FIB

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

RePlan Controller: Millora i Evolució

Victor Jaria Lucha
Director: Carles Farré
Co-Director: David Ametller
Enginyeria del Software
2016/17 - Q2

Resum (Català)

En aquest projecte de final de carrera es desenvoluparà un programari, amb la seva corresponent documentació, per a un projecte anomenat SUPERSEDE.

SUPERSEDE és un projecte que proporcionarà avenços en diverses àrees d'investigació, des de la retroalimentació de l'usuari final i l'anàlisi de les dades contextuals, fins al suport a la presa de decisions en l'evolució i adaptació del programari.

El propòsit del programari que es desenvoluparà en aquest projecte, és simular, millorar i evolucionar un programa ja desenvolupat però que no té robustesa i jocs de proves. Aquest programari ja desenvolupat està programat en Ruby On Rails amb una base de dades SQL3lite. El programari que es vol desenvolupar s'ha de comportar-se igual que l'anterior, fins i tot millor. Ha d'estar desenvolupat en Java, amb una base de dades externa i amb uns jocs de proves potents i que verifiquin el millor possible el bon comportament d'aquest programari.

En aquest document es detallen tant les fases del projecte, com el pressupost, tecnologies utilitzades, requisits, implementació i disseny, entre d'altres.

Resumen (Castellano)

En este proyecto de final de carrera se desarrollará un software, con su correspondiente documentación, para un proyecto llamado SUPERSEDE.

SUPERSEDE es un proyecto que proporcionará avances en varias áreas de investigación, desde la retroalimentación del usuario final y el análisis de los datos contextuales, hasta el apoyo a la toma de decisiones en la evolución y adaptación del software.

El propósito del software que se desarrollará en este proyecto, es simular, mejorar y evolucionar un programa ya desarrollado pero que carece de robustez y juegos de pruebas. Dicho software ja desarrollado está programado en Ruby On Rails con una base de datos SQL3lite. El programario que se desea desarrollar ha de comportarse igual que el anterior, incluso mejor. Ha de estar desarrollado en Java, con una base de datos externa y con unos juegos de pruebas potentes y que verifiquen lo mejor posible el buen comportamiento de dicho software.

En este documento se detallan tanto las fases del proyecto, como el presupuesto, tecnologías utilizadas, requisitos, implementación y diseño, entre otros.

Abstract (English)

In this final degree project will develop a software, with his corresponding documentation, for a project called SUPERSEDE.

SUPERSEDE is a project that will provide advancements in several research areas, from end-user feedback and contextual data analysis, to decision making support in software evolution and adaptation. But the major novel contribution will be in integrating methods and tools from the mentioned areas, thus providing a new solution framework for software evolution and adaptation for data-intensive applications.

The purpose of the software, that will be developed in this project, is to simulate, improve and evolve a program already developed but lacking robustness and test sets. This software has been developed in Ruby On Rails with a SQL3lite database. The program you want to develop must behave just like the previous one, even better. It has to be developed in Java, with an external database and a few powerful test sets and that verify the best possible behavior of such software.

This document details the project phases, as well as the budget, technologies used, requirements, implementation, design, and others.

Índex

1. Introducció

1.1 Introducció

Aquest és un projecte de Final de Grau en l'especialitat d'Enginyeria del Software per la Facultat d'Informàtica de Barcelona. Es tracta d'un projecte en el qual es reprogramarà, millorarà i s'evolucionarà, un controlador d'un programari software per tal de tindre una versió més robusta.

El controlador, anomenat Replan Controller, en un inici està programat en Ruby on Rails + SQL3lite, i es desitja que desenvolupi en Java usant una BD MySQL. Aquest programari és una petita part d'un tot, aquest tot és el projecte anomenat SUPERSEDE[3].

SUPERSEDE, del qual parlarem profundament més endavant, té com principal objectiu proporcionar un nou marc de solucions per a l'evolució i l'adaptació de programari per a aplicacions intensives de dades.

En el document següent es mostra tota la documentació necessària, i que s'ha originat, per a un projecte d'aquestes característiques.

1.2 Actors Implicats

A continuació es mostren els principals *stakeholders* del projecte:

1.2.1. Director i co-director del projecte

Carles Farrè i David Ameller són, respectivament, director i co-director d'aquest projecte TFG. Però, alhora, formen part del projecte SUPERSEDE. Per tant:

Carles Farré i David Ameller, amb els rols següents:

- Responsables del component replan_controller dins del projecte SUPERSEDE i que han encarregat a l'estudiant Victor Jaria una versió millorada. Volen que el projecte es dugui a terme amb èxit per poder substituir el replan_controller per la versió millorada.

- Director i co-director, respectivament, del projecte. Volen que el projecte es realitzi d'acord amb les exigències d'un TFG de la Facultat d'Informàtica.

Ambdós rols, vetllaran perquè la realització del projecte sigui la idònia. Com a membres de l'equip de SUPERSEDE es centraran més en el software i pensant, en un futur, en la incorporació del software al programari de SUPERSEDE. I com a directors, es fixaran més en el fet que el TFG segueixi les pautes establertes per la Facultat.

1.2.2. Alumne

L'alumne que farà el Treball de fi de Grau i que, en conseqüència, obtindrà els corresponents crèdits i certificacions. Serà l'encarregat tant de desenvolupar el software amb els requisits necessàris que aquest comporta, com de generar la documentació necessària per a documentar el projecte, per exemple aquest document.

1.2.3. Clients

Els usuaris que en un futur utilitzaran SUPERSEDE, d'alguna manera i per contra, utilitzaran el ReplanController desenvolupat en aquest projecte.

2. Estat de l'art

2.1 Contextualització

En l'enginyeria del software el terme "fases de desenvolupament" expressa com ha progressat el desenvolupament d'un programari i quan desenvolupament pot requerir. Cada versió important d'un producte passa generalment a través d'una etapa en què s'agreguen les noves característiques (etapa alfa), després una etapa on s'eliminen errors activament (etapa beta), i finalment una etapa en on s'han tret tots els bugs importants (etapa estable). Les etapes intermèdies poden també ser reconegudes. Les etapes es poden anunciar i regular formalment pels desenvolupadors del producte, però els termes s'utilitzen de vegades de manera informal per descriure l'estat d'un producte.

Com ja s'ha dit, el projecte és part d'un software molt complet i d'una gran dimensió i impacte, SUPERSEDE. Una petita part d'aquest gran projecte és el planificador de releases, i per a aquest planificador és indispensable el Replan Controller, el nostre software a desenvolupar.

La quantitat d'aplicacions de programari en forma de serveis web, aplicacions mòbils, etc., s'ha incrementat dramàticament en els últims anys i continua fent-ho. Aquest programari explota les dades recollides a través de diversos sensors (per exemple, incrustats en el medi ambient) i fonts de dades en línia. Els usuaris finals poden accedir a aquests serveis que utilitzen una varietat de dispositius. Les tecnologies mòbils estan en augment i per tant els serveis de programari s'estan convertint en omnipresents en la nostra societat, contribuint així a millorar la qualitat de vida dels ciutadans.

Els proveïdors de programari tenen dificultats en la predicció de l'acceptació dels serveis i aplicacions que ofereixen. La gran diversitat de contextos d'execució, amb diferents perfils d'usuari, el canvi de les condicions ambientals, etc., fa que sigui difícil de personalitzar el programari per a totes les situacions possibles.

La complexitat en el maneig d'aquest tipus de sistemes resulta per oferir noves oportunitats per als enginyers de programari. La retroalimentació proporcionada pels usuaris d'aquests serveis i aplicacions, i la gran quantitat de dades disponibles quan estan sent executats, exigeix tècniques innovadores d'explotació dels mateixos amb la finalitat de superar els desafiaments. Aquesta és la principal motivació darrere del projecte SUPERSEDE.

2.2 Estudi del mercat

La planificació de releases requereix un software complex i que està en augment per millorar la productivitat i eficiència. Moltes vegades el programari no és desenvolupat per un equip principal sinó que el desenvolupament es distribueix entre equips i fins i tot entre companyies. Els dissenyadors han de tenir presents components ja existents i les dependències entre les components, sobretot en el cas d'equips treballant en llocs geogràficament allunyats. Un dels aspectes més importants són les dependències amb altres sistemes.

Com a un bon exemple de planificador de software podem destacar Release Planner[2] de Experts Decisions. ReleasePlanner és una eina que integra l'experiència i la intuïció dels experts humans amb la força computacional dels algoritmes d'optimització. ReleasePlanner ha estat provat amb èxit amb més de 500 clients i projectes de prova. La versió 2.0 incorpora innovacions com ara la capacitat d'intercanviar dades de manera transparent amb JIRA, *when-to-release planning*, i la supervisió tècnica del deute.

3. Formulació del problema

Ara per ara, es té una versió del controlador Replan Controller programada en Ruby on Rails més SQL3 Lite. Es vol que aquest controlador es millori, ja que el que hi ha actualment no està força avançat i és insuficient, i es programi amb un altre llenguatge, Java, usant una BD MySQL.

L'objectiu és que aquest nou controlador sigui més robust i consistent, i que li acompanyin uns jocs de proves seriosos. També es vol que es puguin permetre canvis en les APIs que suporta amb el mínim cost i esforç possible, per tant ha de ser prou adaptable i flexible.

4. Abast

4.1 Abast

L'objectiu principal del projecte és evolucionar, millorar i reprogramar el Replan Controller amb l'objectiu que tingui les mateixes i noves funcionalitats que l'actual controlador. A més de les funcionalitats, aconseguir que sigui un programari més robust i testejat profundament.

4.2 Possibles obstacles

4.2.1 Temps limitat

Un dels aspectes importants a tenir en compte, i més en projectes com aquest, és el temps. En aquest cas tenim un període de temps no molt ampli, uns 4 mesos aproximadament. Durant aquest temps poden sorgir molts imprevistos, i a més, si s'ha de realitzar documentació de manera paral·lela en gran part de la realització del projecte.

Mai saps amb total certesa amb qué et trobaràs a l'hora de desenvolupar software, i s'ha de tindre molt en compte amb el temps.

4.2.2 Integració de diferents softwares

En aquest projecte es reprograma, es millora i s'evoluciona un controlador ja existent en un software. Aquest canvi i reprogramació s'ha de fer el més adaptable possible per després poder-ho integrar correctament al seu projecte original. És aquí quan poden haver-hi problemes o dificultats d'adaptació.

Això comporta que en algun punt del desenvolupament del controlador s'hagin de fer canvis imprevistos i urgents induïts per les necessitats d'altres components del projecte SUPERSEDE. És a dir, s'ha de tindre en compte i conèixer el software al qual s'adaptarà aquest component.

4.3 Metodologies

Les metodologies utilitzades en aquest projecte seran: Agile i TDD (Test-driven development).

4.3.1 Agile

Ja que es disposa de poc temps per a realitzar aquest projecte la millor manera per enfocar-ho és utilitzar metodologies àgils i amb objectius concrets a curt termini.

Aquests tipus de metodologies estan pensades i destinades per treballs o projectes en grup, tot i així, hi ha moltes característiques, idees i conceptes que són aplicables en projectes com aquest encara que es realitzi de forma individual.

A més, les metodologies àgils han estat utilitzades també en assignatures de l'especialitat d'enginyeria del software, com poden ser Enginyeria de Requisits, Projectes d'Enginyeria del Software o Gestió de Projectes de Software, on en un principi es defineixen de manera concreta els objectius i requisits del projecte, s'acorden amb els stakeholders i, seguidament, es passa a desenvolupar de manera iterativa el producte.

Hi haurà, a més, un feedback constant durant tot el projecte. Les metodologies àgils es caracteritzen per un feedback constant per part del client, que en aquest cas seran a GEP els mateixos professors de l'assignatura, i el director i co-director del projecte a la part del desenvolupament del software.

4.3.2 Test-Driven Development

Desenvolupament guiat per proves de programari, o Test-driven development (TDD) és una pràctica d'enginyeria de programari que involucra altres dues pràctiques: Escriure les proves primer (Test First Development) i Refactorització (Refactoring). Per escriure les proves generalment s'utilitzen les proves unitàries (unit test en anglès). En primer lloc, s'escriu una prova i es verifica que les proves fallen. A continuació, s'implementa el codi que fa que la prova passi satisfactòriament i seguidament es refactoritza el codi escrit. El propòsit del desenvolupament guiat per proves és aconseguir un codi net que funcioni. La idea és que els requisits siguin traduïts a proves, d'aquesta manera, quan les proves passin es garantirà que el programari compleix amb els requisits que s'han establert.

Perquè funcioni el desenvolupament guiat per proves, el sistema que es programa ha de ser prou flexible per a permetre que sigui provat automàticament. Cada prova serà prou petita perquè permeti determinar unívocament si el codi provat passa o no la verificació que aquesta li imposa.

4.4 Eines de seguiment i validació

Per poder fer el projecte amb la comoditat més gran possible, i ja que es tracta d'un projecte de desenvolupament de software, s'utilitzarà un repositori online, en aquest cas GitHub[13]. Això ens permetrà per una banda accedir al codi des de qualsevol màquina i, per altra banda, que el director i co-director del projecte estiguin en constant contacte amb el qual es va desenvolupant. Per ells és important, ja que a part de ser directors del projecte també formen part del projecte SUPERSEDE, ja comentat anteriorment, que és part també del seu àmbit laboral.

A més, per tal de fer un seguiment més exhaustiu, s'ha acordat amb el director i co-director de projecte que cada dues setmanes es faran reunions d'uns 30-40 minuts per tal d'aclarir dubtes i veure l'evolució del software.

5. Calendari

El Treball de Fi de Grau té una durada aproximada de 4 mesos, si tenim en compte que comença alhora que GEP, és a dir, la tercera segona setmana de febrer, i definint com a data final el dia de la defensa del projecte. Encara que la defensa sigui a finals de juny, la intenció és acabar un parell de setmanes abans el software per poder ser integrat. En ser una planificació temporal existeix la possibilitat que en algun moment aquesta planificació variï de la que es presenta en aquest document com, per exemple, allargar la duració d'algunes tasques, canviar l'ordre d'execució d'algunes...

6. Descripció de les tasques

6.1 Anàlisi i estudi previ del projecte SUPERSEDE

La part del software que es desitja reprogramar i evolucionar està integrada dintre d'un software. El primer pas és estudiar o veure quin és el software que hi ha abans de programar res, conèixer l'API a la qual es crida, conèixer qui és SUPERSEDE i que es desitja i parlar amb els dos principals skateholders (director i co-director del projecte de fi de grau), entre d'altres. Es descarregarà una petita part del software del projecte SUPERSEDE mitjançant GitHub, el qual s'estudiarà i mirarà a fons.

6.2 Configuració de l'entorn de treball

Per tal de començar a veure/estudiar el software existent i el context del projecte cal configurar un entorn de treball local al nostre ordinador personal. L'IDE utilitzat serà Spring Tool Suite[12]. A l'Spring Tool Suite s'ha de configurar el tomcat pertinent, posar a córrer el software generat pel framework Swagger[10] i convertir el projecte de Maven a Gradle abans de començar el desenvolupament com a tal.

Aquesta fase pot comportar problemes i s'ha de tindre present a l'hora de fer la planificació.

6.3 Gestió de projectes

Aquesta fase consisteix en l'elaboració del document de gestió de projectes (GEP). Les tasques de cadascun dels mòduls i el *timing* de cada una d'elles s'extreu de les indicacions de l'assignatura. Aquesta fase no té cap tasca de dependència prèvia, però sí un calendari estricte i inflexible, ja que la data d'entrega és fixa, i amb indicacions per part del professorat de GEP.

Dintre d'aquesta tasca trobem les subtasques següents: Definició de l'abast i contextualització, Planificació Temporal, Gestió Econòmica i sostenibilitat, Presentació preliminar, Plec de condicions i presentació oral. Les tasques de cadascuna d'elles vénen definides per l'assignatura semipresencial de GEP.

6.4 Testing

Al llarg del projecte s'utilitzarà una eina de testeig d'APIs anomenada Postman[11], per tal de comparar resultats i veure que són satisfactoris. Aquesta eina s'utilitzarà per fer alguns dels testos, encara que la majoria es faran amb JUnit, ja que seran uns testos més seriosos i més còmodes de car al desenvolupador.

Primerament, força part del testeig es farà al principi del projecte, per tal de testejar el controlador que hi ha en Ruby on Rails des d'un principi, veure els resultats que s'obtenen, verificar que són correctes, buscar errors o bugs, etc. A mesura que s'evolucioni el software es faran tests per tal de validar, o no, el nou software, i així complir amb la metodologia de Test Driven Development.

6.5 Desenvolupament del Software

Aquesta és la tasca que més temps comportarà i es farà de forma incremental. Primerament es començarà a desenvolupar a banda del projecte SUPERSEDE, és a dir, es crearà un repositori de GitHub on tindran accés el director, el co-director del projecte i el mateix alumne que serà el desenvolupador. Sobre aquest repositori s'anirà evolucionant el software. Durant aquesta fase es realitzaran reunions cada dues setmanes per tal de comentar o polir detalls que els directors creguin oportuns.

6.6 Millora i adaptació a SUPERSEDE

Un cop finalitzat el desenvolupament del software, o potser quan aquest estigui en procés de finalització, començarà la fase d'integració i adaptació al software de SUPERSEDE, és a dir, el software on actualment està el controlador programat en Ruby on Rails, i que serà també el destí del nou controlador programat en Java.

6.7 Preparació de la defensa oral

Aquesta fase es realitzarà un cop acabada la fase de desenvolupament del software, o bé quan aquesta s'estigui acabant. Tindrà una durada aproximada de 3 setmanes i consistirà a unificar tota la documentació generada al llarg del projecte i preparar una presentació oral per a la defensa.

7. Valoració d'alternatives i pla d'acció

7.1 Temps estimat

Fase	Temps de dedicació (hores) .
Analisi i estudi previ del projecte SUPERSEDE	10
Configuració de l'entorn de treball	30
Gestió de projectes	110
Testing	40
Desenvolupament del Software	300
Millora i adaptació a SUPERSEDE	70
Preparació defensa oral	50
TOTAL	**610**

7.2 Pla d'acció

Utilitzar metodologies àgils ens proporciona la facilitat de què encara que apareguin problemes en el transcurs de les fases es pot modificar per poder complir límits. Si una fase, per exemple, acaba unes hores més aviat del que s'havia previst, doncs s'aprofiten aquestes hores per destinar-les a la següent fase. I, al contrari, si s'acaba més tard alguna de les fases, es poden utilitzar algunes hores de la següent fase, ja que en fer l'estimació d'hores també es té en compte això, la possible flexibilitat.

El projecte quedarà acabat en el temps establert i amb la millor adaptació al projecte SUPERSEDE, amb els corresponents testos i validacions.

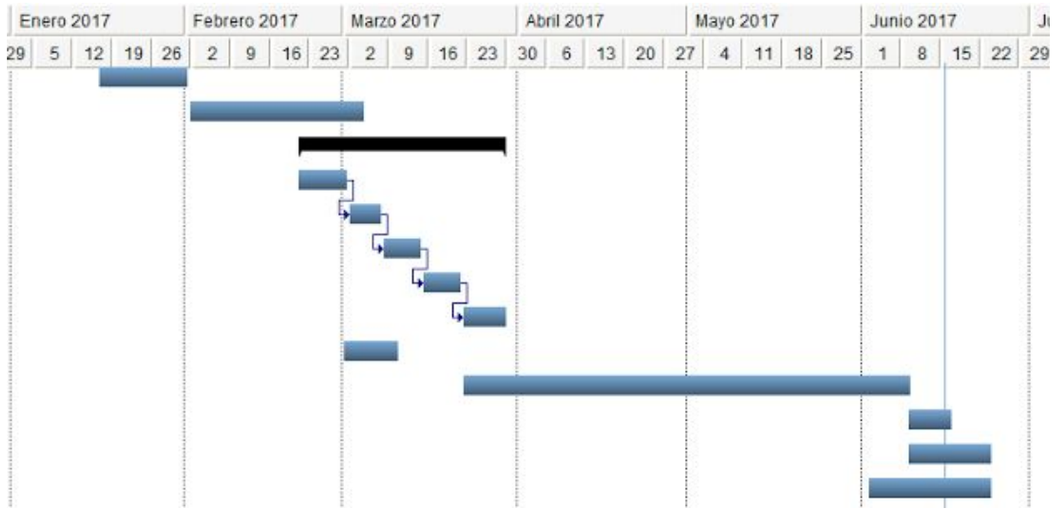
8. Recursos

Per realitzar el projecte s'utilitzaran els següents recursos:

- MacBook Pro Retina
- Entorn de programació: Spring Tool Suite
- Postman per a la realització de testos
- JUnit per a la realització de testos
- Draw.io Diagrams per a la realització dels diagrames UML
- Swagger per a la documentació de l'API utilitzada i per a la generació de codi en Java

9. Diagrama de Gantt

Nombre	Duración	Inicio	Fin	Predecesoras
Analisi i estudi previ del projecte SUPERSEDE	12d?	16/01/2017	31/01/2017	
Configuració de l'entorn de treball	23d?	01/02/2017	03/03/2017	
⊟Gestió de projectes (GEP)	**27d?**	**20/02/2017**	**28/03/2017**	
Abast i contextualització	7d?	20/02/2017	28/02/2017	
Planificació temporal	4d?	01/03/2017	06/03/2017	4
Gestió económica	5d?	07/03/2017	13/03/2017	5
Presentació preliminar	5d?	14/03/2017	20/03/2017	6
Condicions especialitat i presentació prèvia	6d?	21/03/2017	28/03/2017	7
Testing	8d?	28/02/2017	09/03/2017	
Desenvolupament del Software	57d?	21/03/2017	07/06/2017	
Testing	6d?	07/06/2017	14/06/2017	
Millora i adaptació a SUPERSEDE	11d?	07/06/2017	21/06/2017	
Preparació de la defensa oral	16d?	31/05/2017	21/06/2017	



10. Variació de la planificació inicial

10.1 Variacions

A continuació veurem les tasques que hi havia a la planificació inicial, els canvis que hi ha hagut respecte a aquesta.

1. Anàlisi i estudi previ del projecte SUPERSEDE
2. Configuració de l'entorn de treball
3. Gestió de projectes (GEP)
4. Testing
5. Desenvolupament del software
6. Millora i adaptació a SUPERSEDE
7. Preparació de la defensa oral

1. **Canvi 1:** la tasca 2 "Configuració de l'entorn de treball" programada del 01/02/2017 a 03/03/2017 (amb solapament amb la tasca 3 i 4) ha hagut de ser allargada i no va acabar fins a 10/04/2017, i a partir d'aquí es va poder començar a evolucionar el software (tasca 5).

El motiu pel qual la tasca 2 ha hagut de ser allargada no va ser per causa del solapament de les tasques 3 i 4, sinó que amb el IDE Eclipse vàrem tindre molts problemes a l'hora de fer córrer el Controlador al tomcat, fins que vam optar a canviar d'IDE i utilitzar Spring Tool Suite, el qual no ens va donar cap problema des d'un primer moment.

Això fa que la tasca 2 "Configuració de l'entorn de treball" (30 hores) li prengui a la tasca 5 "Desenvolupament del software" (300 hores) al voltant de 40 hores. Per tant la planificació d'aquestes tasques queda de la següent manera:

- Tasca 2 Configuració de l'entorn de treball:
  - Inici: 01/02/2017
  - Final: ~~03/03/2017~~ → 10/04/2017
  - Hores: 30 hores → 70 hores

- Tasca 5 Desenvolupament del software:
  - Inici: ~~21/03/2017~~ → 10/04/2017
  - Final: 07/06/2017
  - Hores: 300 hores → 260 hores

Afectacions: A l'hora de fer la planificació inicial ja es va tindre en compte aquest possible desviament d'hores entre aquestes dues tasques. A la tasca "Desenvolupament del software" es van tindre en compte unes 50 hores de més per tindre un cert coixí de comoditat i, de fet, s'explica explícitament a la planificació inicial, i en el cas d'escurçar les hores es tindria en compte a l'hora de fer el pressupost final (en el cas que afecti).

Aquests canvis estan fets a setmana de 8 de Maig, a la tasca de desenvolupament en un punt força avançat, i és aquest canvi d'escurçament de 40 hores es podia fer sense cap problema.

Pel que fa al pressupost no hi haurà modificacions, ja que les hores totals són les mateixes, i el rol que duu a terme les dues tasques implicades també (Arquitecte).

2. **Canvi 2:** La tasca 4 "Testing" en un principi estava planificada amb començament el 28/02/2027 i finalització 09/03/2017, però finalment no serà així. Es realitzarà abans del desenvolupament del software (tal com està planificat en un principi) i també durant aquest. Tot i així la quantitat d'hores programada és la mateixa, 40.

Això serà així, ja que és s'ha testejat al principi per tindre una idea de com funciona el controlador antic, però també durant el desenvolupament de cadascuna de les operacions, per així saber en cada moment que es vol.

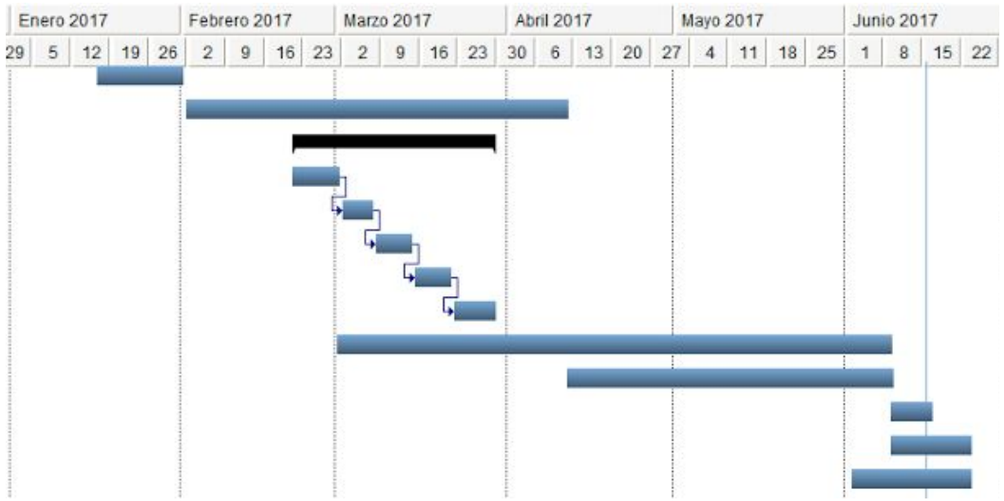
- Tasca 4 Testing:
  - Inici: 28/02/2017
  - Final: ~~09/03/2017~~ → 07/06/2017
  - Hores: 40 hores (les mateixes)

Afectacions: Com ja hem vist l'única afectació és la planificació temporal d'aquesta tasca, però no en la quantitat d'hores totals d'aquesta. Per tant, el pressupost continua sent el mateix.

3. **Canvi 3:** L'IDE utilitzat finalment no va ser Eclipse per problemes tècnics que estan descrits més endavant. L'eina utilitzada finalment va ser Spring Tool Suite. Pel que fa al pressupost no varia, ja que ambdues eines són gratuïtes.

10.2 Variació del Diagrama de Gantt

Nombre	Duración	Inicio	Fin	Predecesoras
Analisi i estudi previ del projecte SUPERSEDE	12d?	16/01/2017	31/01/2017	
Configuració de l'entorn de treball	49d?	01/02/2017	10/04/2017	
Gestió de projectes (GEP)	**27d?**	**20/02/2017**	**28/03/2017**	
Abast i contextualització	7d?	20/02/2017	28/02/2017	
Planificació temporal	4d?	01/03/2017	06/03/2017	4
Gestió económica	5d?	07/03/2017	13/03/2017	5
Presentació preliminar	5d?	14/03/2017	20/03/2017	6
Condicions especialitat i presentació prèvia	6d?	21/03/2017	28/03/2017	7
Testing	72d?	28/02/2017	07/06/2017	
Desenvolupament del Software	43d?	10/04/2017	07/06/2017	
Testing	6d?	07/06/2017	14/06/2017	
Millora i adaptació a SUPERSEDE	11d?	07/06/2017	21/06/2017	
Preparació de la defensa oral	16d?	31/05/2017	21/06/2017	



11. Pressupost

11.1 Estimació dels costos

A continuació detallarem els costos del projecte, els quals estan directament relacionats amb els recursos descrits en l'entrega anterior. Detallarem el hardware i software que hem parlat anteriorment i també parlarem dels recursos humans del projecte i altres despeses relacionades.

11.1.1 Recursos Humans

Aquest projecte està dirigit per una sola persona, és a dir, l'alumne que realitza el TFG, el qual serà l'encarregat de fer de project manager, d'analista, d'arquitecte, i de tester. En el mercat laboral aquests rols no estan pagats de la mateixa manera, per tant, aplicarem sous diferents en funció de les hores realitzades per cadascun dels rols i el seu preu/hora. Buscant i contrastant informació utilitzarem els següents preus en funció de cada rol:

- Project Manager: 25 €/hora[4]
- Analista: 20 €/hora[5]
- Arquitecte 25 €/hora[6]
- Tester 20 €/hora[7]

A continuació, i un cop fets els càlculs, podem veure detalladament els costos humans del projecte:

Fase	Temps de dedicació en hores				Hores/Fase	Cost Estimat (€)
	Project Manager	Analista	Arquitecte	Tester		
Analisi i estudi previ del projecte SUPERSEDE	10	0	0	0	10	250,00
Configuració de l'entorn de treball	0	10	20	0	30	700,00
Gestió de projectes	90	20	0	0	110	2.650,00
Testing	0	0	0	40	40	800,00
Desenvolupament del Software	0	0	280	20	300	7.400,00
Millora i adaptació a SUPERSEDE	0	20	50	0	70	1.650,00
Preparació de la defensa oral	50	0	0	0	50	1.250,00
TOTAL	**150**	**50**	**350**	**60**		**14.700,00**

I detallat per rol:

Rol	Hores previstes	Preu/hora (€)	Cost estimat (€)
Project Manager	150	25	3.750,00
Analista	50	20	1.000,00
Arquitecte	350	25	8.750,00
Tester	60	20	1.200,00
TOTAL	**610**		**14.700,00**

11.1.2 Recursos No Humans

A part dels recursos humans descrits anteriorment, també trobem recursos no humans en dues categories ben diferenciades: hardware i software.

Pel que fa al hardware utilitzat per al projecte tenim un MackBook Pro Retina 13'. Per aquest hardware podem veure la corresponent amortització. Suposant que té una vida útil de 9 anys i utilitzant aquest hardware durant 4 mesos (⅓ d'any) per al projecte llavors hem calculat l'amortització corresponent. Veiem la taula:

HARDWARE				
Producte	Preu	Unitats	Vida útil (anys)	Amortització (€)
MacBook Pro, Retina 13'	1.559,00€	1	9	57,74€
TOTAL	**1.559,00€**	**1**		**57,74€**

Pel que fa al software utilitzarem com a IDE l'Spring Tool Suit, Postman, per realitzar alguns dels testos (encara que la majoria es realitzaran amb JUnit), l'eina Swagger i Workbench[14][15] (per a la base de dades), tots de manera gratuïta.

SOFTWARE				
Producte	Preu	Unitats	Vida útil (anys)	Amortització (€)
Spring Tool Suite	0,00€	1	-	0
Postman	0,00€	1	-	0
Swagger	0,00€	1	-	0
WorkBench	0,00 €	1	-	0
TOTAL	**0,00€**	**1**		**0**

La taula següent mostra el total dels costos de recursos no humans:

TOTAL (SW+HW)	Preu (€)
Hardware	57,74
Software	0
TOTAL	**57,74**

Com podem veure els costos dels recursos no humans són altament baixos en comparació als costos dels recursos humans.

11.1.3 Altres despeses

En aquest projecte, com en molts d'altres, hi han despeses que no formen part de manera explícita en el pressupost, com és per exemple la llum, l'aigua o calefacció que es consumeix durant el transcurs del projecte. En aquest cas aquests tipus de despeses no les tindrem en compte, ja que, la gran part, s'estarien consumint de la mateixa manera que si el projecte no es realitzés, com que la zona de realització del projecte és a casa o al Campus Nord.

Altres despeses com, per exemple, el transport a la universitat (metro) no es tindrà en compte en el pressupost.

No obstant això, el cost del ADSL de casa sí que es tindrà en compte. Aquest cost és de 56€ al mes, per tant, 224€.

11.1.4 Cost total

Un cop calculats tots els costos obtenim el cost total del projecte:

Tipus	Preu (€)
Recursos Humans	14.700,00
Recursos No Humans	57,74
Altres costos	224,00
TOTAL	**14.981,74**

11.2 Viabilitat Econòmica

Tenim en compte el pressupost realitzar anteriorment i que ja disposem del hardware i del software que necessitem per realitzar el projecte, les despeses finals es redueixen a recursos humans. Al ser un projecte de 18 crèdits, les hores establertes en les fases i en el conjunt total del projecte estan dins dels límits calculats. Per tant podem dir que aquest projecte és viable tant en temps com en pressupost.

11.3 Control

Pel que fa a la part de recursos no humans no es podrà realitzar cap tipus de control constant, ja que aquests recursos sortiran de la primera inversió inicial del projecte. Únicament podrem comparar al final del projecte el pressupost inicial amb els costos finals per aquests recursos. Podria ser superior, per exemple, si al llarg del projecte ha calgut algun recurs no plantejat des d'un principi.

Pel que fa als recursos humans sí que es pot dur a terme un control constant. El que es farà és un estudi dels costos al final de cada fase, tenint en compte les hores reals dedicades a cada la fase i comparar-les amb les hores previstes al pressupost. Si la diferència és significativa, s'estudiarà quin o quins han pogut ser els motius, fins al punt de, si escau, corregir les estimacions del pressupost. Es posarà més èmfasi sobretot a la fase d'implementació, que és on es poden notar més la possible diferència d'hores.

12. Sostenibilitat i compromís social

Econòmica

Per saber si un projecte és viable econòmicament, s'ha de realitzar una avaluació de costos, tant de recursos humans com de recursos no humans, el qual es pot veure als apartats anteriors. També s'ha establert un protocol de control per si durant el projecte s'ha d'ajustar el pressupost.

El projecte es podria fer en menys temps si disposéssim de més personal o si es dediquessin més hores al projecte, però, l'alumne que fa aquest projecte, a banda del temps dedicat a aquest, també disposa d'una jornada laboral de 26 hores per l'empresa en la qual treballa. Per tant, en un cas hipotètic en què no es dediquessin hores en l'àmbit laboral, o es dediquessin menys, sí que es podria escurçar el temps establert.

Social

Actualment l'estat espanyol ha passat un fort període de crisi econòmica i, en conseqüència, social. Aquest projecte no té un gran impacte social, ja que és un software petit (en comparació al projecte total) realitzat per a una empresa privada.

De cara als usuaris que utilitzin el projecte SUPERSEDE doncs serà un component més dintre de tot el projecte que ajudarà a la finalitat d'aquest.

Ambiental

El recurs principal que s'utilitzarà en aquest projecte serà un portàtil. El portàtil que s'utilitzarà no està destinat únicament a aquest fi, és a dir, també s'utilitza per a oci i temes personals i universitaris (a banda del TFG), però podríem dir que la major part del temps el seu ús està destinat a aquest projecte.

La documentació del projecte es realitza on-line, per cal dir que l'alumne sovint utilitza paper per apuntar-n'hi coses a importants que es diuen a les reunions amb el director i co-director del projecte, però podríem dir que uns 30 fulls de paper a les reunions i com a esborranys a casa.

Taula de sostenibilitat

S'ha estimat i estudiat quin és l'impacte que tindrà la realització del projecte, i s'ha plantejat que l'impacte sigui el mínim possible. Per dur a terme que l'impacte sigui el mínim possible es reutilitzaran tots els recursos que es pugui, per exemple el paper. En aquest cas s'utilitzarà paper reciclat en la mesura del que es pugui.

	PPP	Vida ùtil	Riscs
Ambiental	2	5	0
Econòmica	7	8	-10
Social	9	15	0
Rang de sostenibilitat	18	27	-10

Total	35

13. Anàlisi de requisits

Un anàlisi de requisits és la descripció d'un sistema de software que es desenvoluparà, que estableix requisits funcionals i no funcionals, conté una descripció detallada dels diferents requisits de programari que ha de complir la solució.

13.1 Requisits funcionals

Aquesta secció presenta els requisits funcionals que ha de complir el sistema. Tots els requisits exposats aquí són essencials. Els requisits van ser especificats perquè sigui fàcil comprovar si el sistema els ofereix o no i si se'ls ofereix adequadament.

Els requisits funcionals es justifiquen amb el *testing* que hem fet al llarg del projecte, que testejan el software i, conseqüentment, els requisits funcionals.

Número	Descripció del requisit
RF1	El software ha de permetre crear, modificar, eliminar i llegir una Feature de la base de dades del sistema.
RF2	El software ha de permetre crear, modificar, eliminar i llegir una Resource de la base de dades del sistema.
RF3	El software ha de permetre crear, modificar, eliminar i llegir una Release de la base de dades del sistema.
RF4	El software ha de permetre crear, modificar, eliminar i llegir una Skill de la base de dades del sistema.
RF5	El software ha de permetre crear, modificar, eliminar i llegir Jobs de la base de dades del sistema.
RF6	El software ha de permetre crear un Plan per a una release específica.
RF7	El software ha de permetre eliminar un Plan per a una release específica.
RF8	El software ha de permetre llegir totes les Features d'un determinat projecte de la base de dades del sistema.
RF9	El software ha de permetre llegir totes les Releases d'un determinat projecte de la base de dades del sistema.

RF10	El software ha de permetre llegir totes les Resources d'un determinat projecte de la base de dades del sistema.
RF11	El software ha de permetre llegir totes les Skills d'un determinat projecte de la base de dades del sistema.
RF12	El software ha de permetre afegir Features a una determinada Release base de dades del sistema.
RF13	El software ha de permetre llegir totes les Releases d'una determinada Feature base de dades del sistema.
RF14	El software ha de permetre llegir tots els Projectes de la base de dades del sistema.
RF15	El software ha de permetre afegir Resources a una determinada Release.
RF16	El software ha de permetre afegir Skills a una determinada Feature.
RF17	El software ha de permetre afegir Skills a una determinada Resource.
RF18	El software ha de permetre afegir dependencies entre Features.
RF19	El software ha de permetre eliminar Features d'una determinada Release.
RF20	El software ha de permetre eliminar Resources d'una determinada Release.
RF21	El software ha de permetre eliminar Skills d'una determinada Feature.
RF22	El software ha de permetre eliminar Skills d'una determinada Resource.
RF23	El software ha de permetre eliminar dependències entre Features.

13.2 Requisits no funcionals

Els requisits no funcionals cobreixen tots els requisits restants que no estan coberts pels requisits funcionals. Ells especifiquen criteris que jutgen el funcionament d'un sistema, en lloc de comportaments específics.

Número	Requisit	Descripció del requisit
RNF1	Seguretat	L'accés a la base de dades està restringit sota contrasenya.
RNF2	Rendiment	El sistema ha de poder suportar certa quantitat d'informació si fos necessari.
RNF3	Portabilitat	El software ha de permetre ser executat en diferents plataformes i/o sistemes operatius.
RNF4	Confiabilitat	La qualitat del producte ha de ser bona i el software ha de ser robust.
RNF5	Correcta migració	La migració que ha experimentat el software ha de ser correcta i exitosa. Ha de ser en Java i sota el sistema d'automatització de compilació Gradle.
RNF6	Testing	S'han d'haver implementat tests a les diverses funcionalitats que té el software.
RNF7	Evolutiu	El programari ha de ser fàcil d'evolucionar en un futur.

Justificacions:

RNF1: L'accés a la base de dades és sota contrasenya i no hi ha manera d'accedir sense aquesta.

RNF2: S'han fet proves amb quantitats d'informació estàndard per aquest software i el rendiment és òptim. Els accessos a la base de dades es minimitzen al màxim.

RNF3: El software no presenta cap problema a l'hora de ser executat en diferents plataformes o sistemes operatius.

RNF4: La qualitat del software s'ha anat comprovant i rectificant a mesura que es desenvolupava. La darrera part del Test Driven Development és la refactorització, la qual comporta revisar la qualitat del codi i deixar un codi net i ben estructurat.

RNF5: El software està desenvolupat en Java i sota el sistema d'automatització de compilació Gradle. El software es comporta tal com es comporta l'antic programari, fins i tot millor que aquest.

RNF6: S'han desenvolupat tests, utilitzant JUnit[18][19], per cada una de les funcionalitats del software.

RNF7: Si en un futur s'han de fer alguns canvis en el software, aquest és fàcil d'evolucionar gràcies a l'estructuració i la qualitat del codi.

14. Especificació

14.1 Model Conceptual

A continuació podem veure el Model Conceptual, que agrupa els principals conceptes gestionats pel Replan Controller.

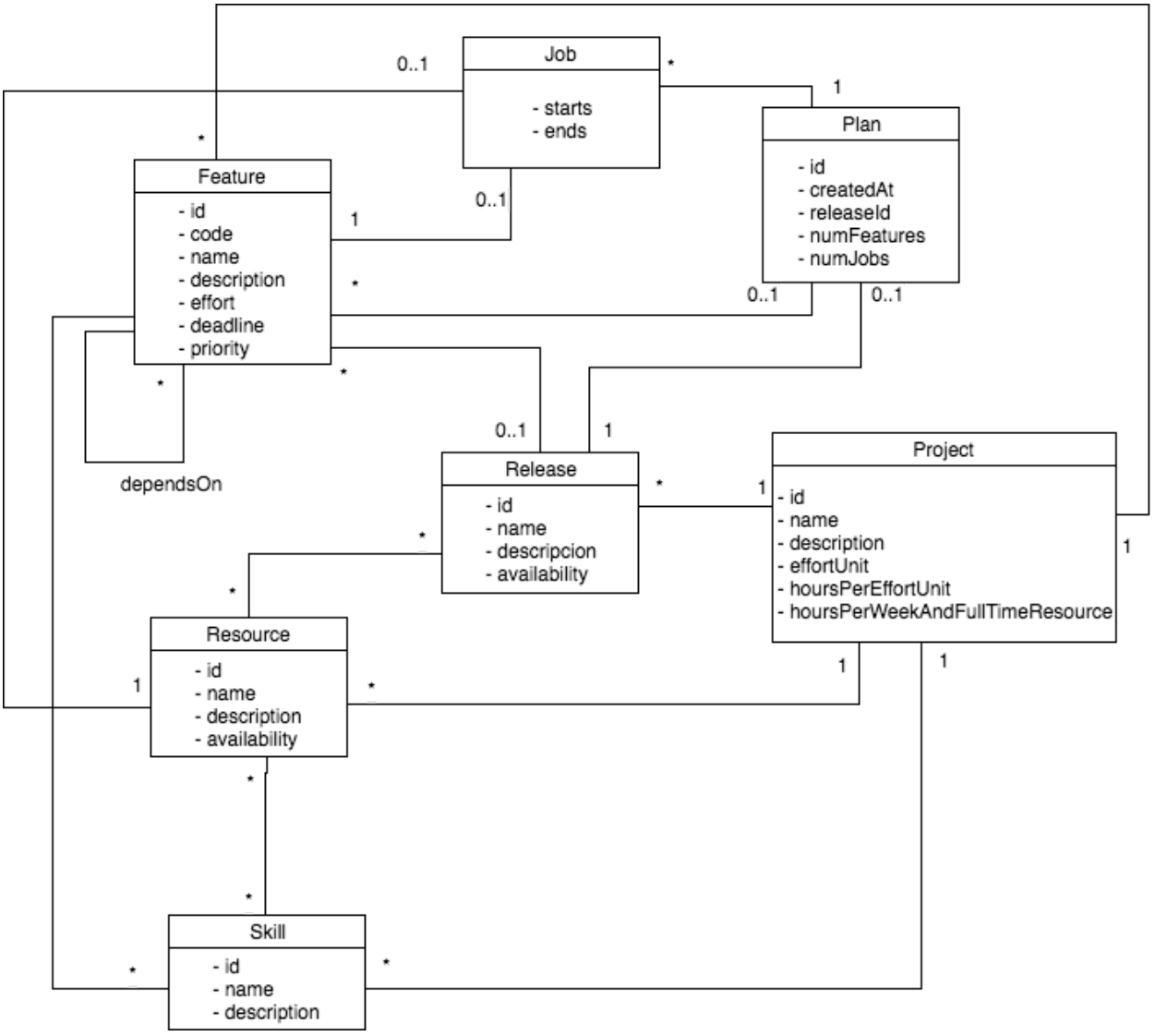
Un dels principals components és el Plan, ja que l'objectiu final del Replan Controller és generar un Plan. A partir de la Release, les Features, les Resources associades i les Skills, el software genera un Plan per Release amb els seus corresponents Jobs.

Cadascun dels Jobs té associades dues dades que indiquen l'inici i l'acabament de dit Job, per exemple, l'hora d'inici podria ser 2016-11-02T09:00:00.000, i l'hora d'acabament 2016-11-03T11:00:00.000Z.

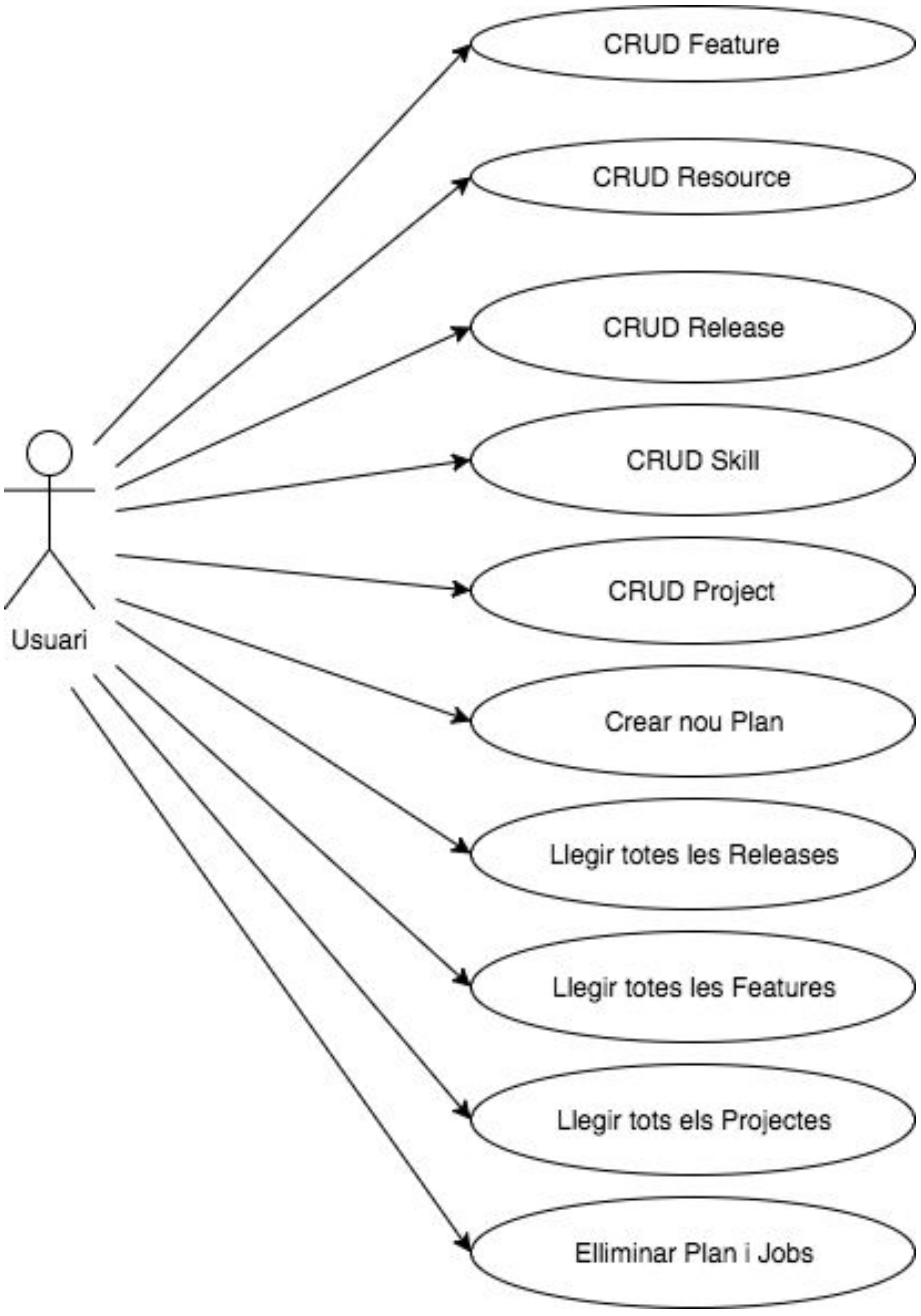
Una Release té associades diferents Features a realitzar al llarg de la Release. Les Features no es poden realitzar en l'ordre que es vulgui, ja que algunes depenen d'altres, això es pot veure en la relació recursiva dependsOn.

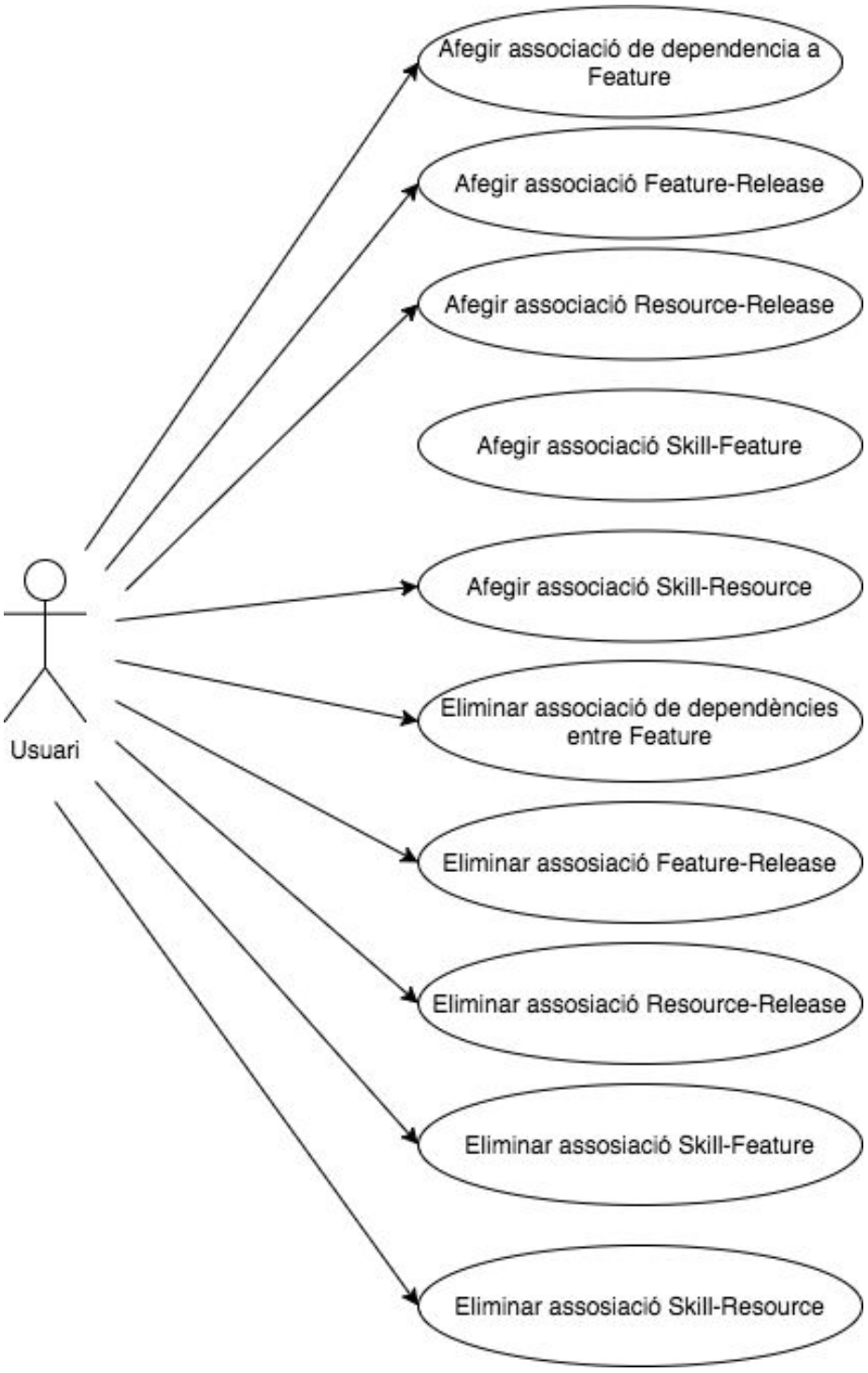
La Release també té associades diferents Resources, és a dir, les persones que podran treballar en aquesta Release. Com a atribut destacable de les Resources tenim l'availability, és a dir, les hores que el treballador disposa a la setmana per treballar.

Les Skills estan relacionades amb les Resources, les Features i els Projects. El Projects (en conseqüència les Releases), ja que cada projecte necessita unes Skills per ser realitzat. Les Resources ja que aquestes Skill les contenen les persones involucrades en el projecte. I la Feature, ja que, igual que el projecte, necessita certes qualitats per dur-la a terme.


Job
- starts
- ends
Plan
- id
- createdAt
- releaseId
- numFeatures
- numJobs
Feature
- id
- code
- name
- description
- effort
- deadline
- priority
dependsOn
Release
- id
- name
- descripcion
- availability
Project
- id
- name
- description
- effortUnit
- hoursPerEffortUnit
- hoursPerWeekAndFullTimeResource
Resource
- id
- name
- description
- availability
Skill
- id
- name
- description
0..1
1
*

14.2 Diagrama de casos d'ús




Afegir associació de dependencia a Feature
Afegir associació Feature-Release
Afegir associació Resource-Release
Afegir associació Skill-Feature
Afegir associació Skill-Resource
Eliminar associació de dependències entre Feature
Eliminar assosiació Feature-Release
Eliminar assosiació Resource-Release
Eliminar assosiació Skill-Feature
Eliminar assosiació Skill-Resource
Usuari

14.3 Operacions del controlador

A continuació es descriuen les operacions del controlador. Per fer-ho especificarem quines són les pre i post condicions, si escau, de cada operació:

1. createProject (body)

· Precondició: body es una estructura no buida.
· Postcondició: Crea un nou projecte amb el contingut de *body.*

2. getProject (projectId)

· Precondició: Existeix un projecte amb id *projectId*.
· Postcondició: Retorna el projecte amb id *projectId.*

3. getProjects ()

· Postcondició: Retorna tots els projectes.

4. modifyProject (projectId, body)

· Precondició: Existeix un projecte amb id *projectId* i body es una estructura Project no buida.
· Postcondició: Modifica el projecte de id *projectId* amb el contingut de *body.*

5. deleteProject (projectId)

· Precondició: Existeix un projecte amb id *projectId.*
· Postcondició: Elimina el projecte amb id *projectId.*

6. createFeature (projectId, body)

· Precondició: Existeix un projecte amb id *projectId* i body es una estructura no buida.
· Postcondició: Crea una nova *feature* al projecte amb id *projectId* amb el contingut de *body.*

7. getFeature (projectId, featureId)

· Precondició: Existeix un projecte amb id *projectId* i una feature amb id *featureId*.
· Postcondició: Retorna la *feature* amb id *featureId* del projecte amb id *projectId.*

8. getFeatures (projectId)

· Precondició: Existeix un projecte amb id *projectId*.
· Postcondició: Retorna les *features* del projecte amb id *projectId.*

9. modifyFeature (projectId, featureId, body)

· Precondició: Existeix un projecte amb id *projectId,* una feature amb id *featureId*, i body es una estructura Feature no buida.
· Postcondició: Modifica la *feature* amb id *featureId*, associada amb el projecte amb id *projectId*, amb el contingut de *body*.

10. deleteFeature (projectId, featureId)

· Precondició: Existeix un projecte amb id *projectId* i una feature amb id *featureId*.
· Postcondició: Elimina la *feature* amb id *featureId* del projecte amb id *projectId.*

11. addNewResourceToProject (projectId, body)

· Precondició: Existeix un projecte amb id *projectId,* i body no es una estructura buida.
· Postcondició: Crea una nova *resource* amb el contingut de *body* al projecte amb id *projectId.*

12. getProjectResources (projectId)

· Precondició: Existeix un projecte amb id *projectId*.
· Postcondició: Retorna les *resources* del projecte amb id *projectId.*

13. modifyResource (projectId, resourceId, body)

· Precondició: Existeix un projecte amb id *projectId,* una resource amb id *resourceId*, i body es una estructura Resource no buida.
· Postcondició: Modifica la *resource* amb id *resourceId*, associada amb el projecte amb id *projectId*, amb el contingut de *body*.

14. deleteResource (projectId, resourceId)

· Precondició: Existeix un projecte amb id *projectId* i una release amb id *releaseId*.
· Postcondició: Elimina la *resource* amb id *resourceId* del projecte amb id *projectId.*

15. addNewReleaseToProject (projectId, body)

· Precondició: Existeix un projecte amb id *projectId,* i body no es una estructura buida.
· Postcondició: Crea una nova *release* amb el contingut de *body* al projecte amb id *projectId.*

16. getRelease (projectId, releaseId)

· Precondició: Existeix un projecte amb id *projectId* i una release amb id *releaseId*.
· Postcondició: Retorna la *release* amb id *releaseId* del projecte amb id *projectId.*

17. getReleases (projectId)

· Precondició: Existeix un projecte amb id *projectId*.
· Postcondició: Retorna les *releases* del projecte amb id *projectId.*

18. modifyRelease (projectId, releaseId, body)

· Precondició: Existeix un projecte amb id *projectId,* una release amb id *releaseId*, i body es una estructura Release no buida.
· Postcondició: Modifica la *release* amb id *releaseId*, associada amb el projecte amb id *projectId*, amb el contingut de *body*.

19. deleteRelease (projectId, releaseId)

· Precondició: Existeix un projecte amb id *projectId* i una release amb id *releaseId*.
· Postcondició: Elimina la *release* amb id *releaseId* del projecte amb id *projectId.*

20. addNewSkillToProject (projectId, body)

· Precondició: Existeix un projecte amb id *projectId,* i body no es una estructura buida.
· Postcondició: Crea una nova *skill* amb el contingut de *body* al projecte amb id *projectId.*

21. getProjectSkills (projectId)

· Precondició: Existeix un projecte amb id *projectId*.
· Postcondició: Retorna les *skills* del projecte amb id *projectId.*

22. modifySkill (projectId, skillId, body)

· Precondició: Existeix un projecte amb id *projectId,* una skill amb id *skillsId*, i body es una estructura Skill no buida.
· Postcondició: Modifica la *skill* amb id *skillId*, associada amb el projecte amb id *projectId*, amb el contingut de *body*.

23. deleteSkill (projectId, skillId)

· Precondició: Existeix un projecte amb id *projectId* i una release amb id *releaseId*.
· Postcondició: Elimina la *skill* amb id *skillId* del projecte amb id *projectId.*

24. addDependenciesToFeature(projectId, featureId, llista<featureId>)

· Precondició: Existeix un projecte amb id *projectId,* una feature amb id *featureId*, i llista es una llista no buida amb estructuras FeatureId.
· Postcondició: Crea les dependències entre les *features* amb ids inclosos a *llista<featureId>* i la *feature* amb id *featureId* i associades al projecte amb id *projectId.*

25. deleteDependenciesFromFeature (projectId, featureId, llista<featureId>)

· Precondició: Existeix un projecte amb id *projectId,* una feature amb id *featureId*, i llista es una llista no buida amb estructuras FeatureId.
· Postcondició: Elimina les dependències entre les *features* amb ids inclosos a *llista<featureId>* i la *feature* amb id *featureId* associada al projecte amb id *projectId.*

26. addFeaturesToRelease (projectId, releaseId, llista<featureId>)

· Precondició: Existeix un projecte amb id *projectId,* una release amb id releaseId, i llista es una llista no buida amb estructuras FeatureId.
· Postcondició: Afegeix les *features* amb ids a *llista<featureId>* a la release amb id *releaseId* i associada amb el projecte amb id *projectId.*

27. getReleaseFeatures (projectId, releaseId)

· Precondició: Existeix un projecte amb id *projectId* i una release amb id *releaseId*.
· Postcondició: Retorna les *features* de la *release* de id *releaseId* asociada al projecte amb id *projectId.*

28. removeFeaturesFromRelease (projectId, releaseId, llista<featureId>)

· Precondició: Existeix un projecte amb id *projectId,* una release amb id *releaseId*, i llista es una llista no buida amb estructuras FeatureId.
· Postcondició: Elimina les relacions entre la *release* amb id *releaseId* associada al projecte amb id *projectId* i les *features* amb ids continguts a *llista<featureId>.*

29. addResourcesToRelease (projectId, releaseId, llista<resourceId>)

· Precondició: Existeix un projecte amb id *projectId,* una release amb id *releaseId*, i llista es una llista no buida amb estructuras ResourceId.
· Postcondició: Crea les relacions entre les *resources* amb ids continguts a *llista<resourceId>* i la release amb id *releaseId* associada al projecte amb id *projectId.*

30. deleteResourcesFromRelease (projectId, releaseId, llista<resourceId>)

· Precondició: Existeix un projecte amb id *projectId,* una release amb id *releaseId*, i llista es una llista no buida amb estructuras ResourceId.
· Postcondició: Elimina les relacions entre les *resources* amb ids inclosos a *llista<resourceId>* i la *release* amb id *releaseId* associada al projecte amb id *projectId.*

31. addSkillsToFeature (projectId, featureId, llista<skillsId>)

· Precondició: Existeix un projecte amb id *projectId,* una feature amb id *featureId*, i llista es una llista no buida amb estructuras SkillId.
· Postcondició: Crea les relacions entre les *skills* amb ids continguts a *llista<skillsId>* i la *feature* amb id *featureId* associada al projecte amb id *projectId.*

32. deleteSkillsFromFeature (projectId, featureId, llista<skillId>)

· Precondició: Existeix un projecte amb id *projectId,* una feature amb id *featureId*, i llista es una llista no buida amb estructuras SkillId.
· Postcondició: Elimina les relacions entre les *skills* amb ids inclosos a *llista<skillId>* i la *feature* amb id *featureId* associada al projecte amb id *projectId.*

33. addSkillsToResource (projectId, resourceId, llista<skillsId>)

· Precondició: Existeix un projecte amb id *projectId,* una resource amb id *resourceId*, i llista es una llista no buida amb estructuras SkillId.
· Postcondició: Crea les relacions entre les *skills* amb ids continguts a *llista<skillsId>* i la *resource* amb id *resourceId* associada al projecte amb id *projectId.*

34. deleteSkillsFromResource (projectId, resourceId, llista<skillId>)

· Precondició: Existeix un projecte amb id *projectId,* una resource amb id *resourceId*, i llista es una llista no buida amb estructuras SkillId.
· Postcondició: Elimina les relacions entre les *skills* amb ids inclosos a *llista<skillId>* i la *resource* amb id *resourceId* associada al projecte amb id *projectId.*

35. cancelLastReleasePlan (projectId, releaseId)

· Precondició: Existeix un projecte amb id *projectId* i una release amb id *releaseId*.
· Postcondició: Elimina el plan de la *release* amb id *releaseId* associada al projecte *projectId* i els jobs associats al plan.

36. getReleasePlan (projectId, releaseId)

· Precondició: Existeix un projecte amb id *projectId* i una release amb id *releaseId*.
· Postcondició: Calcula, retorna i crea el plan per la *release* amb id *releaseId* del projecte amb id *projectId.* Amb el plan calculat també crea els *Jobs* corresponents al plan.

15. Disseny

15.1 Spring Framework

Spring Framework proporciona un model complet de programació i configuració per a aplicacions empresarials modernes basades en Java, en qualsevol tipus de plataforma d'implementació. Un element clau de Spring és el suport d'infraestructures a nivell d'aplicació: Spring se centra en la "plomeria" (*plumbing* en anglès) de les aplicacions empresarials perquè els equips puguin concentrar-se en la lògica empresarial de nivell d'aplicació, sense vincles innecessaris amb entorns de desplegament específics.

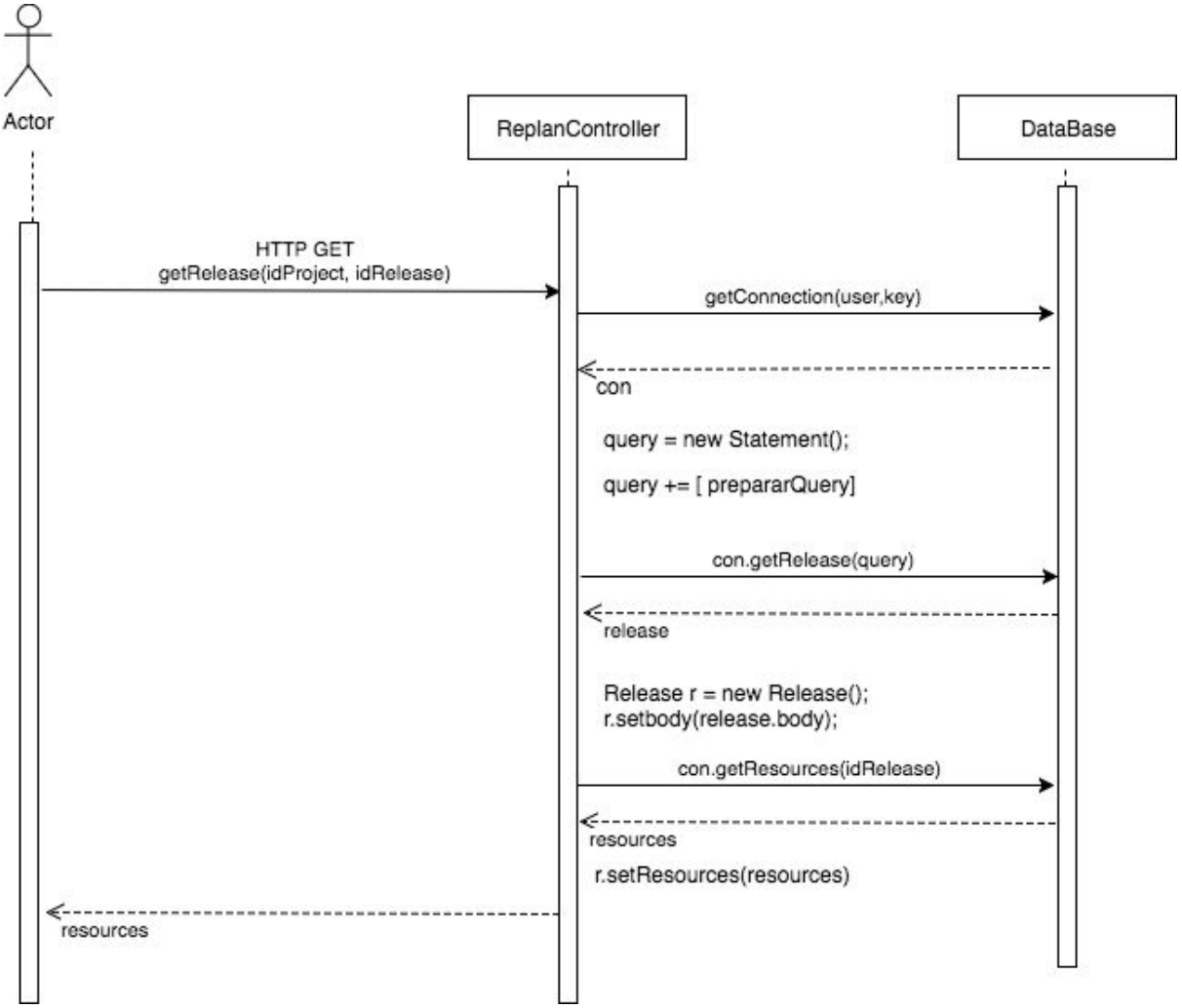
15.2 Model-View-Controller framework

Spring Framework disposa d'un framework d'aplicacions web MVC, que originalment no estava previst. Els desenvolupadors de Spring decidir escriure el seu propi framework web com una reacció al que percebien com el pobre disseny del (llavors) popular framework Jakarta Struts web, així com les deficiències en altres marcs disponibles. En particular, van considerar que hi havia una separació insuficient entre les capes de presentació i sol·licitud de tractament, i entre la capa de maneig de sol·licitud i el model.

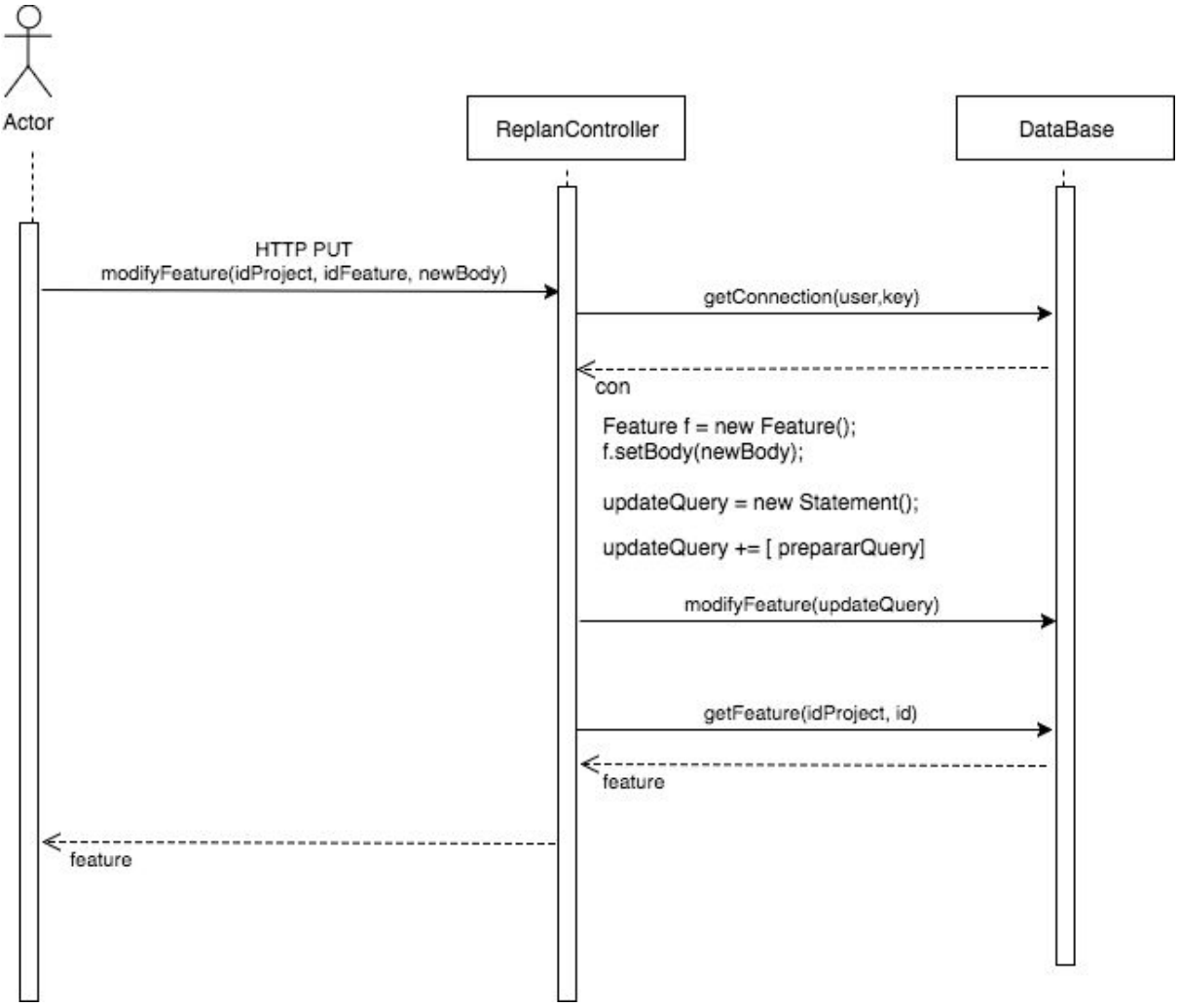
Igual que Struts, Spring MVC és un marc basat en sol·licitud. El marc defineix interfícies d'estratègia per a totes les responsabilitats que han de ser manejades per un marc modern basat en sol·licitud. L'objectiu de cada interfície és ser simple i clar perquè sigui fàcil per als usuaris de Spring MVC escriure les seves pròpies implementacions, si així ho desitgen. MVC obre el camí per a un codi de front end més net. Totes les interfícies estan estretament acoblades a l'API Servlet. Aquest estret acoblament a l'API Servlet és vist per alguns com un fracàs per part dels desenvolupadors de Spring per oferir una abstracció d'alt nivell per a aplicacions basades en web. No obstant això, aquest acoblament assegura que les característiques de l'API de Servlet romanguin disponibles per als desenvolupadors mentre ofereixen un alt marc d'abstracció per facilitar el treball amb aquesta API.

15.3 Diagrames de seqüència

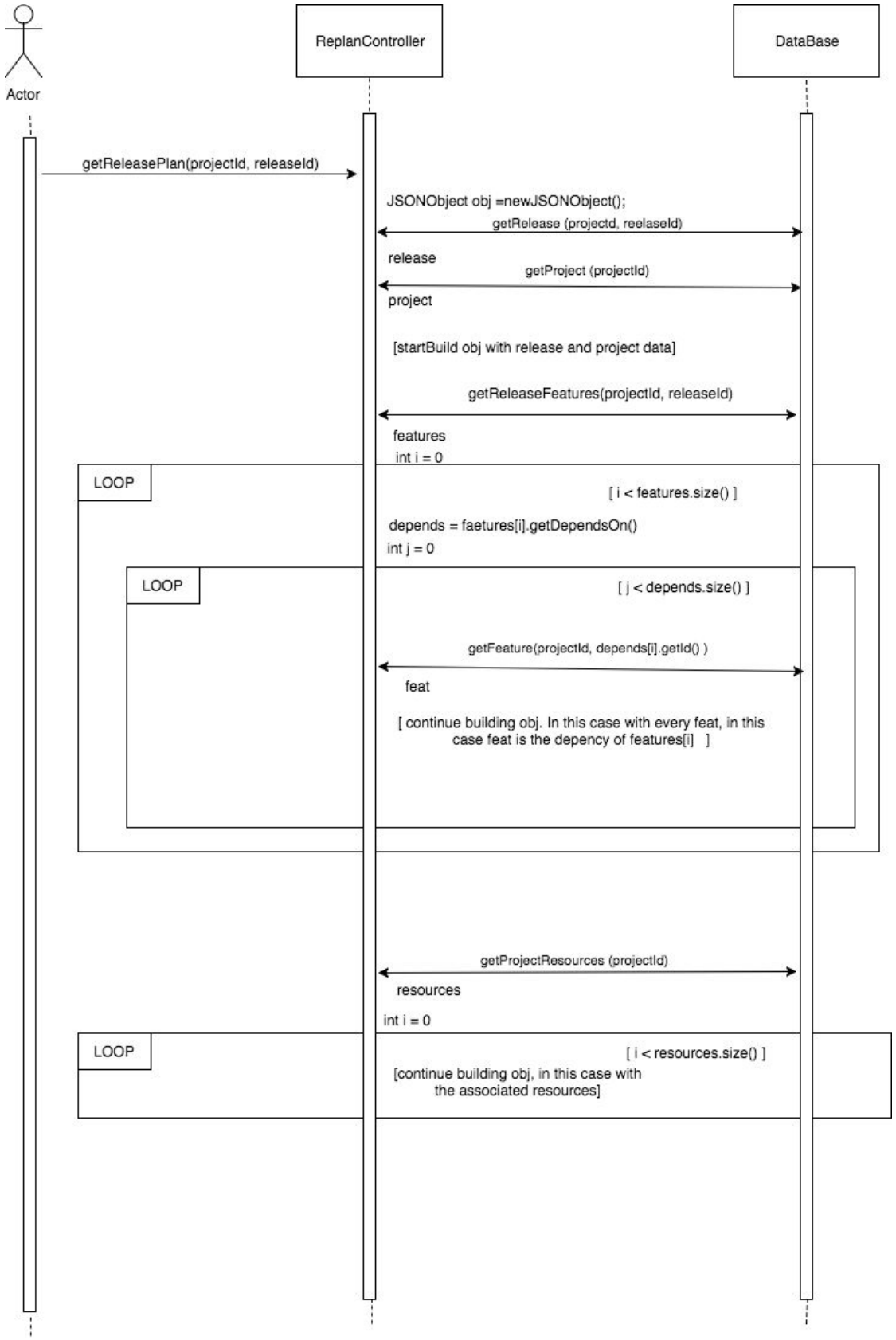
Exemple de GET: getRelease(idProject, idRelease)

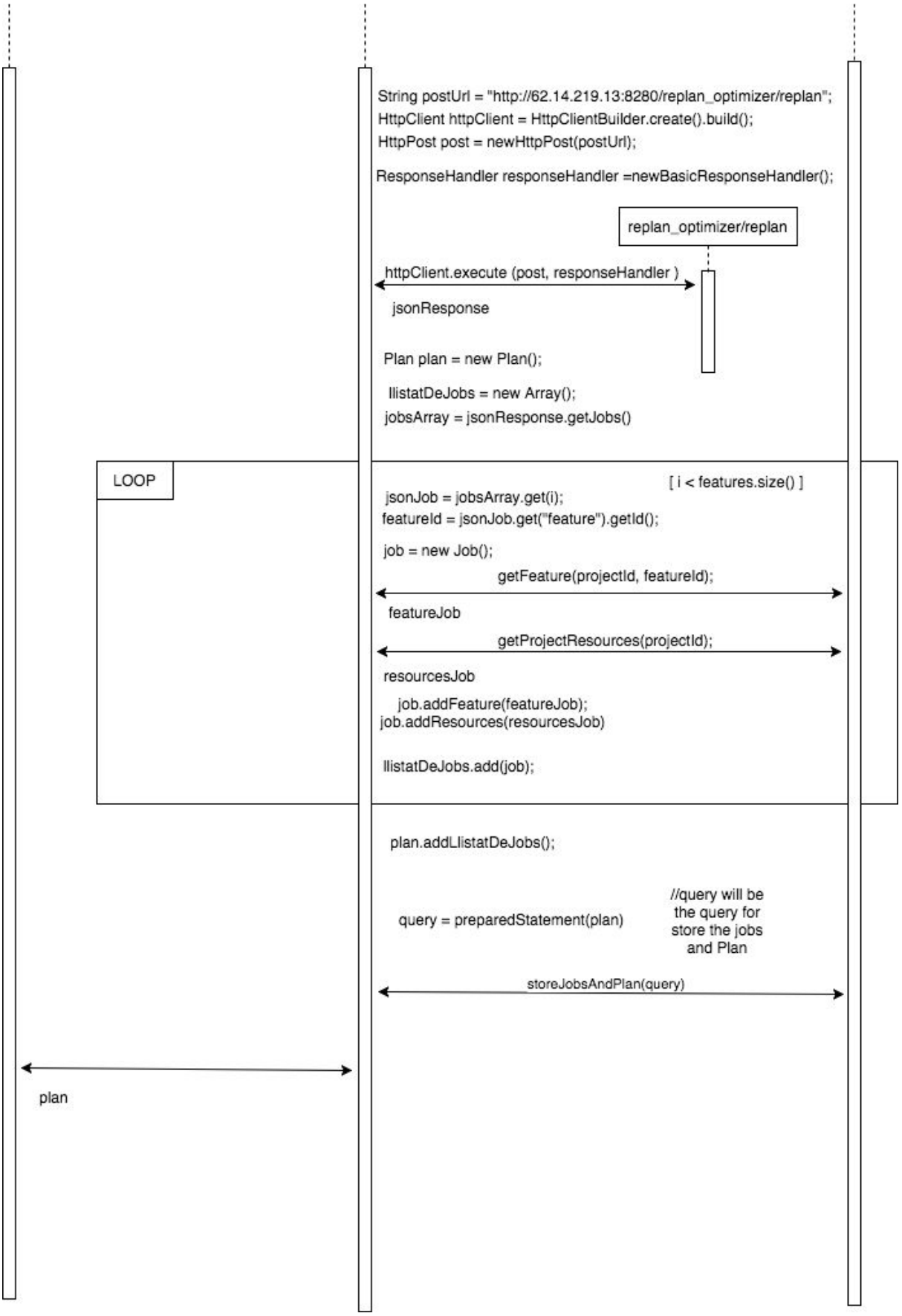


Exemple de PUT: modifyFeature(idProject, idFeature, newBody)



getReleasePlan (idProject, idRelease)




String postUrl = "http://62.14.219.13:8280/replan_optimizer/replan";
HttpClient httpClient = HttpClientBuilder.create().build();
HttpPost post = newHttpPost(postUrl);
ResponseHandler responseHandler =newBasicResponseHandler();
replan_optimizer/replan
httpClient.execute (post, responseHandler)
jsonResponse
Plan plan = new Plan();
llistatDeJobs = new Array();
jobsArray = jsonResponse.getJobs()
LOOP
[i < features.size()]
jsonJob = jobsArray.get(i);
featureId = jsonJob.get("feature").getId();
job = new Job();
getFeature(projectId, featureId);
featureJob
getProjectResources(projectId);
resourcesJob
job.addFeature(featureJob);
job.addResources(resourcesJob)
llistatDeJobs.add(job);
plan.addLlistatDeJobs();
query = preparedStatement(plan)
//query will be the query for store the jobs and Plan
storeJobsAndPlan(query)
plan

15.4 Disseny de la base de dades

A continuació es mostra el llistat de taules, amb els seus corresponents atributs, de la nostra base de dades. Amb les taules es mostren les claus primaries i les claus foranes.

projects (id, name, description, effort_unit, hours_per_effort_unit, hours_per_week_and_full_time_resource)

- primary key (id)

features (id, idProject, code, name, description, effort, deadline, priority)

- primary key (id)
- foreign key clauSkillProject (id_project) references projects (id)

releases (id, id_project, name, description, starts_at, deadline)

- primary key (id)
- foreign key clauReleaseProject (id_project) references projects (id)

resources (id, id_project, name, description, availability)

- primary key (id)
- foreign key clauResourceProject (id_project) references projects (id)

skills (id, id_project, name, description)

- primary key (id)
- foreign key clauSkillProject (id_project) references projects (id)

jobs (id, starts, ends, idResource, idFeature)

- primary key (id)
- foreign key clauJobResource (idResource) references resources (id)
- foreign key clauJobFeature (idFeature) references features (id)

plans (id, createdAt, releaseId, numFeatures, numJobs)

- primary key (id)
- foreign key clauPlanRelease (releaseId) references releases (id)

featureDependencies (idFeature, dependencies)

- primary key (idFeature, dependencies)
- foreign key clauDependencia1 (idFeature) references features (id)
- foreign key clauDependencia2 (dependencies) references features (id)

featureSkill (id_feature, id_skill)

- primary key (id_feature, id_skill)
- foreign key clauFeatureSkill_feature (id_feature) references features (id)
- foreign key clauFeatureSkill_skill (id_skill) references skill (id)

releaseResource (id_release, id_resource)

- primary key (id_release, id_resource)
- foreign key clauReleaseResource_release (id_release) references releases (id)
- foreign key clauReleaseRelease_resource (id_resource) references resources (id)

resourceSkill (id_resource, id_skill)

- primary key (id_resource, id_skill)
- foreign key clauResourceSkill_resource (id_resource) references resources (id)
- foreign key clauRosurcesSkill_skill (id_skill) references skills (id)

planJobs (id_plan, id_job)

- primary key (id_plan, id_job)
- foreign key clauPlanJobs_plan (id_plan) references plans (id)
- foreign key clauPlanJobs_job (id_job) references jobs (id)

16. Implementació

16.1 Tecnologies utilitzades

Spring Tool Suite

Spring Tool Suite és un entorn de desenvolupament basat en Eclipse que es personalitza per a desenvolupar aplicacions Spring. Proporciona un entorn llest per a usar per implementar, depurar, executar i implementar les seves aplicacions de Spring, incloent integracions per Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ i compte amb les ultimes versions d'Eclipse.

Spring Tool Suite suporta l'orientació d'aplicacions a servidors locals, virtuals i basats en el núvol. Està lliurement disponible per al desenvolupament i les operacions de negoci intern s'usen sense límit de temps, totalment open-source i llicenciat sota termes de la llicència pública d'Eclipse.

Gradle

Gradle és un sistema d'automatització de creació de codi obert que es basa en els conceptes d'Apache Ant i Apache Maven i introdueix un llenguatge específic de domini basat en Groovy (DSL) en lloc del formulari XML utilitzat per Apache Maven per declarar la configuració del projecte. Gradle utilitza un graf acíclic dirigit ("DAG") per determinar l'ordre en què es poden executar les tasques.

Gradle va ser dissenyat per a les construccions de projectes múltiples que poden arribar a ser bastant grans, i suporta construccions incrementals mitjançant la determinació intel·ligent de quines parts de l'arbre de construcció estan actualitzades, de manera que qualsevol tasca depenent d'aquestes parts no haurà de ser reexecutat.

Els complements inicials se centren principalment en Java, Groovy i Scala en desenvolupament i desplegament, però més idiomes i fluxos de treball de projectes estan al mapa de ruta.

```
apply plugin: 'java'
```

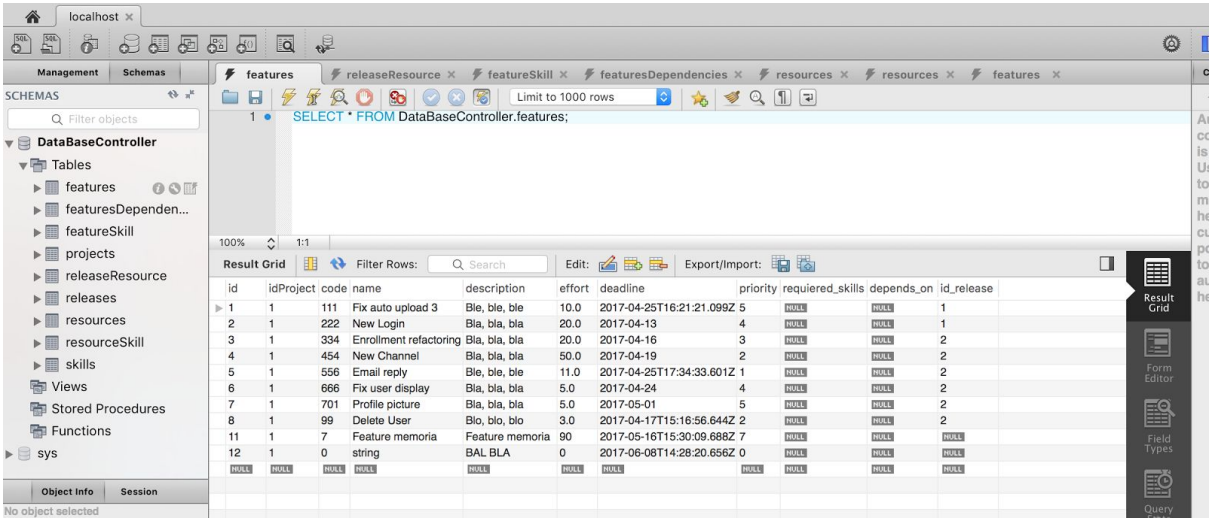
El complement de Java emula molts dels cicles de vida de Maven esperats com tasques en el gràfic acíclic dirigit de dependències per a les entrades i sortides de cada tasca.

Per a projectes que no segueixen les convencions de Maven, Gradle permet configurar l'estructura de directoris. El següent exemple suporta un projecte que conté arxius d'origen en src / java en lloc de la convenció src / main / java forçada per Maven.

MySQL Workbench

MySQL Workbench és una eina de disseny de base de dades visual que integra el desenvolupament de SQL, administració, disseny de base de dades, creació i manteniment en un únic entorn de desenvolupament integrat per al sistema de base de dades MySQL. És el successor de DBDesigner 4 de fabFORCE.net, i reemplaça el paquet anterior del programari, MySQL GUI Eines Bundle.

La imatge d'acontinuació mostra com és el nostre entorn per tindre una idea aproximada:

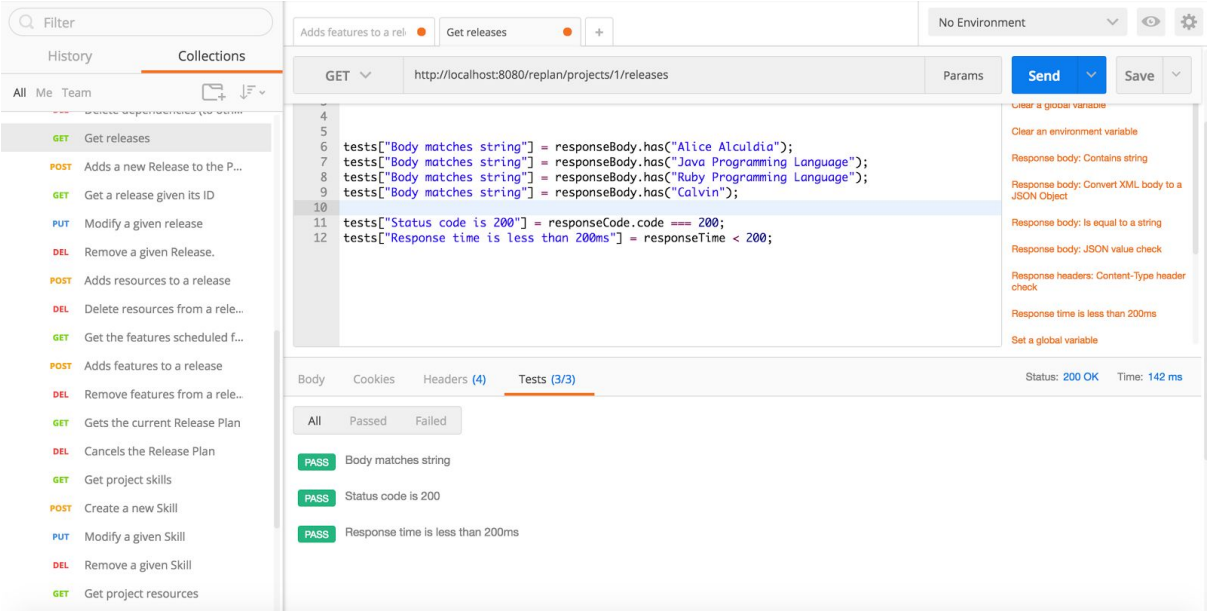


Postman

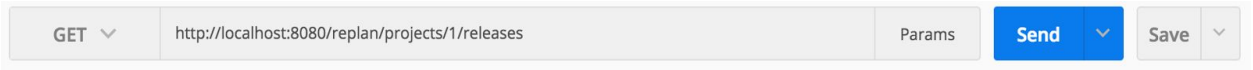
Postman és la cadena d'eines completa per a desenvolupadors d'API, utilitzada per més de 3 milions de desenvolupadors i 30.000 empreses a tot el món. Postman fa que treballar amb API sigui més ràpid i més fàcil al donar suport als desenvolupadors en cada etapa del seu flux de treball, i està disponible per a usuaris de Mac OS X, Windows, Linux i Chrome.

Utilitzarem Postman per fer alguns testos de prova de la API, tant del que ja està desenvolupat en Ruby On Rails, com del nostre a mesura que el desenvolupem. Els testos amb Postman, en comparació amb els testos realitzats amb JUnit, són més pobres, però ens van permetre testejar una mica la informació provinent del software generat en Ruby. A continuació mostrarem algun exemple:

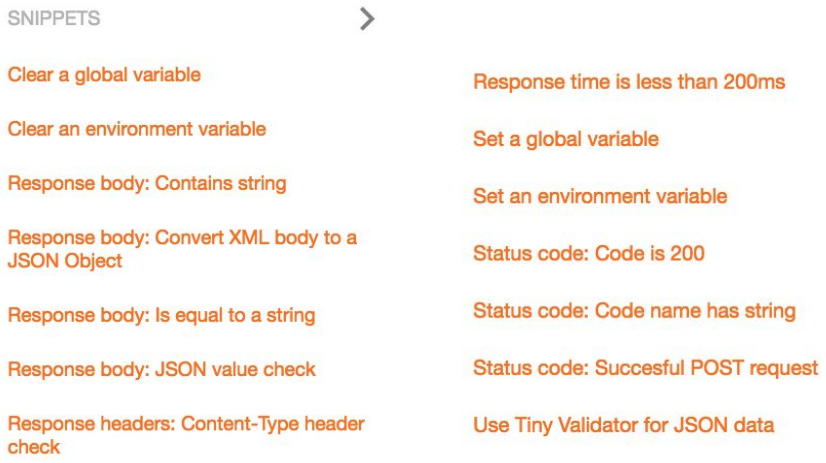
A la següent imatge es mostra l'entorn. A l'esquerre, com podem veure, es mostren totes les operacions de les quals disposem, o bé per testejar o bé per interactuar amb elles.



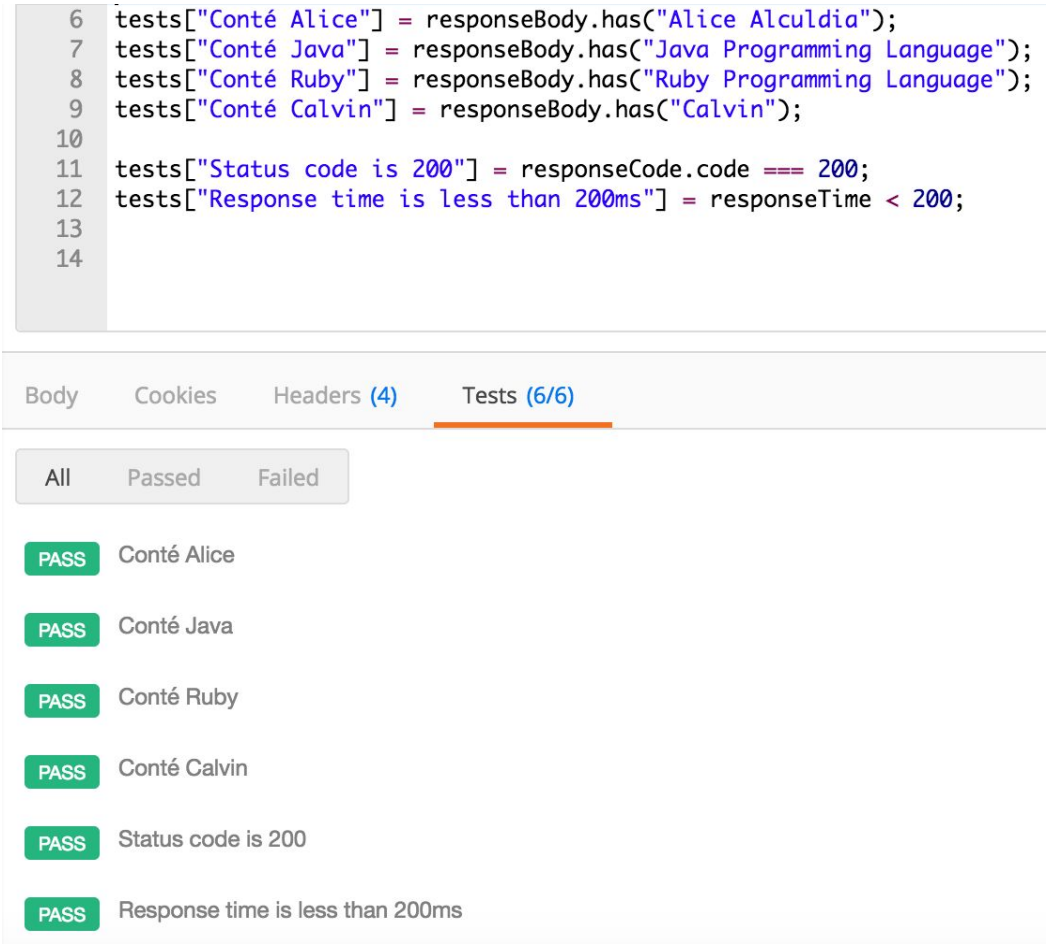
Al capdamunt de la part central disposem un input per escriure la petició desitjada i el tipus de métode.



A la part dreta es mostren els diferents tipus de tests a escollir que ens proporciona Postman per realitzar. Al ser escollits s'escriuen automàticament a la part central on estan escrits els tests, i l'usuari els pot modificar depenent de les seves intencions. Alguns dels tests que ens proporciona Postman són els següents:



I la part central, on queden escrits els tests, i al ser enviada la petició queda constància si els tests són passats o no:



GitHub

GitHub és un servei de *hosting* de repositoris Git, el qual ofereix tota la funcionalitat de Git de control de revisió distribuït i administració de codi de la font (SCM) així com afegint les seves característiques pròpies. A diferència de Git, el qual és estrictament una eina de línia d'ordres, GitHub proporciona una interfície gràfica basada en web i escriptori així com integració del mòbil. També proporciona control d'accés i diverses característiques de col·laboració com *bug tracking*, administració de tasques, i *wikis* per cada projecte.

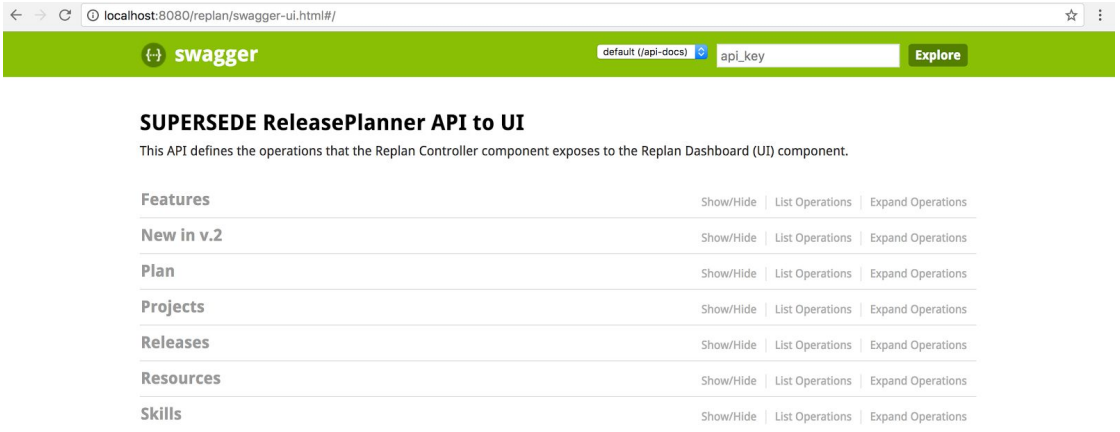
GitHub ofereix ambdós plans; per a repositoris privats i comptes gratuïts, els quals són normalment utilitzats de host en projectes de programari lliure.

En el nostre cas, tindrem un compte de GitHub gratuït i compartir tan amb el director com el co-director del projecte.

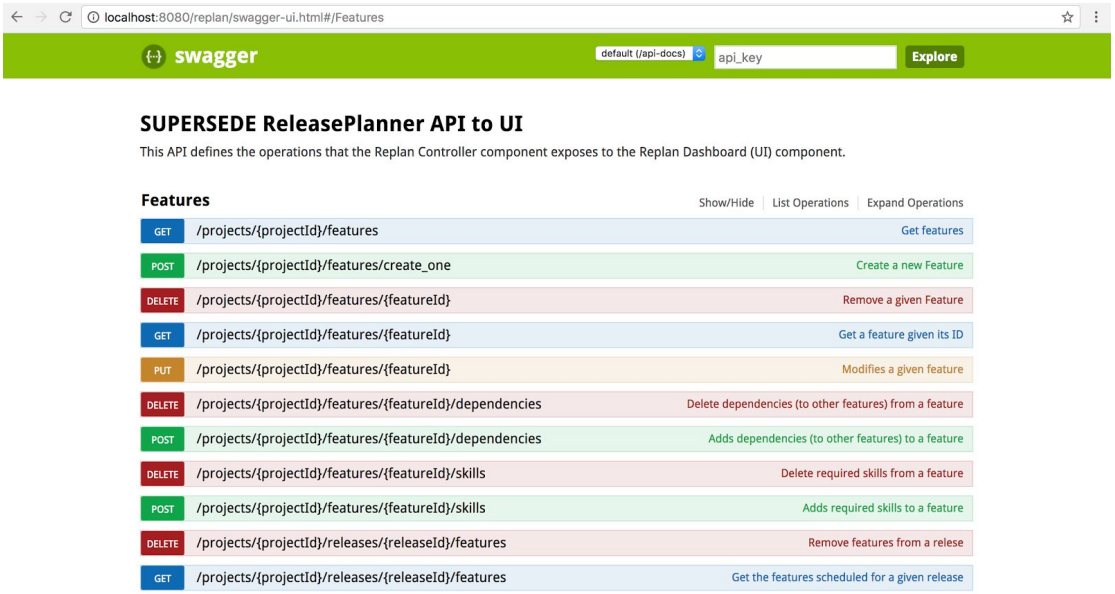
Swagger UI

Swagger UI permet a qualsevol persona - ja sigui el seu equip de desenvolupament o els seus consumidors finals - visualitzar i interactuar amb els recursos de l'API sense tenir cap de la lògica d'implementació al seu lloc. Es genera automàticament a partir de la seva especificació Swagger, amb la documentació visual el que facilita la implementació de back-end i el consum del costat del client.

A continuació tenim una imatge del nostre swagger-UI on podem interactuar amb les operacions de la nostre API, dividides en cadascun dels models, a l'igual que el llistat d'operacions noves de l'última versió.



A la imatge següent, swagger-ui ens mostra amb quines operacions podem interactuar amb les nostres Features. Veiem que tenim *GetFeature, Create a new Feature, Remove a given Feature...* , aquestes operacions estan operatives des del mateix swagger-ui, i podem executar-les.



Aquesta eina es força còmode i ens ha anat molt bé per executar algunes de les operacions de manera ràpida i senzilla.

17. Test Driven Development

Test-driven development (TDD[20]) és una pràctica d'enginyeria de software que involucra altres dues pràctiques: Escriure les proves primer (Test First Development) i Refactorització (Refactoring). En primer lloc, s'escriu una prova i es verifica que les proves fallen. A continuació, s'implementa el codi que fa que la prova passi satisfactòriament i seguidament es refactoriza el codi escrit. El propòsit del desenvolupament guiat per proves és aconseguir un codi net que funcioni. La idea és que els requisits siguin traduïts a proves, d'aquesta manera, quan les proves passin es garantirà que el programari compleix amb els requisits que s'han establert.

En el nostre cas, per dur a terme el TDD hem utilitzat JUnit. JUnit és un conjunt de llibreries que són utilitzades en programació per fer proves unitàries en aplicacions Java. Si es volia desenvolupar l'operació getFeature doncs, primerament es feia el test de manera que no passés (s'escriu una prova i es verifica que la prova falla). En segon lloc, s'implementa el codi de l'operació getFeature perquè la prova passi tenint en compte els requisits. I en tercer lloc es refactoritza el codi, és a dir, modificar el codi pel que fa a l'organització i el disseny.

El TDD ha estat força part del nostre software, tant en temps com en software com a tal. Seguidament explicarem com està estructurat algun dels tests. Hem de tindre presents que al cap i a la fi, un JUnit test no deixa de ser una classe Java.

Dit això, explicarem el test FeatureTest.java, per exemple, i explicarem les parts de la classe:

Part 1: Creem el cos d'una Feature, amb la informació que nosaltres vulguem, i cridem a l'operació que la insereix a la base de dades, en aquest cas createFeature("1", featureData), sent "1" l'id del projecte al qual pertany la Feature i sent featureData el cos de la Feature que hem creat.

Part 2: Fem un getFeature("1", bigFeature), sent "1" el id del projecte al qual pertany la Feature i sent bigFeature el id de la Feature que hem creat, que ho sabem gràcies a la resposta que ens proporciona l'operació cridada a la Part 1. Això ho fem per veure si la Feature que hem creat a la Part 1 ha estat creada amb èxit a la base de dades. Comprovem que la resposta sigui la de la Feature que hem creat. En el cas que no ho sigui el test acaba i no és passat.

Part 3: Creem el cos d'una nova Feature amb la informació diferent de la que hem creat a la Part 1. Cridem a la funció modifyFeature("1", bigFeature, nouCos), on "1" és l'id del projecte al qual pertany la Feature, bigFeature és el seu id, i nouCos és el nou cos de la Feature.

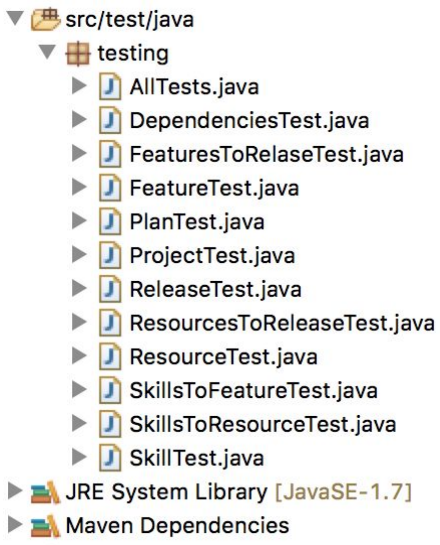
Part 4: Tornem a fer un getFeature("1", bigFeature), sent "1" el id del projecte al qual pertany la Feature i sent bigFeature el id de la Feature. Comprovem que la resposta sigui la de la Feature que hem creat però aquest cop amb el cos de la Feature modificat (amb la informació nova de la Part 3). En el cas que no ho sigui el test acaba i no és passat.

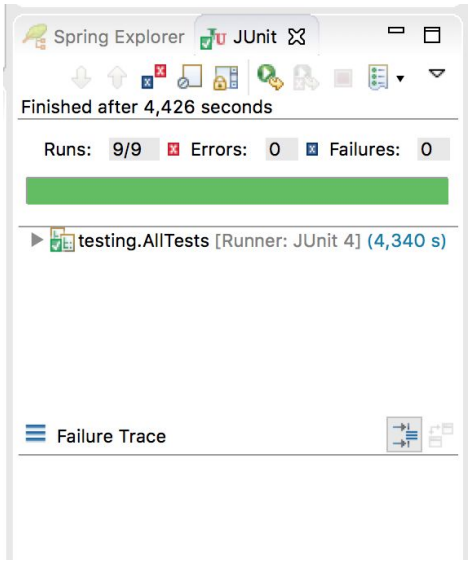
Part 5: Eliminem la Feature. Cridem a l'operació deleteFeature("1", bigFeature). On "1" és l'id del projecte al qual pertany la Feature, i bigFeature l'id de la Feature.

Part 6: Tornem a fer un getFeature("1", bigFeature), sent "1" l'id del projecte al qual pertany la Feature i sent bigFeature el id de la Feature. Comprovem que la resposta ens informa que la Feature no existeix. En el cas que existeixi el test no es passa.

*Totes les operacions cridades són del nostre controlador.

A continuació tenim dues imatges. A l'esquerra, una imatge que ens mostra els tests Junit que tenim al nostre software. Concretament, l'anomenat *AllTests.java* executa tots els tests de la carpeta de *testing*, on tal execució es mostra a la imatge de la dreta.





18. Repositori GitHub

A continuació, es mostra el link del repositori de GitHub que conté el codi font del software:

https://github.com/VictorJaria/ReplanGradleVictorJaria/

Com ja hem dit, aquest repositori és compartit juntament amb els dos directors del projecte, perquè puguin tenir sempre a disposició tots els canvis que es van fent durant el desenvolupament del software.

19. Identificació de lleis i regulacions

Cal fer referència a la llei orgànica de protecció de dades (LOPD[17]), que diu així:

"La Llei Orgànica 15/1999 de 13 de desembre de Protecció de Dades de Caràcter Personal, (LOPD), és una llei orgànica espanyola que té per objecte garantir i protegir, pel que fa al tractament de les dades personals, les llibertats públiques i els drets fonamentals de les persones físiques, i especialment del seu honor, intimitat i privacitat personal i familiar."

Cal tindre en compte aquesta llei, ja que el projecte utilitza informació personal i privada que podria ser força compromesa. Personal ja que hi ha informació de persones (*Resources* és com s'anomenen al llarg del projecte i a la base de dades), potser no és informació gaire compromesa, però s'ha de tindre en compte igualment. També hi ha informació de caràcter més important, estem parlant de dades que fan referència a diferents productes de software que podria ser sensible (*Features, Releases, ...*) i que caldria protegir de tercers.

20. Conclusions

20.1 Competències tècniques

- **CES1.2:** Donar solució a problemes d'integració en funció de les estratègies, dels estàndards i de les tecnologies disponibles. [Bastant]

Al llarg del desenvolupament del software i, en general, de tot el projecte, han sorgit problemes pel que fa a la integració en funció d'estratègies, de desenvolupament i sobretot de tecnologies utilitzades. S'ha sabut buscar solucions als possibles problemes que han sorgit, i adaptar-los en funció de les tecnologies utilitzades, canviar o reorganitzar el calendari si ha calgut i adaptar-nos a les noves solucions.

- **CES1.3:** Identificar, avaluar i gestionar els riscos potencials associats a la construcció de software que es poguessin presentar. [Una mica]

A l'hora de realitzar la planificació temporal del projecte, s'han tingut en compte els possibles riscos potencials que han pogut aparèixer i, que de fet, han aparegut. Quan s'analitzen riscos és important quantificar el nivell d'incertesa i el grau de pèrdues associades a cada risc, per això, quan es va fer la planificació del projecte es va tindre en compte aquest anàlisi dels riscos. De fet, s'ha hagut de modificar la planificació per certs problemes en el desenvolupament, i gràcies a aquest anàlisi de riscos i, en conseqüència, la certa flexibilitat d'aquesta planificació, han facilitat que les modificacions no tinguin conseqüències greus en el projecte.

- **CES1.4:** Desenvolupar, mantenir i avaluar serveis i aplicacions distribuïdes amb suport de xarxa. [En profunditat]

El software desenvolupat i avaluat és un servei distribuït amb suport de xarxa. Accés a la xarxa és un dels requisits necessari perquè el sistema funcioni tal com s'espera.

- **CES1.5:** Especificar, dissenyar, implementar i avaluar bases de dades. [Bastant]

Un dels components principals del projecte és la base de dades. De manera incremental s'ha especificat, dissenyat, implementat, i avaluat una base de dades que compleixi tots els requisits i necessitats del nostre software.

- **CES1.6:** Administrar bases de dades (CIS4.3) [Una mica]

Des del programari desenvolupat al llarg del projecte, es realitzen operacions a una base de dades dinàmica. La informació emmagatzemada es modifica segons les operacions executades, així com operacions d'actualització, esborrat i edició de dades, a més de les operacions fonamentals de consulta.

- **CES1.7:** Controlar la qualitat i dissenyar proves en la producció de software. [En profunditat]

El desenvolupament del software ha estat lligat en tot moment a proves unitàries per testejar cadascuna de les seves funcionalitats. Utilitzant JUnit, un conjunt de llibreries destinades per dur a terme proves de software, s'han dissenyat proves de manera incremental per verificar que la implementació es duia a terme correctament. Primerament es feia la prova que verifica que l'operació o funcionalitat no s'executa correctament (ja que no està implementada). En segon lloc s'implementa el codi per tal que passi la prova tenint en compte els requisits. I en tercer lloc es refactoritza el codi, és a dir, modificar el codi pel que fa a l'organització i el disseny, i és aquí quan es controla la qualitat del software.

- **CES2.1:** Definir i gestionar els requisits d'un sistema software. [Bastant]

Abans de dur a terme el desenvolupament del software s'han hagut de definir els requisits, tant funcionals com no funcionals, gestionar i assegurar que cadascun d'ells el compleixen al final del desenvolupament. Els requisits del software s'han establert juntament amb el director i co-director del projecte, qui tenen contacte amb el client.

20.2 Conclusió personal

Durant el transcurs del projecte, del TFG i GEP, hem tingut èxit en l'anàlisi, el disseny i la implementació del nostre software.

Al començament del projecte vàrem tindre problemes per configurar el Tomcat en el qual el programari estaria corrent, ja que en una de les eines que ens disposàvem a utilitzar teníem problemes, Eclipse. Vam haver de buscar una altra alternativa i la solució va ser canviar d'IDE. A partir d'aquí tot va millorar i vam poder començar a desenvolupar sense problemes, tot i que una mica endarrerits.

Durant el transcurs del desenvolupament no han sorgit gaires problemes, tot ha seguit el transcurs esperat i ha estat un desenvolupament molt continu. Han sorgit dubtes, això sí, però amb l'ajut del director i co-director del projecte s'hi han resolt amb èxit.

En paral·lel s'han desenvolupat les proves unitàries per testejar el software utilitzant la metodologia Test Driven Development. Pel que fa a aquesta part del projecte no hi ha hagut complicacions. Totes les proves s'han realitzat de manera incremental i amb èxit.

D'altra banda, com a conclusions personals, he après a gestionar el temps, a planificar-me en un projecte que duria a terme tot sol, a buscar solucions a problemes greus i no tan greus, i també aspectes tècnics que sempre s'aprenen programant. Al llarg del greu es fan projectes grans, però gairebé sempre són en grup. En aquests projectes, la planificació i la distribució de tasques és més senzilla, ja que els components del grup tenen qualitats diferents, però aquí ha de ser tot organitzat i gestionat per tu mateix, i això és nou i ha estat molt satisfactori.

Annex

SUPERSEDE ReleasePlanner API to UI

This API defines the operations that the Replan Controller component exposes to the Replan Dashboard (UI) component.
More information: https://helloreverb.com
Contact Info: hello@helloreverb.com
Version: 2.0.0
BasePath:/replan

http://apache.org/licenses/LICENSE-2.0.html

Access

Methods

[Jump to Models]

Table of Contents

- POST /projects/{projectId}/releases/{releaseId}/resources
- DELETE /projects/{projectId}/releases/{releaseId}/plan
- DELETE /projects/{projectId}/releases/{releaseId}
- DELETE /projects/{projectId}/releases/{releaseId}/resources
- GET /projects/{projectId}/releases/{releaseId}
- GET /projects/{projectId}/releases/{releaseId}/features
- GET /projects/{projectId}/releases/{releaseId}/plan
- GET /projects/{projectId}/releases
- PUT /projects/{projectId}/releases/{releaseId}
- DELETE /projects/{projectId}/releases/{releaseId}/features

Resources

- POST /projects/{projectId}/resources
- POST /projects/{projectId}/releases/{releaseId}/resources
- POST /projects/{projectId}/resources/{resourceId}/skills
- DELETE /projects/{projectId}/resources/{resourceId}
- DELETE /projects/{projectId}/releases/{releaseId}/resources
- DELETE /projects/{projectId}/resources/{resourceId}/skills
- GET /projects/{projectId}/resources
- PUT /projects/{projectId}/resources/{resourceId}

Skills

- POST /projects/{projectId}/skills
- POST /projects/{projectId}/features/{featureId}/skills
- POST /projects/{projectId}/resources/{resourceId}/skills
- DELETE /projects/{projectId}/skills/{skillId}
- DELETE /projects/{projectId}/features/{featureId}/skills
- DELETE /projects/{projectId}/resources/{resourceId}/skills
- GET /projects/{projectId}/skills
- PUT /projects/{projectId}/skills/{skillId}

Features

Up

POST
/projects/{projectId}/features/{featureId}/dependencies

Adds dependencies (to other features) to a feature (**addDependenciesToFeature**)
This operation allows adding more dependencies on other features by providing a list with the ids of those features

Path parameters

projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")

featureId (required)
Path Parameter — ID of the realese format: int64

Request body

body (required)
Body Parameter — Array of Feature ids

Return type
Feature

Example data
Content-Type: application/json

```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
```

```
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success Feature
404
Project or Feature not found
422
Unprocessable Entity Error

Up

POST /projects/{projectId}/releases/{releaseId}/features

Adds features to a release (**addFeaturesToRelease**)
This operation allows adding more features to a given Release by providing the list of the feature ids.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64
Request body
body (required)
Body Parameter — Array of Feature ids
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project or Release not found
422
Unprocessable Entity Error

Up

POST /projects/{projectId}/features/{featureId}/skills

Adds required skills to a feature (**addSkillsToFeature**)
This operation allows adding more required skills to a given Feature by providing a list of the Skill ids.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
featureId (required)
Path Parameter — ID of the realese format: int64
Request body
body (required)

Body Parameter — Array of Skill ids
Return type
Feature
Example data
Content-Type: application/json

```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success Feature
404
Project or Feature not found
422
Unprocessable Entity Error

Up

POST /projects/{projectId}/features/create_one

Create a new Feature (**createFeature**)
This operation allows defining a new Feature for a given Project.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
Request body
body (required)
Body Parameter — Project parameters
Return type
Feature
Example data
Content-Type: application/json

```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
```

```
  "priority" : ""
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
A new Feature Feature
400
Already exists feature with that 'code' Error
404
Project not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/features/{featureId}/dependencies

Delete dependencies (to other features) from a feature (**deleteDependenciesFromFeature**)
This operation allows deleting dependencies on other features by providing the ids of those features
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
featureId (required)
Path Parameter — ID of the feature format: int64
Query parameters
feature_id (required)
Query Parameter — IDs of the features to remove as dependencies
Return type
Feature
Example data
Content-Type: application/json


```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Success Feature
404

Project or Release not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/features/{featureId}

Remove a given Feature (**deleteFeature**)
This operation allows removing a given Feature.

Path parameters

projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
featureId (required)
Path Parameter — ID of the feature format: int64

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200
Success
404
Project or Feature not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/features/{featureId}/skills

Delete required skills from a feature (**deleteSkillsFromFeature**)
This operation allows deleting required skills from a given Feature by providing the Skill ids.

Path parameters

projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
featureId (required)
Path Parameter — ID of the realese format: int64

Query parameters

skill_id (required)
Query Parameter — IDs of the skills to remove

Return type

Feature

Example data

Content-Type: application/json

```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success Feature
404
Project or Release not found
422
Unprocessable Entity Error

Up
GET /projects/{projectId}/features/{featureId}
Get a feature given its ID (**getFeature**)
This operation returns the main info of a Feature given its featureID.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
featureId (required)
Path Parameter — ID of the feature format: int64
Return type
Feature
Example data
Content-Type: application/json

```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
An array of features Feature
404
Project or Feature not found
422
Unprocessable Entity Error

Up
GET /projects/{projectId}/features
Get features (**getFeatures**)

This operation returns a list of Features for a given Project. This list can be filtered by Feature status: pending or scheduled.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

Query parameters

status (optional)

Query Parameter — any | pending | scheduled

Return type

array[Feature]

Example data

Content-Type: application/json

```
[ {
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
} ]
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

An array of features

404

Project not found

422

Unprocessable Entity Error

Up

GET /projects/{projectId}/releases/{releaseId}/features

Get the features scheduled for a given release (**getReleaseFeatures**)

This operation returns a list of the features scheduled for the given Release.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

releaseId (required)

Path Parameter — ID of the realese format: int64

Return type

array[Feature]

Example data

Content-Type: application/json

```
[ {
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
```

```
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
} ]
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

An array of features

404

Project or Release not found

422

Unprocessable Entity Error

Up

PUT /projects/{projectId}/features/{featureId}

Modifies a given feature (**modifyFeature**)

This operation allows modifying some parameters of a given Feature: name, description, effort, deadline and priority.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

featureId (required)

Path Parameter — ID of the feature format: int64

Request body

body (required)

Body Parameter — Feature parameters that can be modified

Return type

Feature

Example data

Content-Type: application/json


```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success Feature
404
Project or Feature not found
422
Unprocessable Entity Error

Up
DELETE
/projects/{projectId}/releases/{releaseId}/features
Remove features from a relese (**removeFeaturesFromRelease**)
This operation allows removing features from a given Release by providing the Feature ids.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64
Query parameters
featureId (required)
Query Parameter — IDs of the features to remove
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project|Release|Feauture not found
422
Unprocessable Entity Error

NewInV2

Up
POST /projects/{projectId}/features/create_one
Create a new Feature (**createFeature**)
This operation allows defining a new Feature for a given Project.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
Request body
body (required)
Body Parameter — Project parameters
Return type
Feature
Example data
Content-Type: application/json
```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
```

```
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

A new Feature Feature

400

Already exists feature with that 'code' Error

404

Project not found

422

Unprocessable Entity Error

Up

POST /projects

Create a new Project (**createProject**)

This operation allows defining a new Project.

Request body

body (required)

Body Parameter — Project parameters

Return type

Project

Example data

Content-Type: application/json


```
{
  "hours_per_effort_unit" :
1.3579000000000001069366817318950779736042022705078125,
  "name" : "aeiou",
  "description" : "aeiou",
  "effort_unit" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "hours_per_week_and_full_time_resource" :
1.3579000000000001069366817318950779736042022705078125
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
A new Project Project
404
Project not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/features/{featureId}
Remove a given Feature (**deleteFeature**)
This operation allows removing a given Feature.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
featureId (required)
Path Parameter — ID of the feature format: int64
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project or Feature not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}
Remove a given Project (**deleteProject**)
This operation allows removing a given Project.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project not found
422
Unprocessable Entity Error

Up

GET /projects
Get projects (**getProjects**)
This operation retrieves a list of projects.
Return type
array[Project]

Example data
Content-Type: application/json

```
[ {
  "hours_per_effort_unit" :
1.357900000000000106936681731895077973604202270507812 5,
  "name" : "aeiou",
  "description" : "aeiou",
  "effort_unit" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.357900000000000106936681731895077973604202270507812 5
  } ],
  "id" : "",
  "hours_per_week_and_full_time_resource" :
1.357900000000000106936681731895077973604202270507812 5
} ]
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
An array of projects
404
Project not found
422
Unprocessable Entity Error

Plan

Up
`DELETE /projects/{projectId}/releases/{releaseId}/plan`
Cancels the Release Plan (**cancelLastReleasePlan**)
This operation cancels the Release Plan of the given Release.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project or Release not found
422

Unprocessable Entity Error

Up

GET /projects/{projectId}/releases/{releaseId}/plan

Gets the current Release Plan (**getReleasePlan**)
This operation calculates and then returns a new Release Plan for a given Release.

Path parameters

projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64

Return type

Plan

Example data

Content-Type: application/json


```
{
  "release_id" : "",
  "jobs" : [ {
    "depends_on" : [ "" ],
    "feature" : {
      "code" : "",
      "depends_on" : [ "" ],
      "required_skills" : [ "" ],
      "name" : "aeiou",
      "description" : "aeiou",
      "effort" : 1.3579000000000001069366817318950779736042022705078125,
      "id" : "",
      "deadline" : "2000-01-23T04:56:07.000+00:00",
      "priority" : ""
    },
    "resource" : {
      "skills" : [ {
        "name" : "aeiou",
        "description" : "aeiou",
        "id" : ""
      } ],
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : "",
      "availability" : 1.3579000000000001069366817318950779736042022705078125
    },
    "ends" : "2000-01-23T04:56:07.000+00:00",
    "starts" : "2000-01-23T04:56:07.000+00:00"
  } ],
  "created_at" : "2000-01-23T04:56:07.000+00:00",
  "id" : ""
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200
Success Plan
404
Project or Release not found
422
Unprocessable Entity Error

Projects

Up

POST /projects

Create a new Project (**createProject**)

This operation allows defining a new Project.

Request body

body (required)

Body Parameter — Project parameters

Return type

Project

Example data

Content-Type: application/json


```
{
  "hours_per_effort_unit" :
1.3579000000000001069366817318950779736042022705078125,
  "name" : "aeiou",
  "description" : "aeiou",
  "effort_unit" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "hours_per_week_and_full_time_resource" :
1.3579000000000001069366817318950779736042022705078125
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

A new Project Project

404

Project not found

422

Unprocessable Entity Error

Up

DELETE /projects/{projectId}

Remove a given Project (**deleteProject**)

This operation allows removing a given Project.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Success
404
Project not found
422
Unprocessable Entity Error

Up

GET /projects/{projectId}

Get Project info (**getProject**)
This operation returns the main info for a given Project: name, resources, effort unit, etc.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
Return type
Project
Example data
Content-Type: application/json


```
{
  "hours_per_effort_unit" :
1.3579000000000001069366817318950779736042022705078125,
  "name" : "aeiou",
  "description" : "aeiou",
  "effort_unit" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "hours_per_week_and_full_time_resource" :
1.3579000000000001069366817318950779736042022705078125
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Project info Project
404
Project not found
422
Unprocessable Entity Error

Up

GET /projects

Get projects (**getProjects**)
This operation retrieves a list of projects.

Return type
array[Project]
Example data
Content-Type: application/json

```
[ {
  "hours_per_effort_unit" :
1.3579000000000001069366817318950779736042022705078125,
  "name" : "aeiou",
  "description" : "aeiou",
  "effort_unit" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "hours_per_week_and_full_time_resource" :
1.3579000000000001069366817318950779736042022705078125
} ]
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
An array of projects
404
Project not found
422
Unprocessable Entity Error

Up
PUT /projects/{projectId}
Modify Project (**modifyProject**)
This operation allows modyfing some parameters of a given Project: effort unit, number of hours per effor unit and number of hours per week for a full time resource.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
Request body
body (required)
Body Parameter — Project's parameters that can be modified
Return type
Project
Example data
Content-Type: application/json

```
{
  "hours_per_effort_unit" :
1.3579000000000001069366817318950779736042022705078125,
  "name" : "aeiou",
  "description" : "aeiou",
  "effort_unit" : "aeiou",
  "resources" : [ {
```

```
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "hours_per_week_and_full_time_resource" :
1.3579000000000001069366817318950779736042022705078125
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success Project
404
Project not found
422
Unprocessable Entity Error

Releases

Up

`POST /projects/{projectId}/releases/{releaseId}/features`

Adds features to a release (**addFeaturesToRelease**)
This operation allows adding more features to a given Release by providing the list of the feature ids.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64
Request body
body (required)
Body Parameter — Array of Feature ids
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project or Release not found
422
Unprocessable Entity Error

Up

POST /projects/{projectId}/releases

Adds a new Release to the Project (**addNewReleaseToProject**)
This operation allows adding a new Release to a given Project by providing the following parameters: name, description and deadline

Path parameters

projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")

Request body

body (required)
Body Parameter — Parameters of the new Release

Return type

Release

Example data

Content-Type: application/json

```
{
  "starts_at" : "2000-01-23T04:56:07.000+00:00",
  "name" : "aeiou",
  "description" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200
A new release Release
404
Project not found
422
Unprocessable Entity Error

Up

POST /projects/{projectId}/releases/{releaseId}/resources

Adds resources to a release (**addResourcesToRelease**)
This operation allows assigning more resources to a given Release by providing a list with the Resource ids.

Path parameters

projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64

Request body

body (required)
Body Parameter — Array of Resource ids

Return type

Release

Example data

Content-Type: application/json


```
{
  "starts_at" : "2000-01-23T04:56:07.000+00:00",
  "name" : "aeiou",
  "description" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00"
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

Success Release

404

Project or Release not found

422

Unprocessable Entity Error

Up

DELETE /projects/{projectId}/releases/{releaseId}/plan

Cancels the Release Plan (**cancelLastReleasePlan**)

This operation cancels the Release Plan of the given Release.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

releaseId (required)

Path Parameter — ID of the realese format: int64

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

Success

404

Project or Release not found

422

Unprocessable Entity Error

Up

DELETE /projects/{projectId}/releases/{releaseId}

Remove a given Release. (**deleteRelease**)
This operation allows removing a given Release from the Project.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the project format: int64
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Succes
404
Project or Release not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/releases/{releaseId}/resources

Delete resources from a release (**deleteResourcesFromRelease**)
This operation allows deallocating resources from a given Release by providing the ids of those resources.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64
Query parameters
ResourceId (required)
Query Parameter — IDs of the resources to remove
Return type
Release
Example data
Content-Type: application/json

```
{
  "starts_at" : "2000-01-23T04:56:07.000+00:00",
  "name" : "aeiou",
  "description" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Success Release
404
Project or Release not found
422
Unprocessable Entity Error

Up
GET /projects/{projectId}/releases/{releaseId}
Get a release given its ID (**getRelease**)
This operation returns the main info of a Release given its releaseID.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the release format: int64
Return type
Release
Example data
Content-Type: application/json

```
{
  "starts_at" : "2000-01-23T04:56:07.000+00:00",
  "name" : "aeiou",
  "description" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
A Release Release
404
Project or Release not found
422
Unprocessable Entity Error

Up
GET /projects/{projectId}/releases/{releaseId}/features
Get the features scheduled for a given release (**getReleaseFeatures**)

This operation returns a list of the features scheduled for the given Release.

Path parameters

projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64

Return type

array[Feature]

Example data

Content-Type: application/json

```
[ {
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
} ]
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200
An array of features
404
Project or Release not found
422
Unprocessable Entity Error

Up

GET /projects/{projectId}/releases/{releaseId}/plan

Gets the current Release Plan (**getReleasePlan**)

This operation calculates and then returns a new Release Plan for a given Release.

Path parameters

projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64

Return type

Plan

Example data

Content-Type: application/json

```
{
  "release_id" : "",
  "jobs" : [ {
    "depends_on" : [ "" ],
    "feature" : {
      "code" : "",
      "depends_on" : [ "" ],
      "required_skills" : [ "" ],
      "name" : "aeiou",
```

```
      "description" : "aeiou",
      "effort" : 1.3579000000000001069366817318950779736042022705078125,
      "id" : "",
      "deadline" : "2000-01-23T04:56:07.000+00:00",
      "priority" : ""
    },
    "resource" : {
      "skills" : [ {
        "name" : "aeiou",
        "description" : "aeiou",
        "id" : ""
      } ],
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : "",
      "availability" : 1.3579000000000001069366817318950779736042022705078125
    },
    "ends" : "2000-01-23T04:56:07.000+00:00",
    "starts" : "2000-01-23T04:56:07.000+00:00"
  } ],
  "created_at" : "2000-01-23T04:56:07.000+00:00",
  "id" : ""
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

Success Plan

404

Project or Release not found

422

Unprocessable Entity Error

Up

GET /projects/{projectId}/releases

Get releases (**getReleases**)

This operation returns a list with the Releases in a given Project.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

Return type

array[Release]

Example data

Content-Type: application/json


```
[ {
  "starts_at" : "2000-01-23T04:56:07.000+00:00",
  "name" : "aeiou",
  "description" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
```

```
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00"
} ]
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

An array of releases

404

Project not found

422

Unprocessable Entity Error

Up

PUT /projects/{projectId}/releases/{releaseId}

Modify a given release (**modifyRelease**)

This operation allows modifying a given Release providing new values for the following parameters: name, description and deadline

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

releaseId (required)

Path Parameter — ID of the project format: int64

Request body

body (required)

Body Parameter — Release parameters that can be modified

Return type

Release

Example data

Content-Type: application/json


```
{
  "starts_at" : "2000-01-23T04:56:07.000+00:00",
  "name" : "aeiou",
  "description" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00"
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200
Success Release
404
Project or Release not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/releases/{releaseId}/features

Remove features from a relese (**removeFeaturesFromRelease**)
This operation allows removing features from a given Release by providing the Feature ids.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64
Query parameters
featureId (required)
Query Parameter — IDs of the features to remove
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project|Release|Feauture not found
422
Unprocessable Entity Error

Resources

Up

POST /projects/{projectId}/resources

Add a new resource to the project (**addNewResourceToProject**)
This operation allows defining a new Resource to be allocated to the given Project.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
Request body
body (required)
Body Parameter — Resource parameters
Return type
Resource
Example data
Content-Type: application/json

```
{
  "skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
```

```
  "id" : "",
  "availability" : 1.3579000000000001069366817318950779736042022705078125
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

A new Resource Resource

404

Project not found

422

Unprocessable Entity Error

Up

POST /projects/{projectId}/releases/{releaseId}/resources

Adds resources to a release (**addResourcesToRelease**)

This operation allows assigning more resources to a given Release by providing a list with the Resource ids.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

releaseId (required)

Path Parameter — ID of the realese format: int64

Request body

body (required)

Body Parameter — Array of Resource ids

Return type

Release

Example data

Content-Type: application/json


```
{
  "starts_at" : "2000-01-23T04:56:07.000+00:00",
  "name" : "aeiou",
  "description" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00"
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Success Release
404
Project or Release not found
422
Unprocessable Entity Error

Up

POST /projects/{projectId}/resources/{resourceId}/skills

Add skills to a resource (**addSkillsToResource**)
This operation allows assigning more skills to a given Resource by providing a list of the Skill ids.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
resourceId (required)
Path Parameter — ID of the resource format: int64
Request body
body (required)
Body Parameter — Array of Skill ids
Return type
Resource
Example data
Content-Type: application/json

```
{
  "skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "id" : "",
  "availability" : 1.3579000000000001069366817318950779736042022705078125
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Success Resource
404
Project or Resource not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/resources/{resourceId}

Remove a given Resource (**deleteResource**)
This operation removes and deallocates a given Resource from the Project.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
resourceId (required)
Path Parameter — ID of the resource format: int64
Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Success
404
Project or Resource not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/releases/{releaseId}/resources

Delete resources from a release (**deleteResourcesFromRelease**)
This operation allows deallocating resources from a given Release by providing the ids of those resources.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
releaseId (required)
Path Parameter — ID of the realese format: int64
Query parameters
ResourceId (required)
Query Parameter — IDs of the resources to remove
Return type
Release
Example data
Content-Type: application/json

```
{
  "starts_at" : "2000-01-23T04:56:07.000+00:00",
  "name" : "aeiou",
  "description" : "aeiou",
  "resources" : [ {
    "skills" : [ {
      "name" : "aeiou",
      "description" : "aeiou",
      "id" : ""
    } ],
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : "",
    "availability" : 1.3579000000000001069366817318950779736042022705078125
  } ],
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Success Release
404
Project or Release not found
422
Unprocessable Entity Error

[Up](#)

DELETE /projects/{projectId}/resources/{resourceId}/skills

Remove skills from a resource (**deleteSkillsFromResource**)

This operation allows removing skills from a given Resource by providing the Skill ids.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

resourceId (required)

Path Parameter — ID of the resource format: int64

Query parameters

skillId (required)

Query Parameter — IDs of the skills to remove

Return type

Resource

Example data

Content-Type: application/json

```
{
  "skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "id" : "",
  "availability" : 1.3579000000000001069366817318950779736042022705078125
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Success Resource

404

Project or Resource not found

422

Unprocessable Entity Error

[Up](#)

GET /projects/{projectId}/resources

Get project resources (**getProjectResources**)

This operation returns a list of the resources allocated to a given Project.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

Return type

array[Resource]

Example data

Content-Type: application/json

```
[ {
  "skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
```

```
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "id" : "",
  "availability" : 1.3579000000000001069366817318950779736042022705078125
} ]
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

An array of resources

404

Project not found

422

Unprocessable Entity Error

Up

PUT /projects/{projectId}/resources/{resourceId}

Modify a given Resource (**modifyResource**)

This operation allows modifying the parameters of a given Resource, namely name, description, and availability.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

resourceId (required)

Path Parameter — ID of the resource format: int64

Request body

body (required)

Body Parameter — Resource parameters that can be modified

Return type

Resource

Example data

Content-Type: application/json


```
{
  "skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "id" : "",
  "availability" : 1.3579000000000001069366817318950779736042022705078125
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

Success Resource

404

Project or Resource not found

422

Unprocessable Entity Error

Skills

Up

POST /projects/{projectId}/skills

Create a new Skill (**addNewSkillToProject**)

This operation allows defining a new Skill in the context of the given Project.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

Request body

body (required)

Body Parameter — Skill parameters

Return type

Skill

Example data

Content-Type: application/json

```
{
  "name" : "aeiou",
  "description" : "aeiou",
  "id" : ""
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

A new Skill Skill

404

Project not found

422

Unprocessable Entity Error

Up

POST /projects/{projectId}/features/{featureId}/skills

Adds required skills to a feature (**addSkillsToFeature**)

This operation allows adding more required skills to a given Feature by providing a list of the Skill ids.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

featureId (required)

Path Parameter — ID of the realese format: int64

Request body

body (required)

Body Parameter — Array of Skill ids

Return type

Feature

Example data

Content-Type: application/json

```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
```

```
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

Success Feature

404

Project or Feature not found

422

Unprocessable Entity Error

Up

POST /projects/{projectId}/resources/{resourceId}/skills

Add skills to a resource (**addSkillsToResource**)

This operation allows assigning more skills to a given Resource by providing a list of the Skill ids.

Path parameters

projectId (required)

Path Parameter — ID of the project (e.g. "1" or "siemens")

resourceId (required)

Path Parameter — ID of the resource format: int64

Request body

body (required)

Body Parameter — Array of Skill ids

Return type

Resource

Example data

Content-Type: application/json


```
{
  "skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "id" : "",
  "availability" : 1.3579000000000001069366817318950779736042022705078125
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

Success Resource

404

Project or Resource not found

422
Unprocessable Entity Error

[Up](#)

DELETE /projects/{projectId}/skills/{skillId}

Remove a given Skill (**deleteSkill**)
This operation allows removing a given Skill from the Project settings.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
skillId (required)
Path Parameter — ID of the project format: int64
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project or Skill not found
422
Unprocessable Entity Error

[Up](#)

DELETE /projects/{projectId}/features/{featureId}/skills

Delete required skills from a feature (**deleteSkillsFromFeature**)
This operation allows deleting required skills from a given Feature by providing the Skill ids.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
featureId (required)
Path Parameter — ID of the realese format: int64
Query parameters
skill_id (required)
Query Parameter — IDs of the skills to remove
Return type
Feature
Example data
Content-Type: application/json

```
{
  "code" : "",
  "depends_on" : [ "" ],
  "required_skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "effort" : 1.3579000000000001069366817318950779736042022705078125,
  "id" : "",
  "deadline" : "2000-01-23T04:56:07.000+00:00",
  "priority" : ""
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Success Feature
404
Project or Release not found
422
Unprocessable Entity Error

Up

DELETE /projects/{projectId}/resources/{resourceId}/skills

Remove skills from a resource (**deleteSkillsFromResource**)
This operation allows removing skills from a given Resource by providing the Skill ids.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
resourceId (required)
Path Parameter — ID of the resource format: int64
Query parameters
skillId (required)
Query Parameter — IDs of the skills to remove
Return type
Resource
Example data
Content-Type: application/json

```
{
  "skills" : [ {
    "name" : "aeiou",
    "description" : "aeiou",
    "id" : ""
  } ],
  "name" : "aeiou",
  "description" : "aeiou",
  "id" : "",
  "availability" : 1.3579000000000001069366817318950779736042022705078125
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
Success Resource
404
Project or Resource not found
422
Unprocessable Entity Error

Up

GET /projects/{projectId}/skills

Get project skills (**getProjectSkills**)
This operation retrieves a list of the possible skills required in the given Project.

Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
Return type
array[Skill]
Example data
Content-Type: application/json
```
[ {
  "name" : "aeiou",
  "description" : "aeiou",
  "id" : ""
} ]
```
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
An array of skills
404
Project not found
422
Unprocessable Entity Error

Up
`PUT /projects/{projectId}/skills/{skillId}`
Modify a given Skill (**modifySkill**)
This operation allows modifying the parameters of a given Skill, namely name and description.
Path parameters
projectId (required)
Path Parameter — ID of the project (e.g. "1" or "siemens")
skillId (required)
Path Parameter — ID of the project format: int64
Request body
body (required)
Body Parameter — Skill parameters that can be modified
Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Success
404
Project or Skill not found
422
Unprocessable Entity Error

Up
Models

[Jump to Methods]
Table of Contents

1. Error
2. Feature

3. FeatureData
4. FeatureId
5. Job
6. NewFeatureData
7. NewProjectData
8. NewRelease
9. Plan
10. Project
11. ProjectData
12. Release
13. ReleaseData
14. Resource
15. ResourceData
16. ResourceId
17. Skill
18. SkillData
19. SkillId

Error -
Up

code (optional)
Integer format: int32
message (optional)
String
fields (optional)
String
Feature -
Up

id (optional)
Integer Unique identifier representing a feature
code (optional)
Integer Unique identifier from WP3.3
name (optional)
String Display name of feature.
description (optional)
String Description of feature.
effort (optional)
BigDecimal Effort to implement the feature.
deadline (optional)
Date format: date-time
priority (optional)
Integer It coms from Wp3.3
required_skills (optional)
array[Skill]
depends_on (optional)
array[Feature] array of features
FeatureData -
Up

name (optional)
String Display name of feature.
description (optional)
String Description of feature.
effort (optional)
BigDecimal Effort to implement the feature.
deadline (optional)
Date format: date-time
priority (optional)
Integer It coms from Wp3.3
FeatureId -

[Up](#)

feature_id (optional)
Integer Unique identifier representing a feature
Job -
[Up](#)

starts (optional)
Date format: date-time
ends (optional)
Date format: date-time
resource (optional)
Resource
feature (optional)
Feature
depends_on (optional)
array[Job] array of jobs
NewFeatureData -
[Up](#)

code (optional)
Integer Unique identifier from WP3.3
name (optional)
String Display name of feature.
description (optional)
String Description of feature.
effort (optional)
BigDecimal Effort to implement the feature.
deadline (optional)
Date format: date-time
priority (optional)
Integer It coms from Wp3.3
NewProjectData -
[Up](#)

name (optional)
String Display name of project.
description (optional)
String
effort_unit (optional)
String e.g. "bin"
hours_per_effort_unit (optional)
BigDecimal
hours_per_week_and_full_time_resource (optional)
BigDecimal
NewRelease -
[Up](#)

name (optional)
String Display name of release.
description (optional)
String
starts_at
Date format: date-time
deadline
Date format: date-time
Plan -
[Up](#)

id (optional)
Integer
created_at (optional)

date format: date
release_id (optional)
Integer
jobs (optional)
array[Job] array of jobs
Project -
Up

id (optional)
Integer
name (optional)
String Display name of project.
description (optional)
String
effort_unit (optional)
String e.g. "bin"
hours_per_effort_unit (optional)
BigDecimal
hours_per_week_and_full_time_resource (optional)
BigDecimal
resources (optional)
array[Resource]
ProjectData -
Up

effort_unit (optional)
String e.g. "bin"
hours_per_effort_unit (optional)
BigDecimal
hours_per_week_and_full_time_resource (optional)
BigDecimal
Release -
Up

id (optional)
Integer
name (optional)
String Display name of release.
description (optional)
String
starts_at (optional)
Date format: date-time
deadline (optional)
Date format: date-time
resources (optional)
array[Resource]
ReleaseData -
Up

name (optional)
String Display name of release.
description (optional)
String
starts_at (optional)
Date format: date-time
deadline (optional)
Date format: date-time
Resource -
Up

id (optional)
Integer

name (optional)
String Display name of resource.
description (optional)
String
availability (optional)
BigDecimal
skills (optional)
array[Skill]
ResourceData -
Up

name (optional)
String Display name of resource.
description (optional)
String
availability (optional)
BigDecimal
ResourceId -
Up

resource_id (optional)
Integer Unique identifier representing a resource
Skill -
Up

id (optional)
Integer
name (optional)
String Display name of skill.
description (optional)
String
SkillData -
Up

name (optional)
String Display name of skill.
description (optional)
String
SkillId -
Up

skill_id (optional)
Integer Unique identifier representing a skill

Referències

1. Fases del desarrollo de software [En línia] <https://es.wikipedia.org/wiki/Fases_del_desarrollo_de_software> [Consulta: 24 feb. 2017]

2. Release Planner [En línia] <http://www.expertdecisions.com/> [Consulta: 24 feb. 2017]

3. SUPERSEDE [En línia] <https://www.supersede.eu/> [Consulta: 25 feb. 2017]

4. Cuál es el salario de un project manager [En línia] <http://www.obs-edu.com/es/blog-investigacion/project-management/cual-es-el-salario-de-un-project-manager> [Consulta: 10 mar. 2017]

5. Salario: Analista web en España [En línia] <http://espana.jobtonic.es/salary/26526/16124.html> [Consulta: 10 mar. 2017]

6. Guía salarial para programadores [En línia] <https://www.jobfluent.com/blog/guia-salarial-para-programadores-barcelona-2015?lang=es> [Consulta: 10 mar. 2017]

7. Salario: Tester en España [En línia] <http://espana.jobtonic.es/salary/26526/16702.html> [Consulta: 10 mar. 2017]

8. Los mejores salarios seguirán siendo para los ingenieros [En línia] <http://cincodias.com/cincodias/2016/05/10/sentidos/1462903664_151432.html> [Consulta: 10 mar. 2017]

9. Elaborar el presupuesto para un proyecto de Software de pequeña o mediana escala [En línia] <http://ingluisfransv.blogspot.com.es/2015/07/elaborar-el-presupuesto-para-un.html> [Consulta: 11 mar. 2017]

10. Swagger [En línia] <http://swagger.io/> [Consulta: 11 mar. 2017]

11. Postman [En línia] <https://www.getpostman.com/> [Consulta: 11 mar. 2017]

12. Spring Tool Suite [En línia] <https://spring.io/tools/sts> [Consulta: 11 mar. 2017]

13. The world's leading software development platform · GitHub [En línia] <https://github.com/> [Consulta: 11 mar. 2017]

14. Worbench - Products [En línia] <https://www.mysql.com/products/workbench/> [Consulta: 12 febrer 2017]

15. Worbench - Downloads [En línia] <https://dev.mysql.com/downloads/workbench/> [Consulta: 12 febrer. 2017]

16. Gestión de Riesgos en la informàtica del software [En línia] <http://dis.um.es/~barzana/Informatica/IAGP/IAGP_riesgos.html> [Consulta: 15 abril 2017]

17. Ley Orgánica de Protección de Datos de Carácter Personal (España) [En línia] <https://es.wikipedia.org/wiki/Ley_Org%C3%A1nica_de_Protecci%C3%B3n_de_Datos_de_Car%C3%A1cter_Personal_(Espa%C3%B1a)> [Consulta: 17 abril 2017]

18. JUnit [En línia] <https://es.wikipedia.org/wiki/JUnit> [Consulta: 1 març 2017]

19. JUnit 4 [En línia] <http://junit.org/junit4/> [Consulta: 1 març 2017]

20. Test Driven Development - TDD [En línia] <https://en.wikipedia.org/wiki/Test-driven_development> [Consulta: 1 març 2017]

21. SUPERSEDE Release Planner API to UI Documentation - per l'annex - [En línia] <https://supersede-project.github.io/replan/replan_controller/API-UI.html> [Consulta: 19 jun 2017]