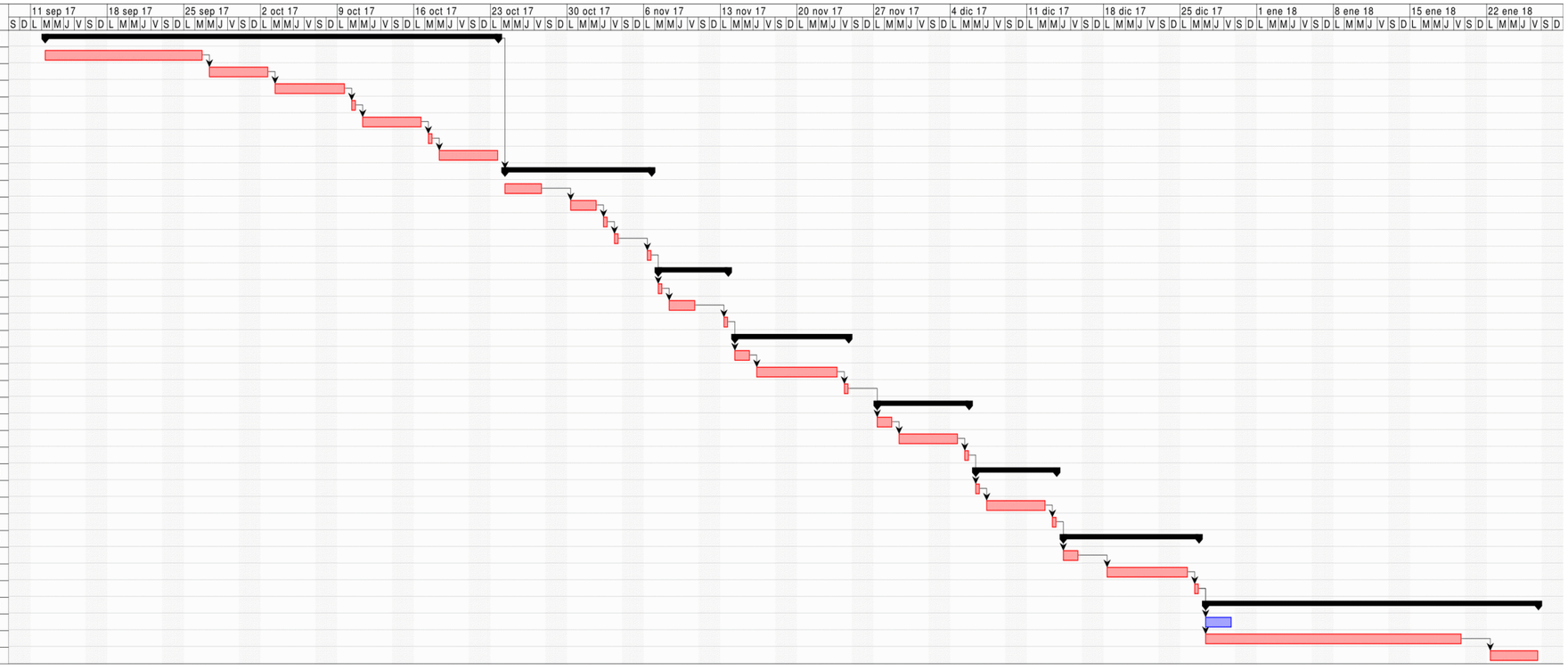
[10] B. Lavoie, R. Kittredge, T. Korelsky, i O. Rambow, «A framework for MT and multilingual NLG systems based on uniform lexico-structural processing», en *Proceedings of the sixth conference on Applied natural language processing* -, 2000, p. 60-67.

[11] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, i G. Bruno, «Automated Discovery of Structured Process Models: Discover Structured vs. Discover and Structure», Springer, Cham, 2016, p. 313-329.

Annex

Diagrama de Gantt del projecte

		Nombre	Inicio	Trabajo	Terminado
1		**Fita inicial**	**12/09/17 8:00**	**240 horas**	**23/10/17 17:00**
2		Definició de l'abast	12/09/17 8:00	88 horas	26/09/17 17:00
3		Planificació temporal	27/09/17 8:00	32 horas	2/10/17 17:00
4		Gestió econòmica i sostenibilitat	3/10/17 8:00	40 horas	9/10/17 17:00
5		Revisió de competències	10/10/17 8:00	8 horas	10/10/17 17:00
6		Presentació preliminar	11/10/17 8:00	32 horas	16/10/17 17:00
7		Referències i revisió general	17/10/17 8:00	8 horas	17/10/17 17:00
8		Presentació oral i document final	18/10/17 8:00	32 horas	23/10/17 17:00
9		**Anàlisi, disseny i preparatius del projecte**	**24/10/17 8:00**	**80 horas**	**6/11/17 17:00**
10		Anàlisi	24/10/17 8:00	32 horas	27/10/17 17:00
11		Disseny	30/10/17 8:00	24 horas	1/11/17 17:00
12		Cerca i descàrrega del material	2/11/17 8:00	8 horas	2/11/17 17:00
13		Configuració del IDE	3/11/17 8:00	8 horas	3/11/17 17:00
14		Sincronització amb altres projectes	6/11/17 8:00	8 horas	6/11/17 17:00
15		**BPMN i anàlisi lingüístic**	**7/11/17 8:00**	**40 horas**	**13/11/17 17:00**
16		Conversor XML a BPMN	7/11/17 8:00	8 horas	7/11/17 17:00
17		Anàlisi d'etiquetes BPMN	8/11/17 8:00	24 horas	10/11/17 17:00
18		Proves	13/11/17 8:00	8 horas	13/11/17 17:00
19		**Arbre RPST**	**14/11/17 8:00**	**72 horas**	**24/11/17 17:00**
20		Anàlisi i disseny	14/11/17 8:00	16 horas	15/11/17 17:00
21		Implementació	16/11/17 8:00	48 horas	23/11/17 17:00
22		Proves	24/11/17 8:00	8 horas	24/11/17 17:00
23		**Planificació i generació de missatges DSynT**	**27/11/17 8:00**	**56 horas**	**5/12/17 17:00**
24		Anàlisi i disseny	27/11/17 8:00	16 horas	28/11/17 17:00
25		Implementació	29/11/17 8:00	32 horas	4/12/17 17:00
26		Proves	5/12/17 8:00	8 horas	5/12/17 17:00
27		**Agregació de missatges i refinació**	**6/12/17 8:00**	**48 horas**	**13/12/17 17:00**
28		Anàlisi i disseny	6/12/17 8:00	8 horas	6/12/17 17:00
29		Implementació	7/12/17 8:00	32 horas	12/12/17 17:00
30		Proves	13/12/17 8:00	8 horas	13/12/17 17:00
31		**Realització del text**	**14/12/17 8:00**	**72 horas**	**26/12/17 17:00**
32		Anàlisi i disseny	14/12/17 8:00	16 horas	15/12/17 17:00
33		Implementació	18/12/17 8:00	48 horas	25/12/17 17:00
34		Proves	26/12/17 8:00	8 horas	26/12/17 17:00
35		**Fita final**	**27/12/17 8:00**	**208 horas**	**26/01/18 17:00**
36		Proves generals	27/12/17 8:00	24 horas	29/12/17 17:00
37		Redacció de memòria	27/12/17 8:00	144 horas	19/01/18 17:00
38		Presentació final	22/01/18 8:00	40 horas	26/01/18 17:00



Exemples de textos generats

- **Procés: Hospital**
  - **Diagrama BPMN**

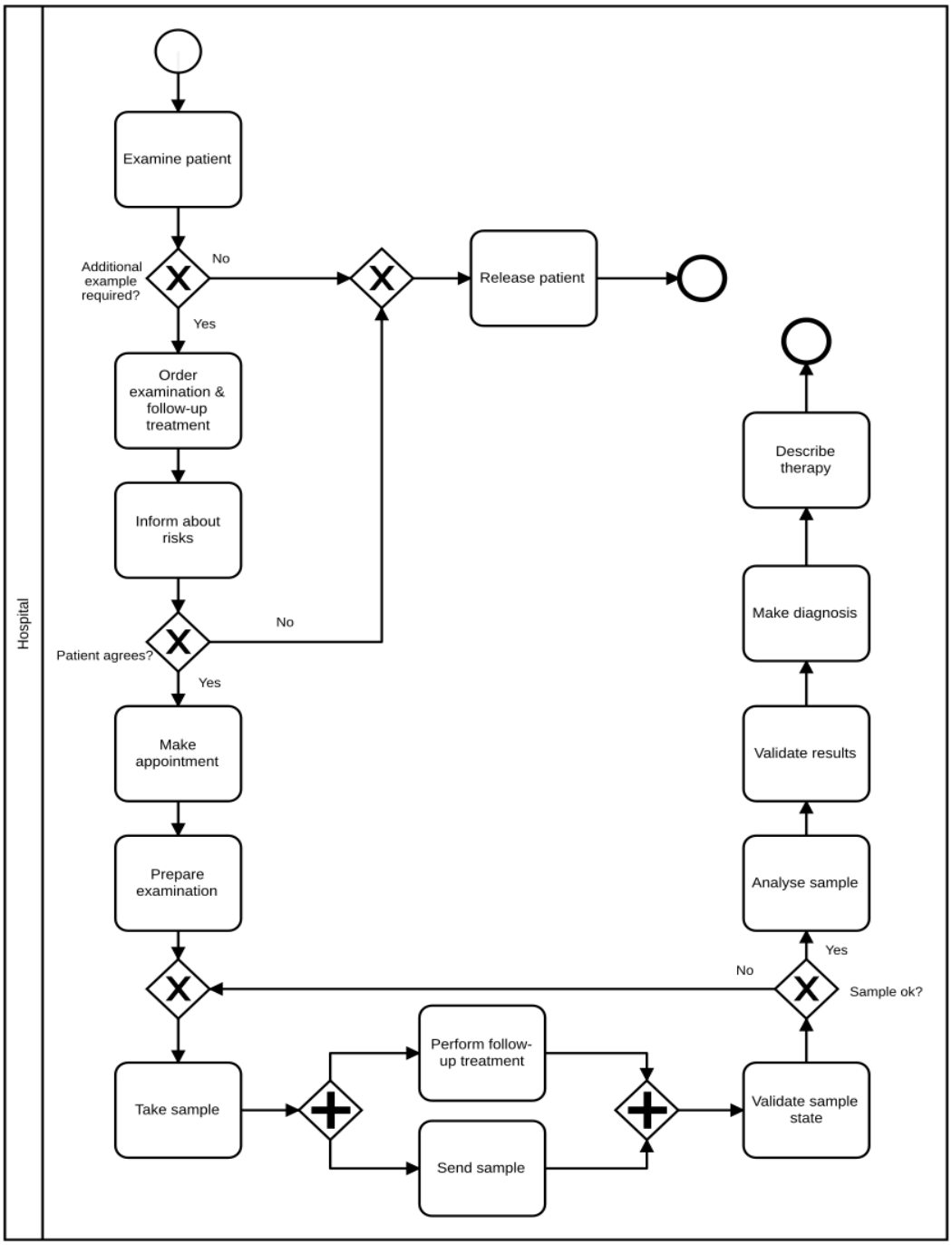


Figura 11: Diagram BPMN del procés Hospital

- **Versió antic alumne**

The user hospital does the next things. First, hospital validate sample state, and then there are 2 things to be done at the same time:
- On the group 1, starting with hospital send sample. After that, there are two parallel activities:
  * First parallel group hospital analyse sample.
  * Second parallel group hospital perform follow-up treatment.

After that, it ends with hospital make diagnosis.
- On the branch 2, this sequence starts with hospital describe therapy. At the end, hospital validate results.

- **Versió: Henrik Leopold**

The process begins when the Hospital examines a patient. Then, the process contains an a region which allows for different execution paths. One option from start to end is the following.
- The Hospital makes the appointment. Afterwards, the Hospital prepares the examination. Subsequently, the Hospital takes the sample. Then, the Hospital sends the sample. Afterwards, the Hospital performs the follow-up treatment. Subsequently, the Hospital validates the sample state.

However, the region allows for a number of deviations. Then, the process is finished.

- **Versió: Actual**

The process begins when the Hospital examines a patient. Then, one of the following branches is executed:

- The Hospital conducts the release patient.
- The Hospital conducts the order examination & follow-up treatment. Afterwards, the Hospital informs about the risk. Subsequently, the Hospital conducts the release patient.
- The Hospital conducts the order examination & follow-up treatment. Then, the Hospital informs about the risk. Afterwards, the Hospital makes the appointment. Subsequently, the Hospital prepares the examination. Then, the Hospital takes the sample. Afterwards, the process is split into 2 parallel branches:
  - The Hospital sends the sample.
  - The Hospital performs the follow-up treatment.

Once all 2 branches were executed, the Hospital validates the sample state. As long as is the Hospital repeats the steps. Once is the Hospital analyses the sample. Subsequently, the Hospital validates the result. Then, the Hospital makes the diagnosis. Afterwards, the Hospital describes the therapy.
Subsequently, the process is finished.

- **Procés: Universitat de Colònia**
  - **Diagrama BPMN**

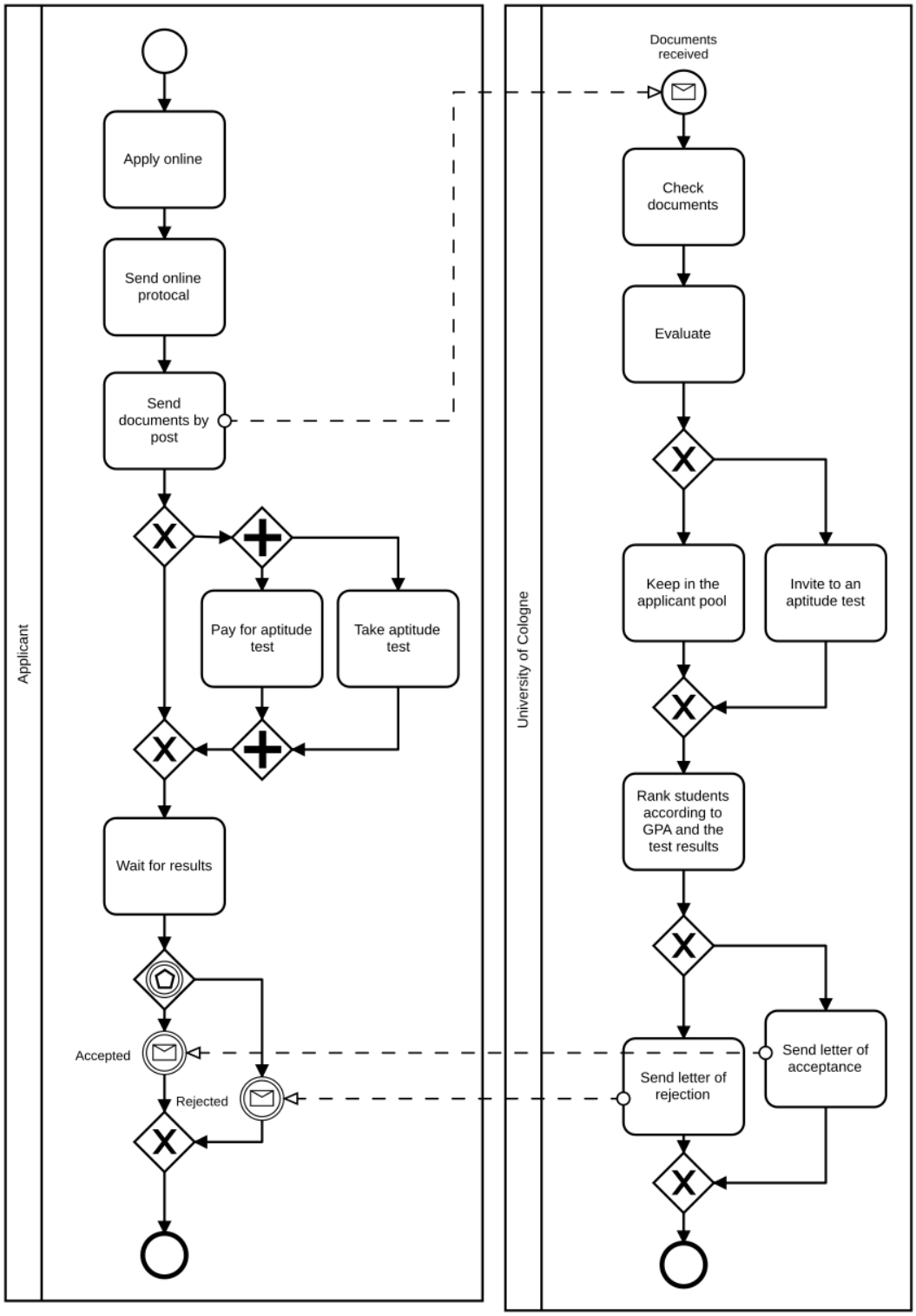


Figura 12: Diagram BPMN del procés Universitat de Colònia

- **Versió Henrik Leopold**

The Applicant process begins when the Applicant applies an online. Then, it sends the online protocal. Afterwards, the Applicant sends the documents by the post. If it is necessary, the Applicant process is split into 2 parallel branches:

- The Applicant takes the aptitude test.
- The Applicant pays for the aptitude test.

Once all 2 branches were executed, the Applicant waits for the results. Subsequently, one of the following branches is executed: Then, the Applicant process is finished.

The UniversityofCologne process begins when the UniversityofCologne checks the documents. Then, it evaluates. Afterwards, one of the following branches is executed:

- The UniversityofCologne keeps in the applicant pool.
- The UniversityofCologne invites to the an aptitude test.

Once one of the following branches was executed the UniversityofCologne ranks the students according to gpa and thes the test results. Subsequently, one of the following branches is executed:

- The UniversityofCologne sends the letter of the acceptance.
- The UniversityofCologne sends the letter of the rejection.

Then, the UniversityofCologne process is finished.

- **Versió projecte actual**

The process begins when the Applicant applies an online. Then, it sends the online protocal. Afterwards, the Applicant sends the document by post. If is it the process is split into 2 parallel branches:

- The Applicant takes the aptitude test.
- The Applicant pays for the aptitude test.

Once all 2 branches were executed, the Applicant waits for the result. Subsequently, one of the following branches is executed:

- The Applicant receives a message.
- The Applicant receives a message.

Then, the process is finished.

The UniversityofCologne conducts the check documents. Then, it evaluates. Afterwards, one of the following branches is executed:

- The UniversityofCologne keeps in the applicant pool.
- The UniversityofCologne invites to an aptitude test.

Once one of the previous branches was executed, the UniversityofCologne ranks the students according to gpa. Subsequently, one of the following branches is executed:

- The UniversityofCologne sends the letter of rejection.
- The UniversityofCologne sends the letter of acceptance.

Then, the UniversityofCologne process is finished.

Exemples de converses pel chatbot

Procés: Procure parts (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<conversation>
     <node id="5" text="The Central Purchasing conducts
the check purchase order.S Vendor known?">
          <option ref="6" text="no"/>
          <option ref="7" text="yes"/>
     </node>
     <node id="9" text="The Central Purchasing
continues with the subprocess."/>
     <node id="2" text="Then, the process is
finished"/>
     <node id="6" text="The Central Purchasing creates
the vendor.">
          <option ref="7" text="Next"/>
     </node>
     <node id="0" initial="true" text="The process
begins when the part is required by the Department.">
          <option ref="3" text="Parts out of stock"/>
          <option ref="1" text="Parts in stock"/>
     </node>
     <node id="3" text="The Department specifies the
requirement.">
          <option ref="4" text="Next"/>
     </node>
     <node id="1" text="The Department retrieves the
part from storage.">
          <option ref="2" text="Next"/>
     </node>
     <node id="8" text="The Approver approves the
purchase order.">
          <option ref="9" text="Next"/>
     </node>
     <node id="4" text="The Department completes the
purchase order.">
          <option ref="5" text="Next"/>
     </node>
     <node id="7" text="The Central Purchasing creates
the order template from purchase order.">
          <option ref="8" text="Next"/>
     </node>
     <node id="10" text="The Central Purchasing s the
Vendor known?">
          <info data="Who s the Vendor known?"/>
```

```
          <option ref="5" text="Look at the indicated
task"/>
     </node>
     <node id="11" text="The Central Purchasing
continues with the subprocess.">
          <info data="Who continues with the
subprocess?"/>
          <option ref="9" text="Look at the indicated
task"/>
     </node>
     <node id="12" text="The Central Purchasing creates
the vendor.">
          <info data="Who creates the vendor?"/>
          <option ref="6" text="Look at the indicated
task"/>
     </node>
     <node id="13" text="The Department requires the
part.">
          <info data="Who requires the part?"/>
          <option ref="0" text="Look at the indicated
task"/>
     </node>
     <node id="14" text="The Department specifies the
requirement.">
          <info data="Who specifies the requirement?"/>
          <option ref="3" text="Look at the indicated
task"/>
     </node>
     <node id="15" text="The Department retrieves the
part from storage.">
          <info data="Who retrieves the part from
storage?"/>
          <option ref="1" text="Look at the indicated
task"/>
     </node>
     <node id="16" text="The Approver approves the
purchase order.">
          <info data="Who approves the purchase
order?"/>
          <option ref="8" text="Look at the indicated
task"/>
     </node>
     <node id="17" text="The Department completes the
purchase order.">
          <info data="Who completes the purchase
order?"/>
          <option ref="4" text="Look at the indicated
task"/>
```

```
    </node>
    <node id="18" text="The Central Purchasing creates
the order template from purchase order.">
        <info data="Who creates the order template
from purchase order?"/>
        <option ref="7" text="Look at the indicated
task"/>
    </node>
</conversation>
```

Procés: Procure parts (JSON)

```
{
  "conversation": {
    "node": [{
      "id": "2",
      "text": "Then, the process is finished",
      "option": []
    }, {
      "initial": "true",
      "id": "0",
      "text": "The process begins when the part is
required by the Department.",
      "option": [{
        "ref": 3,
        "text": "Parts out of stock"
      }, {
        "ref": 1,
        "text": "Parts in stock"
      }]
    }, {
      "id": "6",
      "text": "The Central Purchasing creates the
vendor.",
      "option": [{
        "ref": 7,
        "text": "Next"
      }]
    }, {
      "id": "7",
      "text": "The Central Purchasing creates the order
template from purchase order.",
      "option": [{
        "ref": 8,
        "text": "Next"
      }]
    }, {
      "id": "9",
      "text": "The Central Purchasing continues with
the subprocess.",
      "option": []
    }, {
```

```
      "id": "8",
      "text": "The Approver approves the purchase
order.",
      "option": [{
        "ref": 9,
        "text": "Next"
      }]
    }, {
      "id": "4",
      "text": "The Department completes the purchase
order.",
      "option": [{
        "ref": 5,
        "text": "Next"
      }]
    }, {
      "id": "5",
      "text": "The Central Purchasing conducts the
check purchase order.S Vendor known?",
      "option": [{
        "ref": 7,
        "text": "yes"
      }, {
        "ref": 6,
        "text": "no"
      }]
    }, {
      "id": "1",
      "text": "The Department retrieves the part from
storage.",
      "option": [{
        "ref": 2,
        "text": "Next"
      }]
    }, {
      "id": "3",
      "text": "The Department specifies the
requirement.",
      "option": [{
        "ref": 4,
        "text": "Next"
      }]
    }, {
      "id": "10",
      "text": "The Department requires the part.",
      "info": {
        "data": "Who requires the part?"
      },
      "option": [{
        "ref": 0,
        "text": "Look at the indicated task"
      }]
    }, {
      "id": "11",
```

```
      "text": "The Central Purchasing creates the
vendor.",
      "info": {
        "data": "Who creates the vendor?"
      },
      "option": [{
        "ref": 6,
        "text": "Look at the indicated task"
      }]
    }, {
      "id": "12",
      "text": "The Central Purchasing creates the order
template from purchase order.",
      "info": {
        "data": "Who creates the order template from
purchase order?"
      },
      "option": [{
        "ref": 7,
        "text": "Look at the indicated task"
      }]
    }, {
      "id": "13",
      "text": "The Central Purchasing continues with
the subprocess.",
      "info": {
        "data": "Who continues with the subprocess?"
      },
      "option": [{
        "ref": 9,
        "text": "Look at the indicated task"
      }]
    }, {
      "id": "14",
      "text": "The Approver approves the purchase
order.",
      "info": {
        "data": "Who approves the purchase order?"
      },
      "option": [{
        "ref": 8,
        "text": "Look at the indicated task"
      }]
    }, {
      "id": "15",
      "text": "The Department completes the purchase
order.",
      "info": {
        "data": "Who completes the purchase order?"
      },
      "option": [{
        "ref": 4,
        "text": "Look at the indicated task"
      }]
    }, {
```

```
      "id": "16",
      "text": "The Central Purchasing s the Vendor
known?",
      "info": {
        "data": "Who s the Vendor known?"
      },
      "option": [{
        "ref": 5,
        "text": "Look at the indicated task"
      }]
    }, {
      "id": "17",
      "text": "The Department retrieves the part from
storage.",
      "info": {
        "data": "Who retrieves the part from storage?"
      },
      "option": [{
        "ref": 1,
        "text": "Look at the indicated task"
      }]
    }, {
      "id": "18",
      "text": "The Department specifies the
requirement.",
      "info": {
        "data": "Who specifies the requirement?"
      },
      "option": [{
        "ref": 3,
        "text": "Look at the indicated task"
      }]
    }]
  }
}
```