Modelando

una gran

arquitectura empresarial

Caso de uso: *Ciclo DevOps*

Roberto Frutos Renedo
Julio 2015

Índice

1. Acerca de mí

Me llamo Roberto Frutos Renedo. Nací un 20 de octubre de 1985 en Valladolid, España y ahora mismo estoy trabajando para ISBAN (Grupo Santander) en el área de Arquitectura Tecnología y Operaciones, en Madrid.

Mi andadura por Madrid comenzó siendo bastante joven, una vez que terminé la carrera de Ingeniería Técnica Informática. Desde ese momento, comencé a trabajar en un sector, el del software, que me apasiona, y que creo que tiene mucho que ofrecer a personas inquietas y entusiastas como yo.

Me entusiasma conocer y dominar cualquier tipo de tecnología, también motivado por la gran variedad de las mismas con las que me he tenido que enfrentar, unas veces por decisión propia, otras por las circunstancias del momento, lo cual me ha permitido acumular una gran experiencia a lo largo de estos años como profesional del mundo del software. Creo que esta experiencia, junto con mi carácter entusiasta por conocer y aprender, son las razones que me han llevado a tratar de plasmar mi experiencia a la hora de diseñar la arquitectura de un gran sistema informático como es en este caso un ciclo de DevOps.

En cuanto a mis aficiones, todo lo relacionado con las dos ruedas. Cuando puedo trato de buscar un hueco para hacer MTB, una de mis pasiones desde que era bien pequeño y mi padre me compró mi primera bicicleta, desde ese momento no he parado. Además de la bicicleta, me encantan las motos, hasta hace un año practicaba moto-cross, pero por diversas razones lo fui abandonando y finalmente vendí la moto. Ahora estoy buscando una nueva, así que pronto retomaré esta excitante afición.

Por otra parte, creo que soy un poco adicto a los retos, y así como hasta el momento había dejado de lado el aprendizaje de otras lenguas, el año pasado decidí apuntarme a la Escuela Oficial de Idiomas, y aunque nunca haya sido una de las materias que me haya apasionado especialmente, conseguí superar el primer curso de intermedio (B 1.1), por lo que si todo va bien este año conseguiré el certificado B1.

Como con esto parece que aún tengo algo de tiempo libre (esto no lo piensa así mi pareja), estoy trabajando/estudiando con el objetivo de conseguir la certificación en ArchiMate 2.1 antes de retomar la Escuela Oficial de Idiomas.

Este es un extracto de cómo soy, mis intereses personales, aficiones y carácter.

2. Motivaciones

A lo largo de mis pocos años como profesional del sector, unos nueve, siempre me ha rondado la idea de transmitir mi experiencia en ciertos campos en los que he ido adquiriendo un gran conocimiento, pero debido a la escasez de tiempo libre o a que quizás me he considerado excesivamente joven como para escribir algo con cierto valor para otros profesionales del mundo del software, lo he ido aparcando y hasta ahora no me había decidido a hacerlo.

Aún no tengo claro si esto ha cambiado, es decir, si este documento será de utilidad a otras personas, uno de mis objetivos es que así sea.

Creo que este es el momento apropiado para hacerlo, y que mi experiencia modelando una gran arquitectura empresarial puede ser de gran utilidad a otros que se enfrenten a un problema similar por primera vez. Está claro que hay cosas que mejorar en la arquitectura que mostraré a continuación, aún está en proceso de construcción, por lo que considero que es algo normal en esta fase. Además, comentar que sólo se mostrarán algunas partes de la misma, pero creo que será más que suficiente como para tener una visión bastante buena de cómo abordar un problema similar.

Cualquiera que tenga acceso al documento puede contactarme con el objetivo de transmitirme sus valoraciones, críticas o comentarios acerca del mismo en la siguiente dirección de correo electrónico: robfrut@yahoo.es

3. Contexto

Para situarnos en el contexto del problema, voy a tratar de clarificar las circunstancias, condicionantes y razones por la que se comenzó a abordar este proyecto, del que únicamente quiero tratar de expresar mis conclusiones particulares.

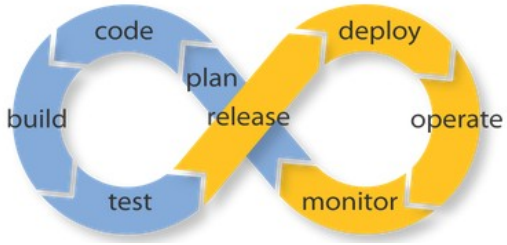
De una parte hasta ahora, se está hablando y comentando en todo tipo de artículos, libros y compañías del sector, los beneficios y bondades de DevOps. No voy a entrar en debatir, comentar o argumentar a favor o en contra. Simplemente decir que a nuestro área llegó el proyecto de implantar un ciclo completo DevOps, así como la tarea de evangelizar esta cultura a lo largo de la compañía, con el fin último de conseguir desarrollos con un menor time-to-market. Tarea, este segunda, más complicada si cabe aun que la primera, no por ello imposible ni rechazada por nuestro equipo.

Por otra parte, contextualizar la compañía. Una organización enorme, con multitud de departamentos, cada uno especializado en cada tarea inimaginable por cualquiera en el mundo del software. Los plazos para la consecución de proyectos de desarrollo de software son inasumibles por cualquiera de los implicados en los mismos. Desde el arranque del mismo, con la petición de infraestructura y accesos a los diferentes sistemas, hasta el delivery del mismo en los diferentes entornos productivos. Todo esto, provoca que los aires del exterior terminen llegando aquí también, y nos terminemos embarcando en este proyecto tan fascinante a la vez que complejo.

Se ha de comentar también, la importancia de dos requisitos o restricciones de negocio impuestas desde la organización como premisa a la hora de acometer dicho proyecto: uno, el uso de herramientas opensource, y dos, la aplicación de estándares conocidos en el mundo del software, así como, en su defecto, estándares defacto aceptados en la industria.

3.1. Sistema a modelar

Como ya se ha comentado anteriormente, el sistema a modelar es un ciclo completo de DevOps. En la web, podréis encontrar multitud de información teórica y práctica de qué es DevOps, su cultura, cómo aplicarlo empleando diferentes tecnologías y cuáles son los puntos más complejos donde prestar especial atención, ya que pueden ser objeto de diversas problemáticas.



Comentar la importancia cultural de este cambio, ya que implica la colaboración y trabajo conjunto entre dos mundos que hasta el momento habían vivido aislados uno del otro, sin importar lo más mínimo la repercusión que estaba provocando en los productos finales y, por tanto, en la imagen percibida por el cliente final que, a la postre, es quien da validez a un producto, tanto en lo respectivo a su adecuación a las funcionalidades solicitadas, como en los plazos y condiciones de entrega finales.

En este punto, destacar que en estos momentos aún continuamos trabajando en conseguir que los departamentos de desarrollo y operaciones se integren y trabajen de la manera más eficiente

posible, al mismo tiempo que estamos tratando de mostrar cuáles serán su rol y funciones en un sistema que es completamente disruptivo con el anterior en este sentido. Con esto, quiero referirme a cómo se han de estructurar las aplicaciones, qué tipo de pruebas realizar, cómo controlar que se está desarrollando el software con la suficiente calidad, cómo monitorizar las aplicaciones, cómo tratar los cambios de configuración de las mismas...y otro gran número de tareas que en este escenario se abordan de manera totalmente diferente a como se venían haciendo hasta la fecha.

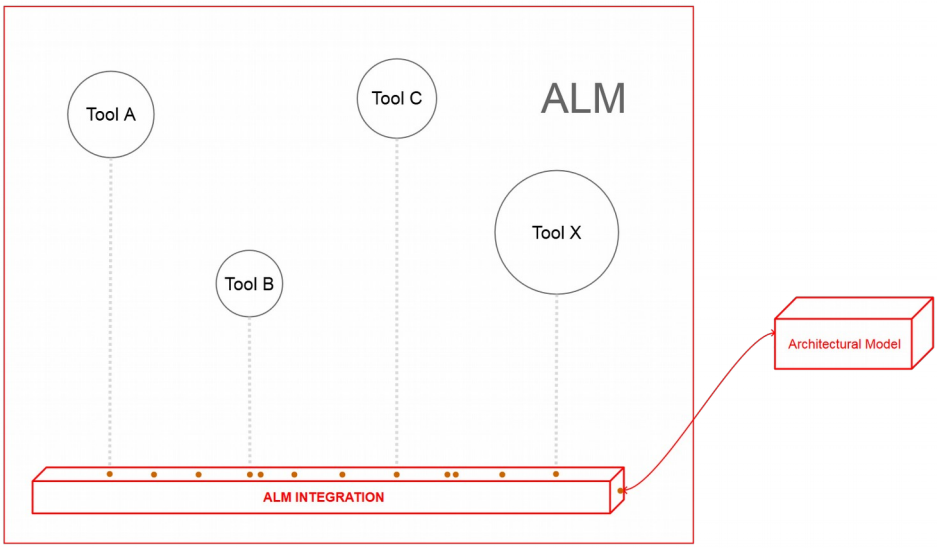
3.2. Primeros pasos

Tras determinar cuál es el sistema a diseñar y teniendo un ligera idea del mismo, se decide comenzar a investigar en profundidad el tema. Esto me deriva en varios meses de investigación y aprendizaje de qué es esto de DevOps, así como de numerosas pruebas de algunas herramientas que podrían sernos de utilidad para cubrir algunas funcionalidades que parecían aflorar en dicha arquitectura. Acerca de esto, comentar que la mayoría a día de hoy han sido desechadas por unas razones u otras debido a que hemos ido madurando una arquitectura que en esos momentos no era más que un boceto de lo que es a día de hoy.

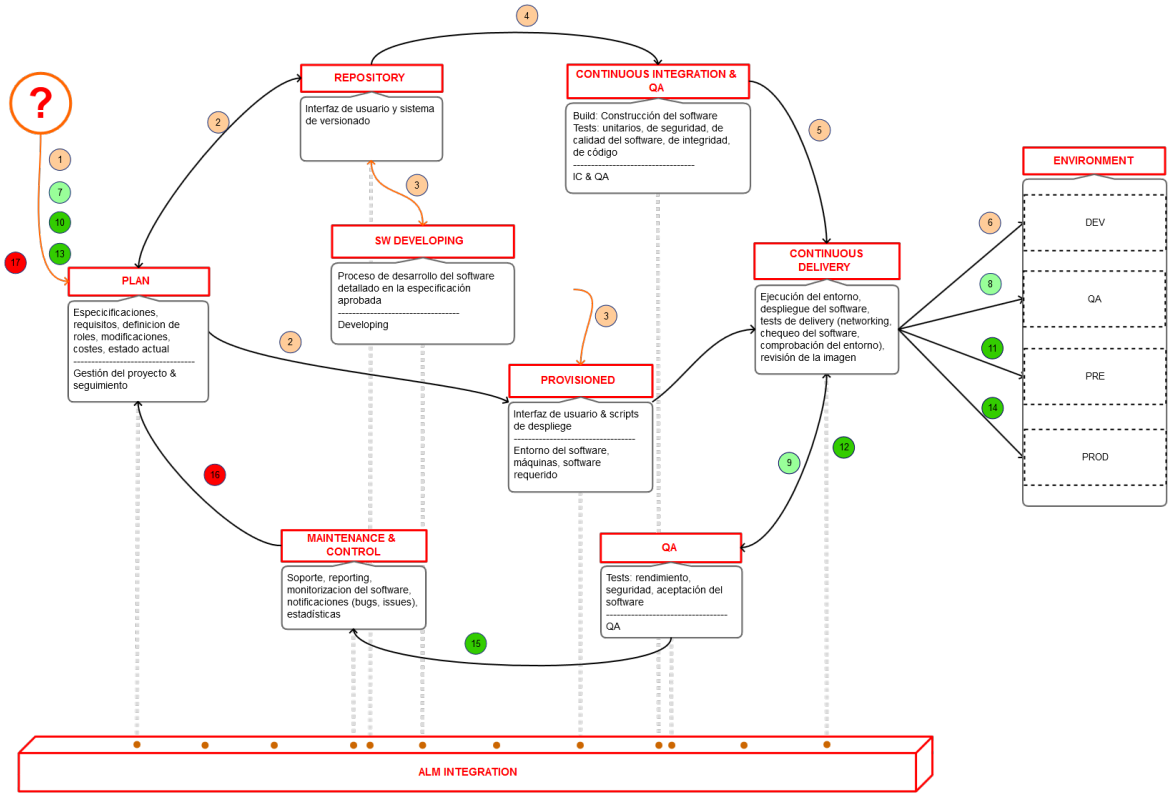
En este punto, comienzo a ver varios módulos en los que podría focalizar mis esfuerzos a la hora de clarificar esta arquitectura tan compleja. Básicamente son:

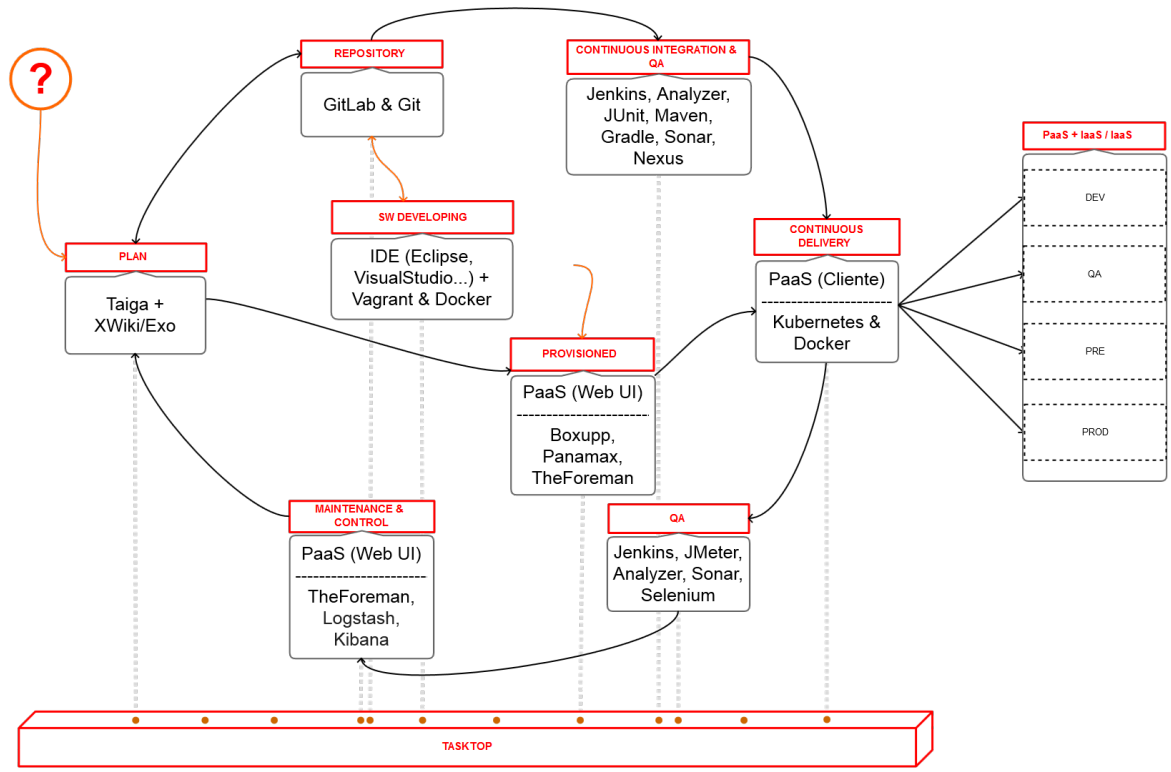
- Gestión del proyecto
- Diseño de la infraestructura
- Desarrollo del software
- Integración continua
- Entrega continua
- Aceptación del software
- Soporte y mantenimiento

Esta pequeña división funcional del ciclo DevOps, me lleva a tratar de plasmar en diversos bocetos las relaciones entre estas funciones, los roles participantes en el ciclo DevOps, las herramientas que podrían cubrir estas funcionalidades y, en definitiva, a clarificar las ideas que durante algún tiempo había ido recopilando y asimilando.

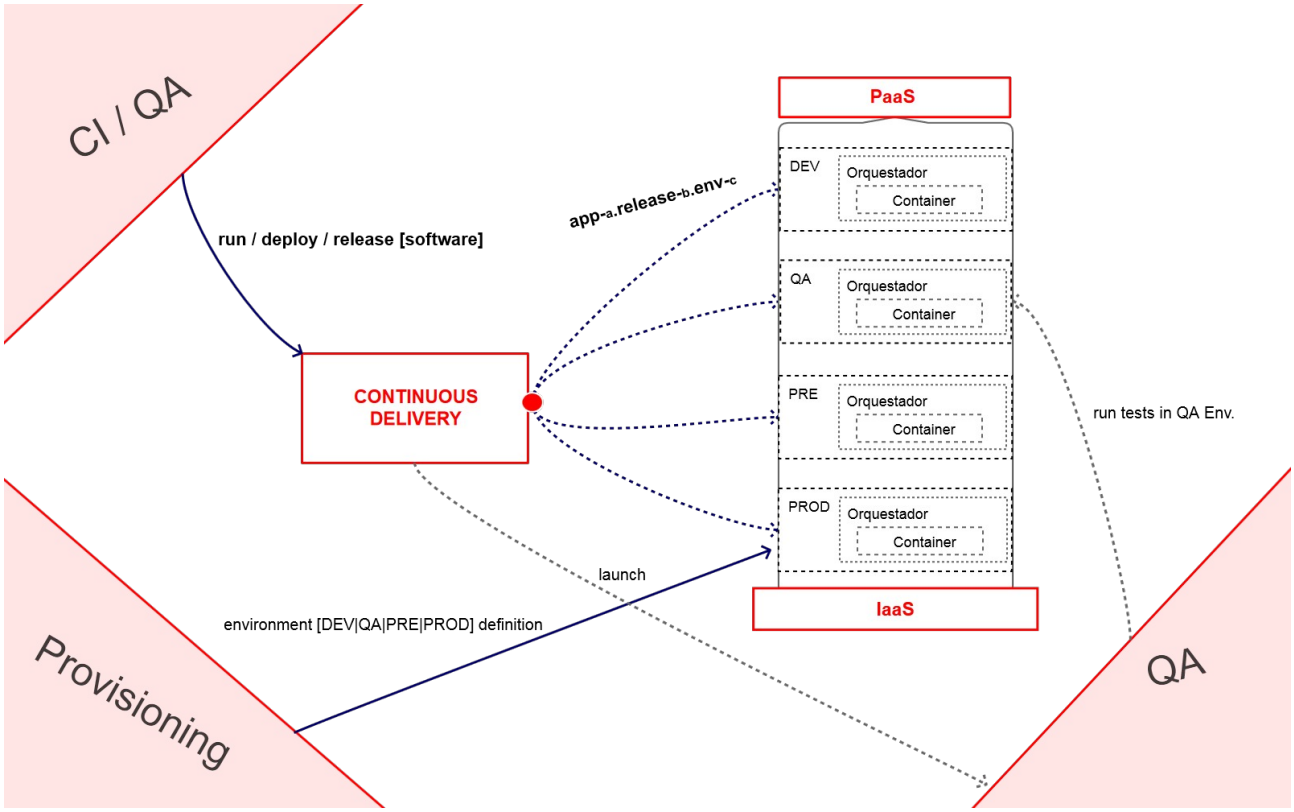


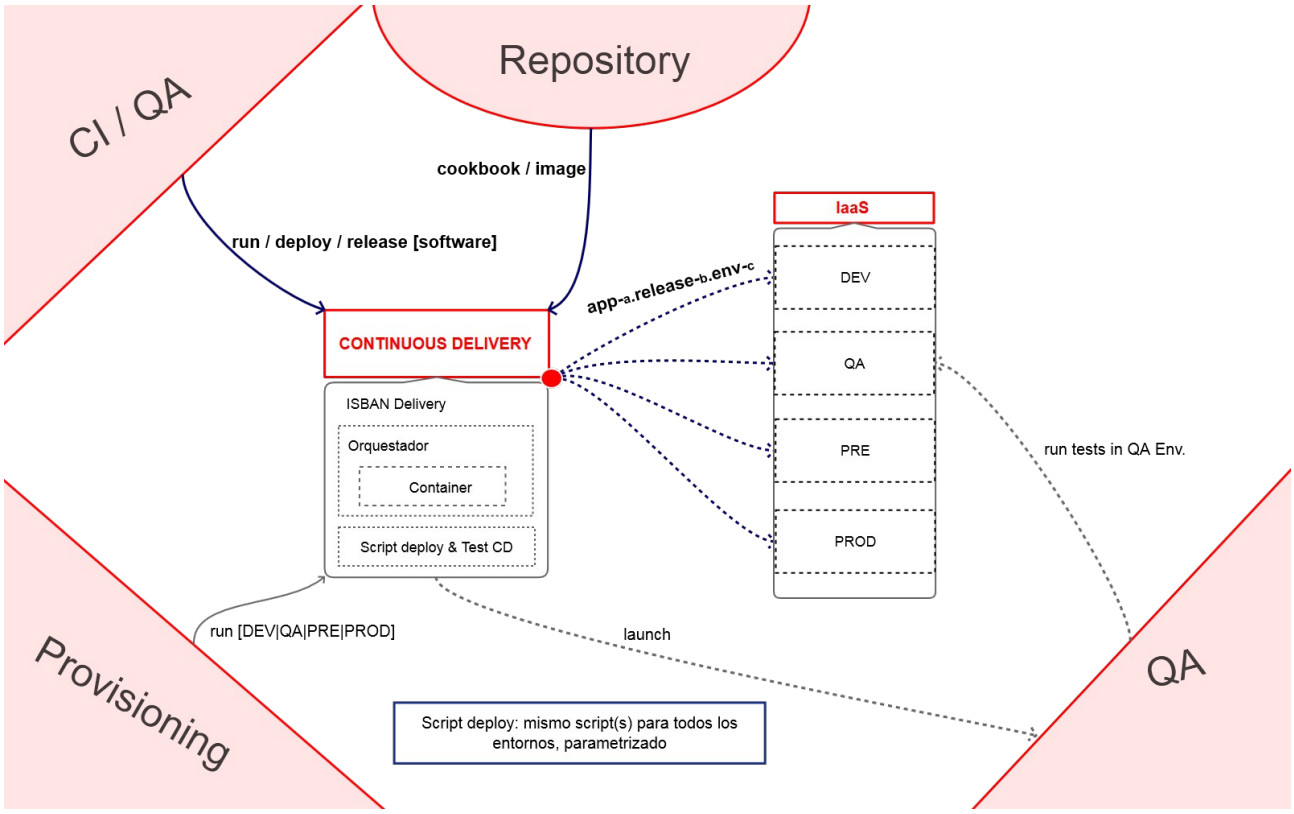
Como queda presente, son unos bocetos iniciales para reflejar algunas ideas. Éstos me harían ver que necesitaba algo más y más complejo con lo que poder representar una arquitectura de estas dimensiones y complejidad.





Además de los bocetos anteriores, cuyo propósito es el de clarificar el ciclo DevOps en su conjunto, también dediqué cierto tiempo en asimilar una de las partes más confusas a la par que crítica del mismo, la entrega continua.





En los bocetos hay diferentes errores que no se han corregido, puesto que son eso, bocetos, documentos que no se deben mantener vivos. Han cumplido con su objetivo y no son más que una versión primitiva de la arquitectura que queremos construir.

Los bocetos fueron de gran ayuda, recomiendo esta práctica a la hora de diseñar una arquitectura de este nivel, ya que nos permitirá saber donde tenemos que poner especial detalle llegado el momento de modelar el sistema mediante un lenguaje estándar.

3.3. ArchiMate 2.1

Aquí entra en juego el lenguaje estándar para modelar arquitecturas empresariales, ArchiMate. Desde este momento, comienzo a hacer un traspaso del conocimiento adquirido y plasmado en los borradores, a un modelo de arquitectura empresarial, entendible por todos nosotros y que nos permita hablar de elementos del sistema bajo el mismo marco de referencia, es decir, que utilicemos un lenguaje común.

En los siguientes puntos, se comentará cuáles han sido algunas de las diferentes vistas que he ido realizando con el objetivo de mejorar los bocetos de partida y ampliar el detalle que tenía hasta el momento de la arquitectura.

He de comentar que, hasta el comienzo de este apasionante proyecto, ArchiMate era un desconocido para mi, por lo que, llegados a este momento, sucedieron varias semanas de duro aprendizaje a fondo del mismo. Recomendar un libro que me ha sido de gran utilidad durante esta fase, así como agradecer a su autor, Gerben Wierda, la ayuda prestada con ciertas dudas.

“Mastering ArchiMate” by Gerben Wierda

Pero, como pasa con todo en esta vida, hasta que no lo haces no lo aprendes y comprendes de verdad. Dejo aquí una cita que ha marcado mi manera de ver las cosas en el mundo del software:

"Me lo contaron y lo olvidé; lo vi y lo entendí; lo hice y lo aprendí." - Confucio

Así, doy comienzo a un pequeño análisis de qué herramientas había disponibles para realizar el modelado con este lenguaje. Mencionar que la herramienta utilizada para modelar la arquitectura ha sido Archi, debido a que es opensource y de licencia libre. Es en este momento, cuando me pongo manos a la obra, y es cuando empiezo a descubrir todo el potencial de ArchiMate, su gran valor.

He de transmitir que, hasta la fecha, tanto la herramienta como el lenguaje han cumplido a la perfección con su cometido, proporcionando los mecanismos para añadir todo el detalle que he ido necesitando dar a la arquitectura.

4. Usando el lenguaje estándar

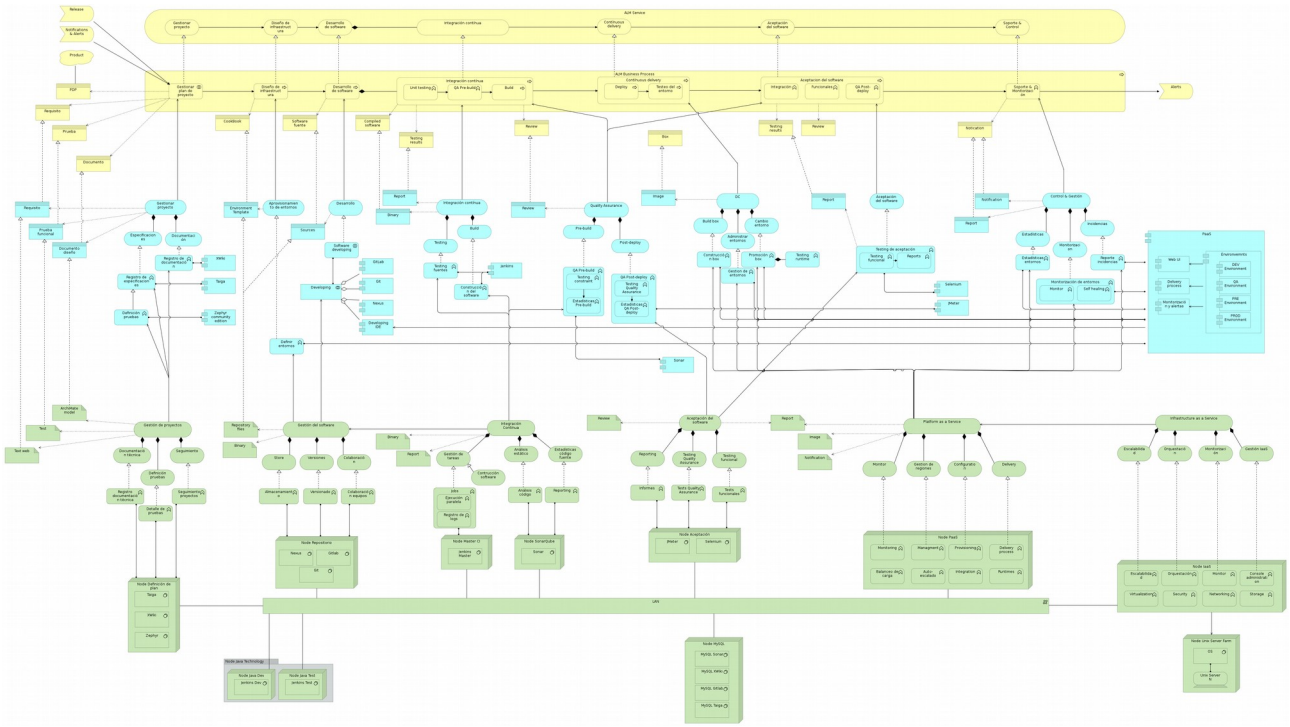
4.1. Modelando el sistema completo, ¿porqué?

Tras acometer los primeros bocetos, surge la necesidad de modelar la arquitectura de una manera más formal. Para ello, comenzamos a modelar el sistema con ArchiMate. En un primer momento, debido al poco conocimiento de la herramienta de modelado y de algunos importantes patrones a tener en cuenta, intento modelar la arquitectura en una sola vista, con el fin de, una vez detallada la arquitectura completa, ir obteniendo diferentes vistas que pudieran resultar de interés a los diferentes skateholders de la misma. Es decir, focalizar el sistema en diferentes puntos de interés en función de quién quisiera conocer el mismo.

Ésto puede parecer descabellado, y lo es. Pero de todos los errores se aprende, es una lección importante y que siempre hemos de tener presente, aunque nos cueste reconocerlo. De la poca utilidad de esta vista para los otros implicados, tuve constancia en el primer acercamiento de la arquitectura a otros miembros de la organización. En ese momento me dí cuenta de que únicamente me servía a mí, conocedor 100% de la arquitectura presente y futura a la que queríamos llegar.

No obstante, he de decir que, al contrario que para ellos, para mí fue de suma importancia intentar plasmar en una instantánea la arquitectura, ya que conseguí una visión del sistema que no habría podido tener de otro modo. He de comentar también, que ya podía adivinar el duro mantenimiento de la misma y la complejidad de tratar de entenderlo para una persona que no estuviese un contacto directo con el proyecto que estábamos realizando. Todo eso, como más adelante detallaré, desembocó finalmente en eliminar dicha vista del modelo de arquitectura empresarial DevOps que estábamos realizando.

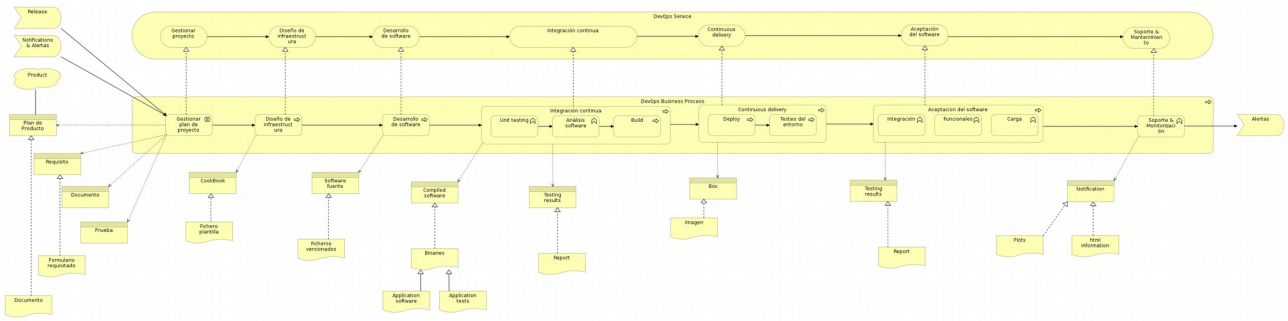
Tras mucho buscar, he conseguido recuperar la vista a la que he estado haciendo referencia, por lo que aquí os dejo un ejemplo de una vista carente de utilidad, debido a su poca legibilidad y enorme complejidad.



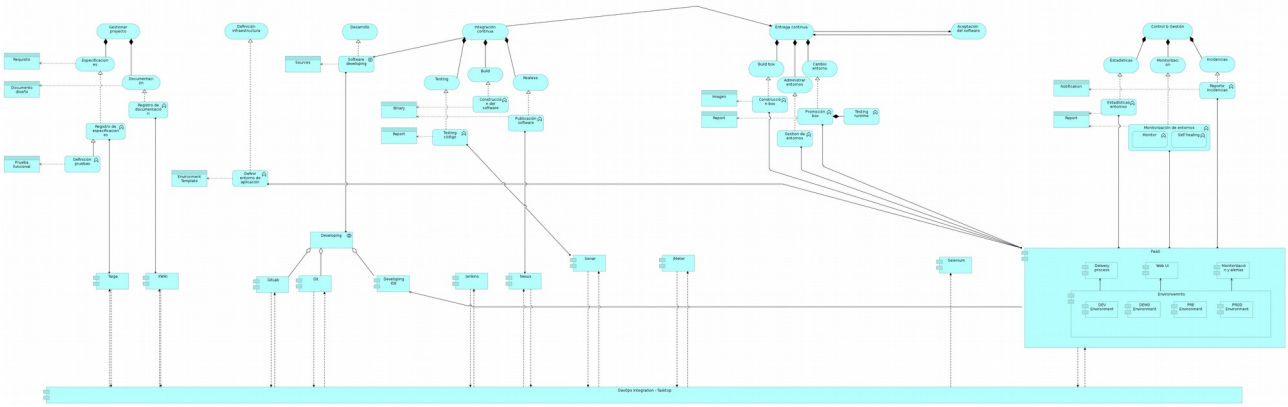
4.2. Vista por capas del ciclo

Como la visión del sistema completo era algo que no podíamos tener en una sola vista, algo que con el tiempo se ha convertido en evidente, decido aplicar un dicho muy conocido en el mundo en general, “divide y vencerás”. Así, divido las vistas de la arquitectura en función de la capa a la que hacen referencia, obteniendo tres vistas más sencillas y en las que podría continuar detallando (sin llegar a volverse algo inmanejable) la arquitectura DevOps.

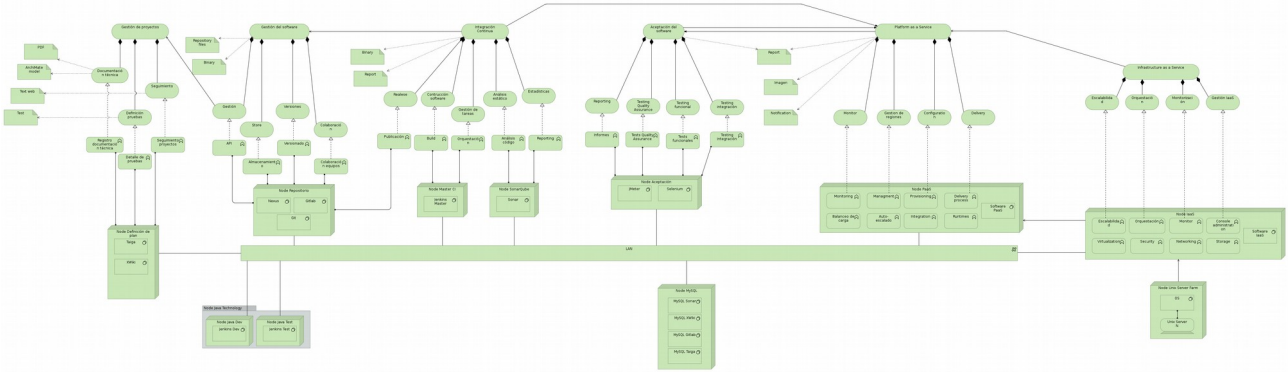
Vista capa de negocio de la arquitectura.



Vista capa de aplicación de la arquitectura.



Vista capa de infraestructura de la arquitectura.



4.3. Vista transversal de cada módulo

Llegados a este punto, surge la necesidad de tener una visión transversal de cada módulo que me ha resultado de interés. Con esto, trato de conseguir una visión completa de la arquitectura de un servicio de negocio en el ciclo DevOps, con el fin de que el equipo encargado de afrontar la implantación de dicho servicio pueda tener el detalle completo de a qué se enfrenta y conocer como debe realizar su tarea.

En este momento, es una idea que comienzo a dar forma poco a poco, y que más adelante detallaremos, mostrando algunos ejemplos este tipo de vistas.

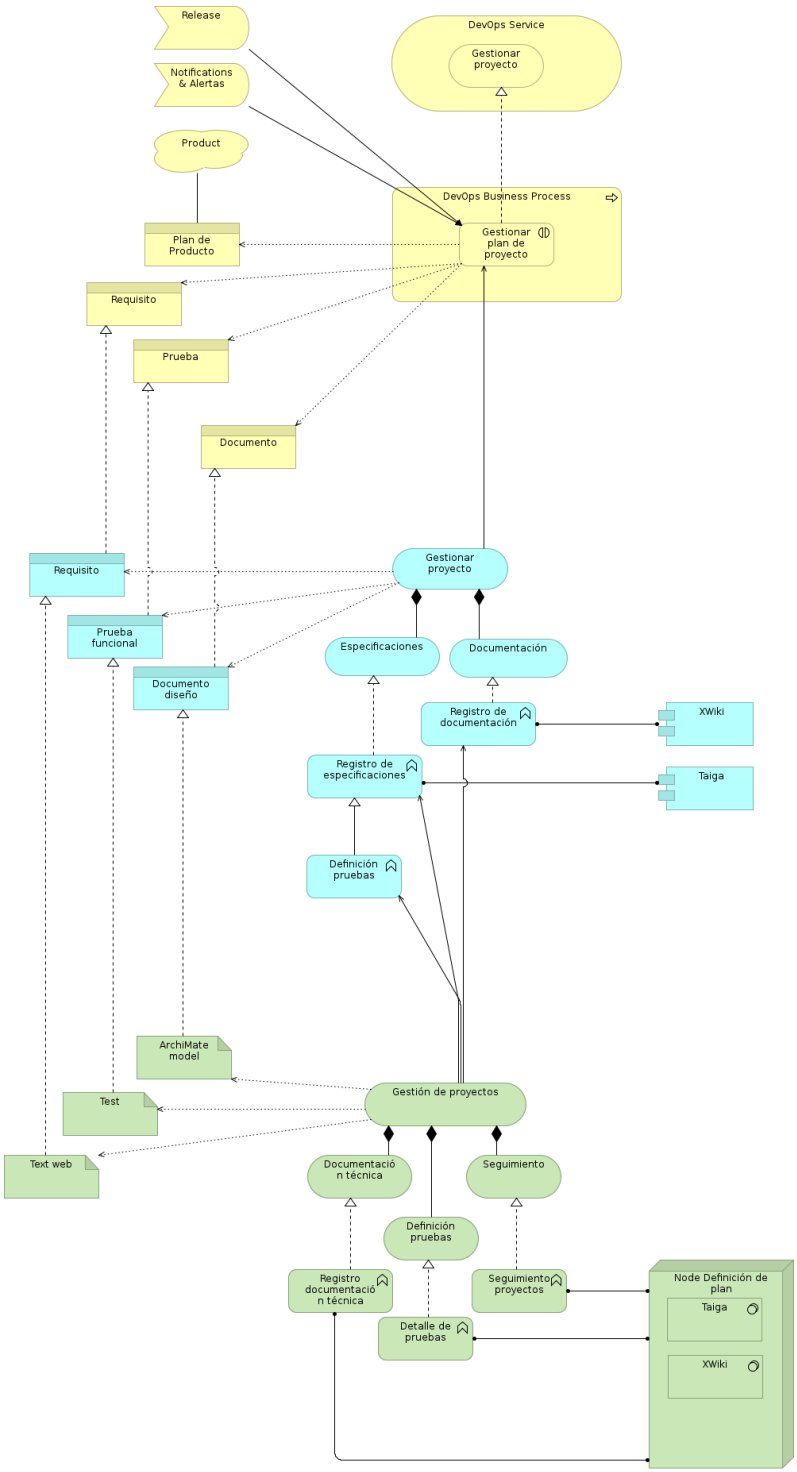
4.4. Conclusiones y errores

Para cerrar este punto, comentar un gran obstáculo con el que todo arquitecto de software se enfrentará al diseñar una arquitectura de tal envergadura, y que no es otro que lo que se ha venido explicando hasta el momento: la poca mantenibilidad y legibilidad de vistas complejas, pero su gran utilidad para aclarar la arquitectura, las relaciones entre los diferentes elementos y el tener una visión global de nuestro sistema. Podemos considerar este tipo de vistas como un paso previo a algo más mantenible y entendible por toda persona involucrada en un proyecto de este tipo.

Hasta aquí, se han cometido diversos errores, claro está. Como ya se ha comentado antes, el abordar una vista completa de toda la arquitectura, ha sido el principal. Dar las gracias a los comentarios constructivos de las personas expertas en este tipo de modelado (departamento de Arquitectura de Aplicaciones) y a mi responsable directo en el departamento, que sirvieron para darme cuenta de la situación y tratar de enmendarla cuanto antes.

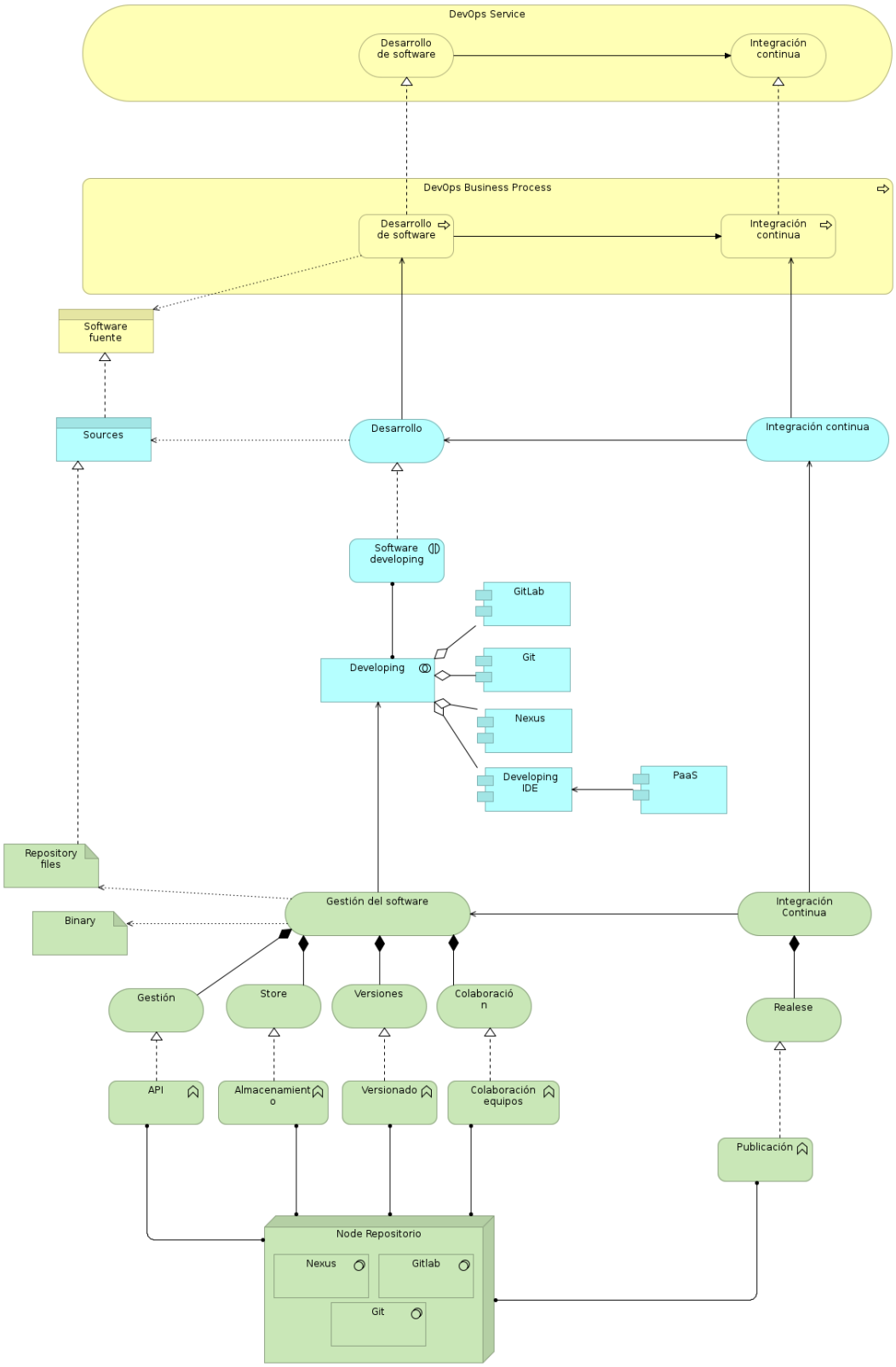
5. Vistas transversales

5.1. Gestión del proyecto



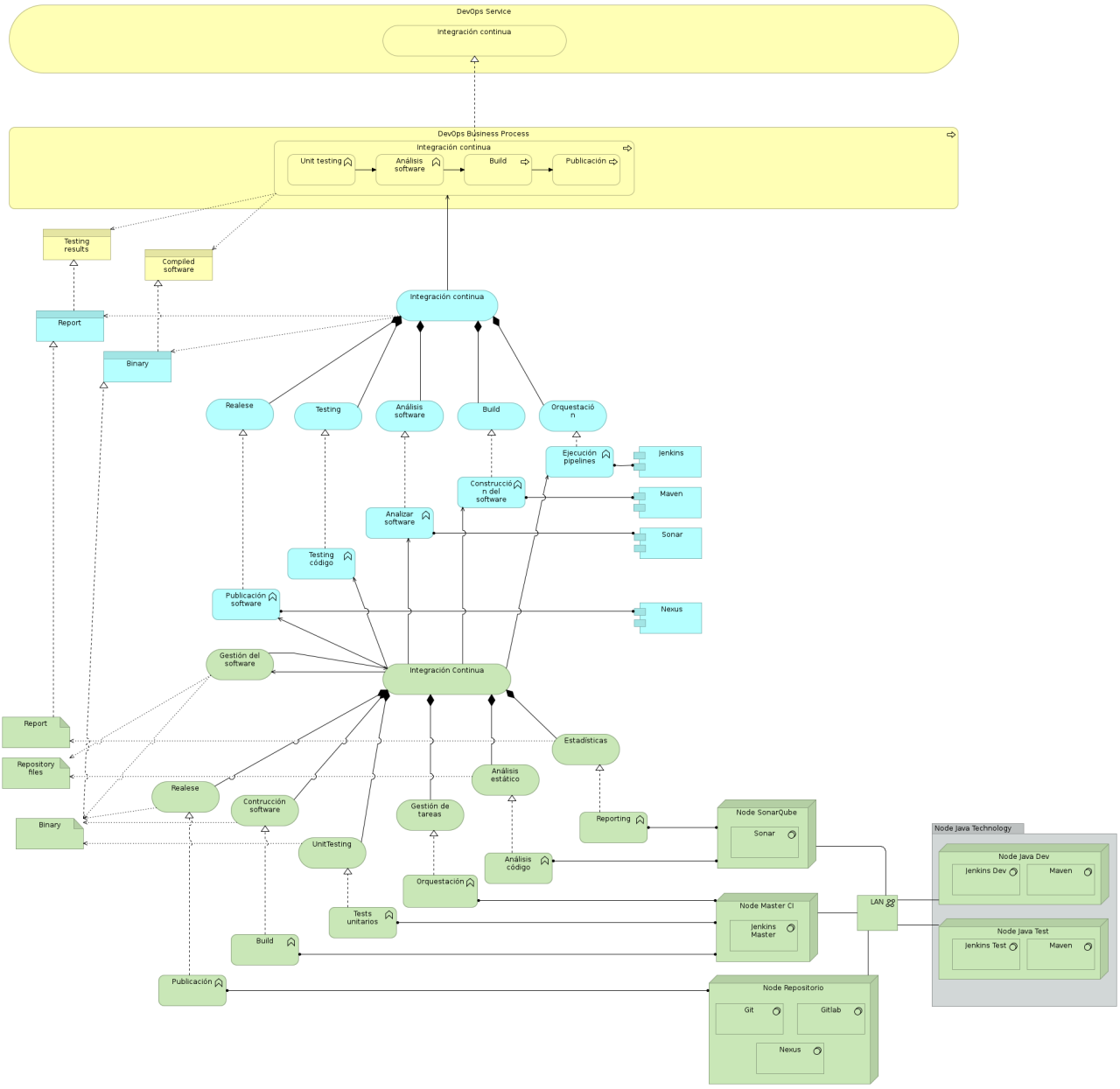
Determino, que la mejor manera de representar en la arquitectura DevOps, la definición de las especificaciones, tareas e historias de usuario es con una iteración de negocio. Esta iteración dará lugar a una serie de información que será clave para poder afrontar el proceso de desarrollo del software. Vemos que hay dos componentes de aplicación que proporcionan la funcionalidad requerida para que se pueda ofrecer estos servicios desde dicha capa. Estos componentes, serán instalados en un nodo específico destinado a este fin. Considero de suma importancia la funcionalidad que proporcionará este software de cara al servicio final de negocio: registro de documentación técnica, detalle de las pruebas a realizar y poder hacer un seguimiento del proyecto.

5.2. Desarrollo de software



Es importante en esta vista, la relación de negocio entre el servicio de desarrollo de software y el de integración continua, considerando que el primero es un precursor del segundo. Por otro lado, hemos de ver el desarrollo de software como una iteración de aplicación, que mediante una colaboración entre diferentes componentes es capaz de cubrir el servicio de desarrollo de software del que hace uso la capa de negocio. Importante, que esto se soporte desde la capa de infraestructura, con un nodo en el que se instale el software adecuado (GitLab, Git, Nexus) para soportar el servicio (Gestión del software) del que hacen uso los componentes de aplicación.

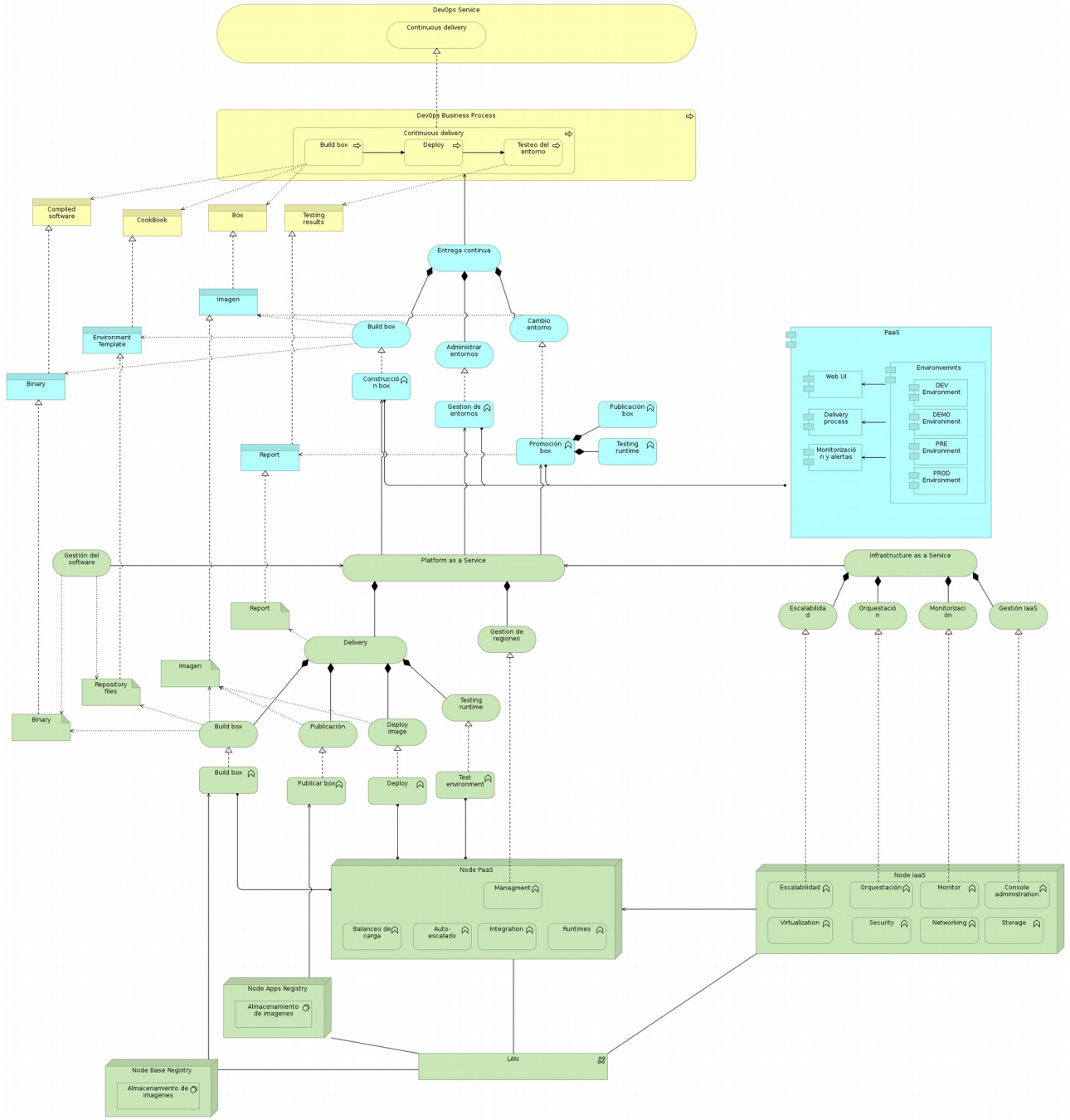
5.3. Integración continua



En este punto, entender claramente lo que considero es el proceso de integración continua, es decir, las pruebas unitarias, el análisis de código, la construcción del software y la publicación del mismo. Esta delimitación me ha llevado a diversas discusiones y debates, sin duda alguna, es conveniente hacerla por razones de modularidad y mantenimiento. Quede claro, que ésta es mi visión particular y que no tiene por qué ser la mejor solución. Destacar que hay diversos condicionantes entre cada funcionalidad o proceso que conforma la integración continua y que determinan si continuar avanzando hacia adelante.

Con esto como base, podemos entender mejor el servicio de aplicación del que hace uso el proceso de integración continua. Los cuatro componentes de aplicación son críticos para proveer las funcionalidades adecuadas a este servicio (ejecución de tareas, construcción del software, análisis de software y publicación de artefactos). En cuanto a la infraestructura necesaria para este módulo, es algo más complejo que los casos vistos anteriormente, ya que es importante reflejar la necesidad de tener dos nodos esclavos por tecnología soportada en la integración continua, además del nodo principal y un nodo específico para el análisis estático de software.

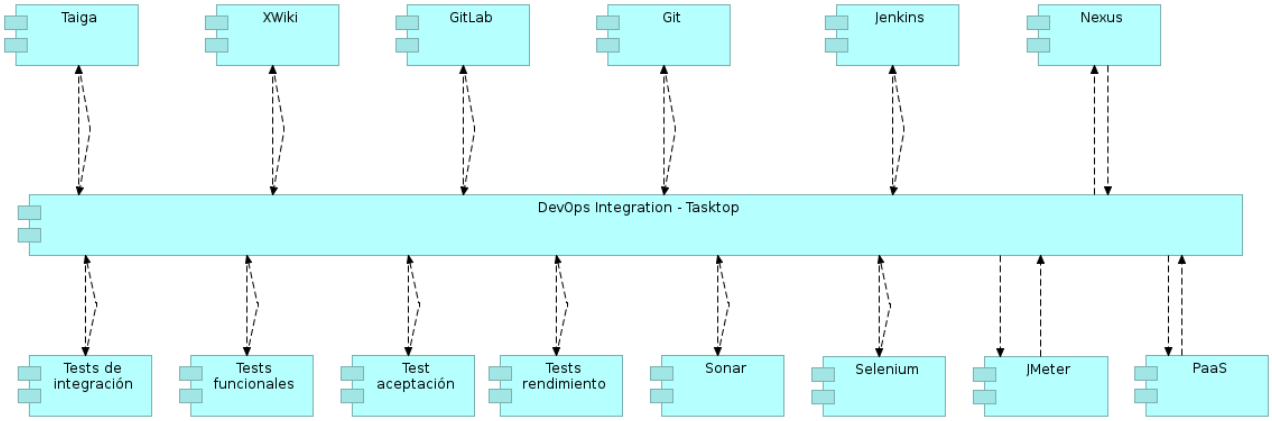
5.4. Entrega continua



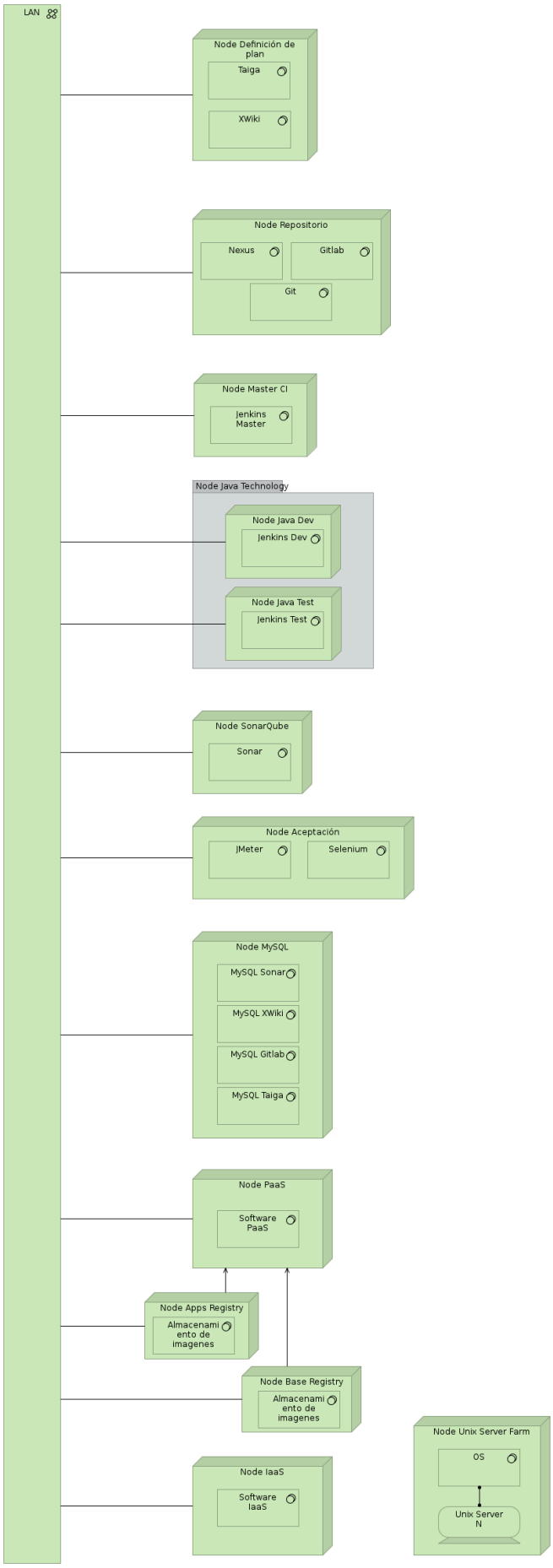
Es importante tener claro qué estamos considerando entrega continua: la construcción de una imagen con nuestro artefacto, el despliegue de esta imagen y la comprobación de que se ha desplegado correctamente, sin realizar pruebas del software, ya que he considerado de suma importancia tener un servicio de "Aceptación del software" que se ocupa de esto mismo y de evaluar los resultados con el objetivo de determinar si el software está correcto en el entorno y si es posible una promoción hacia el siguiente entorno. Teniendo en cuenta esto, se toma la decisión de cubrir este servicio de despliegue de aplicativos mediante un PaaS que nos proporcione la gestión de los entornos (Dev, Cert...), nos provea de un mecanismo de promoción entre los mismos y de otras funcionalidades de utilidad como la monitorización de aplicaciones, servicios de alertas, etc. Como se puede ver, ésto supone detallar la capa de infraestructura con el mayor detalle posible, tanto en los servicios disponibles hacia la capa de aplicación, como en la funcionalidad del PaaS en la que se apoyan los mismos.

6. Caminando hacia otras vistas de interés

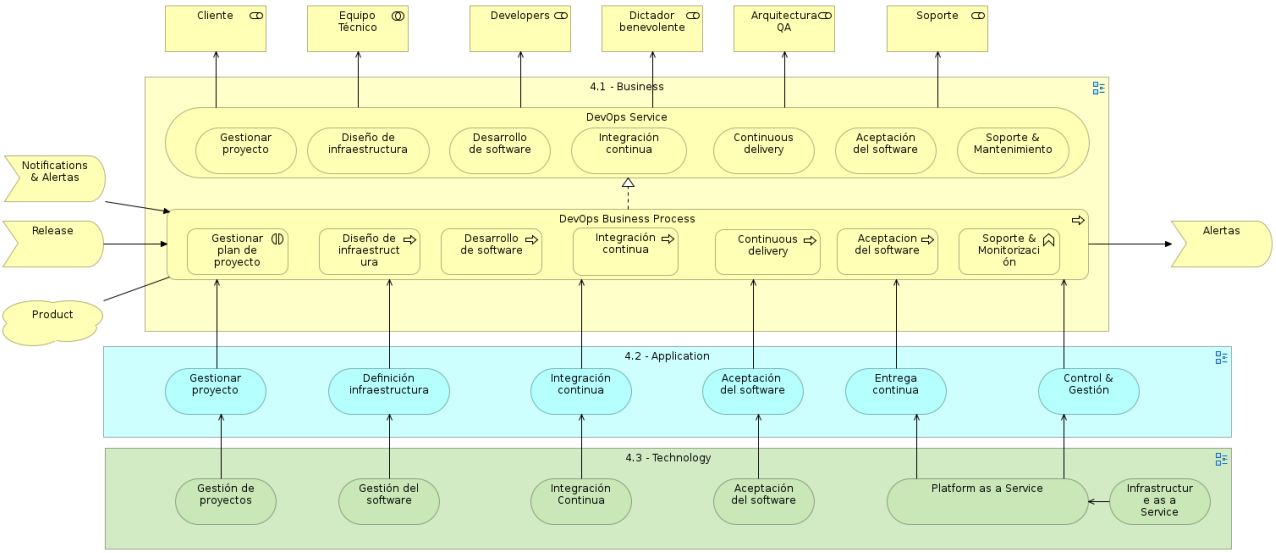
6.1. Comunicación de componentes



6.2. Propuesta infraestructura

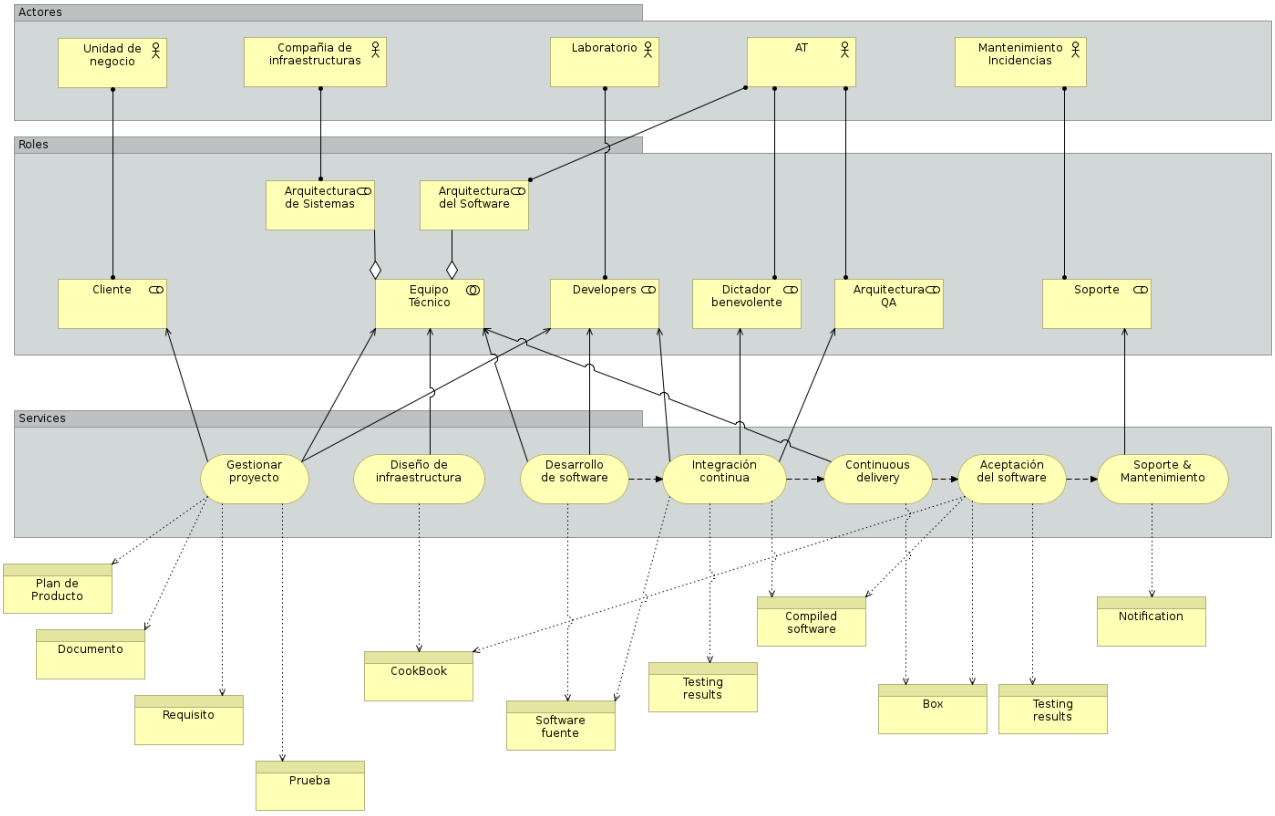


6.3. Vista general



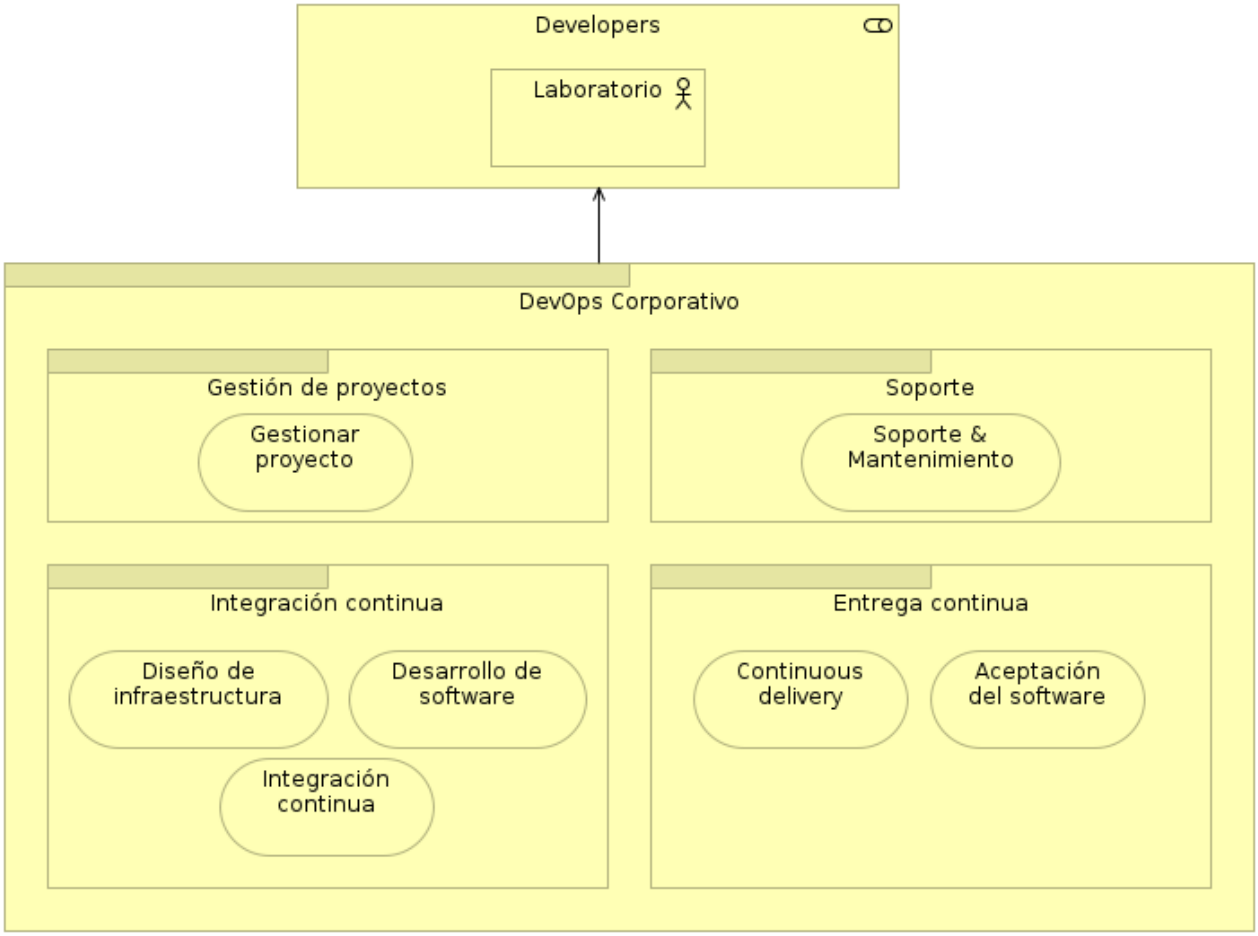
En esta vista trato de plasmar una visión general de los servicios externos de cada capa y cómo son utilizados por el resto de capas, es decir,la relación de uso a alto nivel entre los servicios de las diferentes capas. Así mismo, se tiene una visión general y básica de qué usuarios hacen uso de los servicios de negocio que proporciona el ciclo DevOps (Cliente, Desarrolladores, Soporte...). Ésta es una vista de gran utilidad para tener una visión general de qué arquitectura tenemos y de los servicios provistos por la misma. Destacar la posibilidad existente de que desde esta vista podemos obtener mayor detalle de cada capa y así conocer cuál es la arquitectura diseñada en las mismas.

6.4. Vista de interfaz con los servicios



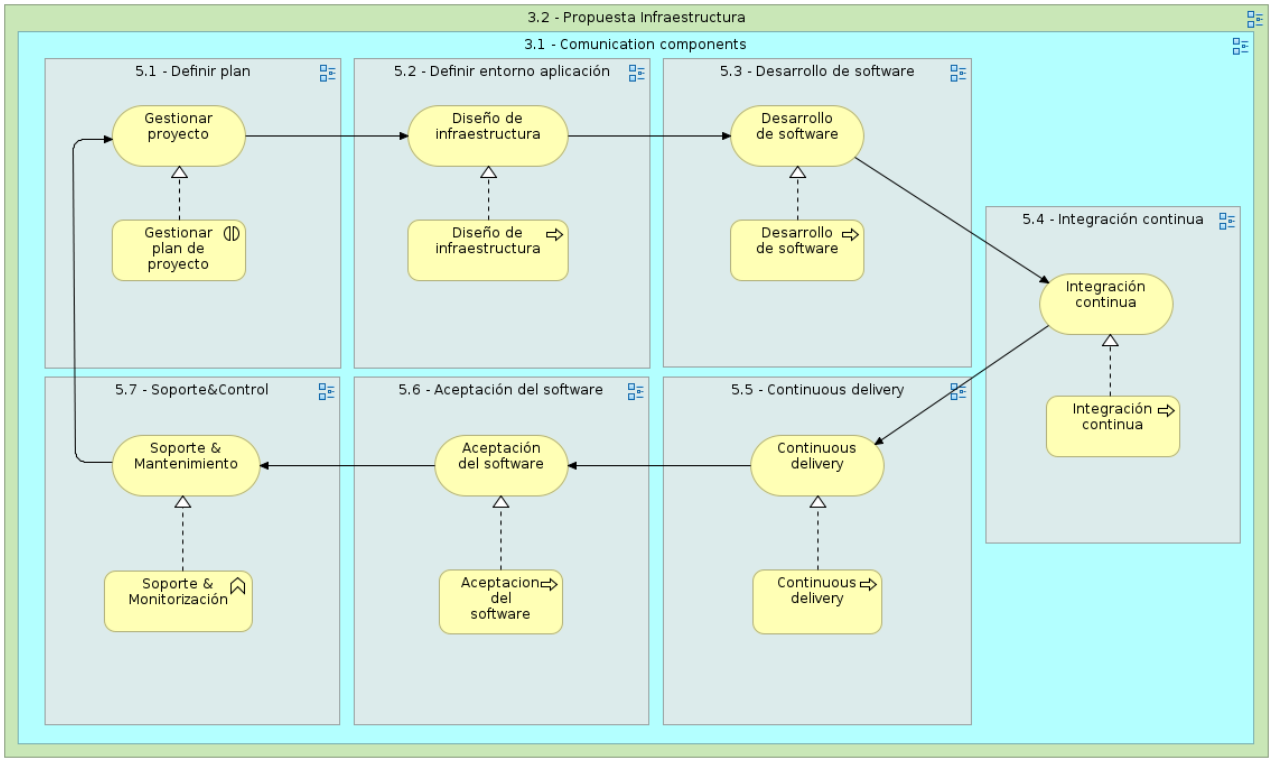
A través de esta vista, se ofrece un mayor detalle de cómo es la interacción con los servicios de negocio de DevOps, quiénes son los actores que participan y el rol que asumen en el ciclo. Se detalla también el uso que se hace de los objetos de negocio por parte de cada servicio, sin entrar en el detalle de las relaciones entre los mismos.

6.5. Vista de producto



De cara a negocio, no soy un experto en este tema, pero sí, a lo largo de los años he podido percibir que siempre es importante tener una perspectiva de qué pretendemos ofrecer con un producto, desarrollo o servicio (en este caso un servicio de DevOps) y cómo se puede utilizar, ya sea el paquete completo (DevOps Corporativo) o un módulo de manera individual (Gestión de proyectos, Soporte, Integración continua o Entrega continua). En este punto, trato de dividir DevOps Corporativo en lo que podrían ser productos más pequeños o módulos, que considero son utilizables sin llegar a tener que hacer un uso completo de nuestro ciclo DevOps.

6.6. Vista del ciclo



Es una vista en la que he trabajado de forma tardía, debería de haber abordado una visión de este tipo con anterioridad. Debido a que he intentado realizar un desarrollo ascendente de la arquitectura DevOps, reduciendo la complejidad pero aumentando el ángulo de visión, se ha hecho en la parte final del diseño. Aun así, considero de vital importancia tener esta vista, ya que nos permite navegar desde una visión global de DevOps hacia vistas con diferentes detalles del ciclo, como puede ser la vista transversal de cada módulo, la propuesta de infraestructura para una primera fase o los componentes de aplicación necesarios en el ciclo.

6.7. Valorando la utilidad de vistas estándar

Llegados a este punto, habéis visto algunas de las vistas principales de esta arquitectura. Hay otras, pero las he considerado de menor interés. Con todo esto, tenemos un diseño de arquitectura DevOps bastante completo, no definitivo, pero sí que abarca la mayoría de los elementos que queríamos definir. Aun así, sufrirá bastantes cambios hasta la implantación de la misma, por lo que este modelo deberá de mantenerse lo más actualizado posible, con el fin de tener en los modelos la visión más cercana a la realidad.

Mi objetivo en estos momentos es el de continuar avanzando en la definición de la arquitectura, mejorando y depurando las vistas actuales, además de incorporar otro tipo de vistas que aporten valor a la misma. En este sentido, estoy empezando a tener en cuenta la posibilidad de completar la arquitectura que estamos trabajando con algunas de las vistas estándar que sugiere ArchiMate 2.1, por lo que estoy realizando un trabajo de entendimiento de estas vistas, así como de su idoneidad para esta arquitectura.

En la actualidad está en un estado bastante básico, por lo que no creo conveniente aventurarme a realizar adelantos de estos avances. Simplemente comentar que, ya que el aprendizaje real del modelado empresarial se ha hecho con la práctica, he comenzado a detectar una situación curiosa, y no es otra que la correspondencia de ciertas vistas que he ido diseñando con algunas de las vistas estándar de ArchiMate 2.1. Esto es realmente importante ya que, sin guiarme fielmente por las vistas que nos sugiere el estándar y únicamente basándome en mi criterio, sentido común y necesidades del equipo, he desembocado en bastantes similitudes con su propuesta.

7. Conclusión final

Como conclusión final a todo este trabajo, he de destacar la importancia del uso de lenguajes estándar para modelar arquitecturas. Basándome en mi experiencia, creo que éste es un aspecto poco extendido a día de hoy en las compañías españolas y creo que es fundamental, tanto para trabajar sobre los mismos conceptos como para documentar las arquitecturas de una organización. En este sentido, como arquitectos de software, somos responsables de utilizar los lenguajes de modelado estándar apropiados para cada situación.

Por otra parte, es importante que comente la facilidad de uso de la herramienta utilizada para este proyecto, Archi, completamente recomendable para esta labor, me ha sorprendido gratamente.

Poner especial hincapié en el uso de vistas completas pero no complejas. Es importante el matiz, completas en el sentido de modelar partes o módulos de la arquitectura pero sin llegar a una extrema complejidad. Es primordial esto último, ya que será clave para un futuro mantenimiento y entendimiento de la arquitectura que se ha diseñando. Éste ha sido un error importante que he cometido en algún punto del modelado de esta arquitectura, y por ello quiero dejar constancia.

He de comentar que, aunque no sigáis completamente las recomendaciones del estándar en lo referente a puntos de vista, sí lo tengáis presente a lo largo de vuestro diseño. En mi caso, he decidido avanzar sin prestar demasiada atención al estándar y, en un punto intermedio de madurez, hacer una recapitulación y comparativa. En este sentido, he podido observar que no he tenido una desviación excesiva entre la información plasmada en mi arquitectura y la propuesta por la documentación de referencia. Pero este aspecto dependerá de vosotros, de vuestra experiencia, de vuestro criterio, de vuestras necesidades y de qué vistas son las que os requieran desde vuestra organización. Es importante que, si no disponéis de requerimientos e información en este sentido, os baséis en el estándar ArchiMate 2.1.

Aunque en este documento no se ha abordado, comentar la importancia de tener otra visión de los diferentes procesos de negocio detectados en la arquitectura. Sobre este aspecto, recomendar el modelado de procesos de negocio mediante el estándar BPMN2. Actualmente, estoy trabajando en otro documento en el que se mostrarán y comentarán algunos de los procesos de negocio de esta arquitectura modelados mediante BPMN2. Espero poder publicarlo próximamente.