Una Técnica para la Construcción de Casos de Prueba Funcionales Basados en Casos de Uso en el Contexto de SPL en Pequeñas Organizaciones – ttSPL

María Del Mar Granda Escobar.
Elkin Francisco Hurtado Hurtado.

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Línea de Investigación en Ingeniería de Software
Departamento de Sistemas
Grupo IDIS - Investigación y Desarrollo en Ingeniería de Software
Popayán, Mayo de 2017

Una Técnica para la Construcción de Casos de Prueba Funcionales Basados en Casos de Uso en el Contexto de SPL en Pequeñas Organizaciones – ttSPL

UNIVERSIDAD DEL CAUCA

María Del Mar Granda Escobar.
Elkin Francisco Hurtado Hurtado.

Trabajo de grado para optar al título de Ingenieros de Sistemas.

Directora: Ma.C. Marta Cecilia Camacho Ojeda.
Codirector: PhD. Julio Ariel Hurtado Alegría.

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Línea de Investigación en Ingeniería de Software
Departamento de Sistemas
Grupo IDIS - Investigación y Desarrollo en Ingeniería de Software
Popayán, Mayo de 2017

Agradecimientos, María Del Mar Granda Escobar

Me gustaría que estas líneas sirvieran para expresar mi más profundo y sincero agradecimiento, a todas aquellas personas que con su ayuda han colaborado en la realización de este trabajo.

Doy infinitas gracias a Dios por permitirme culminar esta última etapa de mi carrera como futura Ingeniera de Sistemas, por su amor, misericordia y fortaleza para seguir adelante, y ser Él quien guíe mi camino para lograr mis sueños.

Un agradecimiento muy especial merece la comprensión, paciencia, apoyo incondicional y el ánimo recibido de mis padres, José Eduardo Granda, Carmen Rosa Escobar, mis hermanas y sobrinos, ya que sin ellos no hubiera sido posible este gran logro.

Mi gratitud a la Ma.C. Marta Cecilia Camacho, directora de esta investigación, por su orientación, seguimiento y supervisión continúa de la misma. Quién con sus conocimientos y experiencias permitió llevar a cabo este trabajo de grado.

Un especial reconocimiento merece el interés mostrado por mi trabajo y las sugerencias recibidas por el PhD. Julio Ariel Hurtado, codirector de esta investigación, pero sobre todo por la motivación y apoyo recibido a lo largo de estos años.

Quisiera hacer extensiva mi gratitud a mis compañeros del Departamento de Sistemas, especialmente al equipo de Investigación y Desarrollo en Ingeniería de Software (IDIS) de la Universidad por su colaboración.

A todos ellos, muchas gracias.

Agradecimientos, Elkin Francisco Hurtado Hurtado.

Quiero agradecer a todas aquellas personas, amigos, compañeros y profesores que de alguna u otra forma hicieron parte del recorrido hacia este trabajo final.

Un Especial agradecimiento a nuestra directora Marta Cecilia Camacho y codirector Julio Ariel Hurtado, quienes aportaron su tiempo y conocimiento para que éste trabajo fuese posible, a mi compañera María del Mar Granda, por el esfuerzo, dedicación y paciencia, indispensable para un buen trabajo y a mi madre que me enseño el valor de la libertad a la libertad porque me enseñó a vivir y a la vida por permitirme llegar hasta este punto.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

CAPÍTULO I

INTRODUCCIÓN

1.1. Motivación

Los procesos de software se han convertido en una de las piedras angulares para fortalecer la industria de software, por lo que es necesario contar con un proceso reutilizable a cada proyecto, con el fin de lograr una producción sistemática del software [1]. Sin embargo, la especialidad que una empresa de software va adquiriendo alrededor de un dominio o mercado, hace que sea reutilizable más allá del proceso, es decir, los mismos artefactos se convierten en activos empresariales potencialmente reutilizables de proyecto a proyecto, generando ahorros significativos de tiempos de desarrollo, esfuerzos y costos [2]. Para ello, las empresas deben definir estrategias que le permitan desarrollar, reutilizar y mantener estos activos de software [3], entre otros, la arquitectura de software, los componentes, patrones, marcos de trabajo, diseños, planes de proyecto, las pruebas, demás documentos y código de software [4]. En particular, las pequeñas organizaciones, ya que su funcionamiento varía de acuerdo algunos factores como: el tamaño, el sector del mercado, el tiempo en el negocio y la gama de productos que desarrollan. Además, estos factores permiten que estas organizaciones tengan la capacidad de adoptar cualquier proceso o mejora de procesos de software, que facilite la gestión de sus proyectos [5].

Uno de los enfoques de reutilización de activos, es el de Líneas de Productos Software (Software Product Lines - SPL), en el cual se aplica una estrategia de reutilización planificada y orientada a los diferentes artefactos de desarrollo de software [6]. Este enfoque se basa en un modelo de fabricación en que una familia de productos se construye bajo el concepto de producción en línea, cuyos productos miembros son susceptibles de compartir una arquitectura y de ser construidos desde un mismo conjunto de activos [6]. Una familia de productos, se refiere a un conjunto de productos que pertenecen a un mismo dominio o mercado, y la producción en línea se refiere a una estrategia planificada (manual, automática o semiautomática) para derivar los productos que pertenecen a la familia a partir de los activos reutilizables [3][6][7].

Sin embargo, planificar la reutilización en el enfoque SPL es una actividad compleja, debido a la cantidad de combinaciones de activos requeridos para derivar los productos de software [7]. Por ello, requiere identificar y definir puntos comunes y variables en los diferentes activos de software, desarrollar los activos de acuerdo a estas posibilidades de variación (elementos comunes y variantes) y detallar planes de producción a partir de esta infraestructura reutilizable en la construcción de productos derivados [6]. Este aspecto de la complejidad, es compensado a largo plazo en un aumento de la productividad, reducción de costos de producción y mantenimiento, así como el tiempo de salida al mercado [2][7][9].

Adicionalmente, a esta complejidad general, cada disciplina del proceso debe abordar complejidades, este es el caso de la disciplina de pruebas, en donde el enfoque de reutilización, se enfrenta al doble reto de planear, diseñar y ejecutar un proceso especializado de pruebas, que le permita controlar la calidad de sus activos reutilizables

(artefactos), siendo los mismos artefactos de prueba, activos reutilizables [3]. Por lo tanto, el proceso de pruebas debe administrar adecuadamente los activos bajo prueba, teniendo en cuenta su variabilidad, que deben ser identificada, definida y administrada [2][5].

1.2. Contexto del Problema

El desarrollo de software incluye algunas actividades específicas que se ocupan de validar y verificar la calidad de los productos software [4], una de esas actividades fundamentales son las pruebas de software, las cuales deben ser planeadas, diseñadas, ejecutadas y reportadas [10]. Normalmente, en el diseño de las pruebas, se incorporan casos de prueba y datos de prueba, que se obtienen a partir de los requerimientos del producto software a desarrollar [11]. Sin embargo, esta actividad es compleja y propensa a errores [12]. Este escenario es más complejo aun cuando se está realizando mantención o cuando se prueban piezas que serán altamente reutilizadas [3]. Una estrategia, es poder reutilizar las pruebas de tal manera que también el proceso de prueba se vea beneficiado en términos de esfuerzo y productividad [7].

Elegir estrategias que soporten el proceso de pruebas, es una de las tareas más importantes dentro de la planificación de una organización desarrolladora de software [10], para verificar la coherencia de los requisitos y el diseño [5]. Sin embargo, los procesos, técnicas y herramientas de soporte en su mayoría son desarrolladas para una función específica (sistemas individuales) y se encuentran aisladas de una estructura reutilizable, dispuesta a una integración, evolución y mantenimiento, además de tener la capacidad de mantener la trazabilidad entre un conjunto variable de artefactos software [7].

El hecho de construir nuevos productos software a partir de activos existentes, busca reducir y evitar la propagación de errores en los productos software derivados [7][13]. Por tanto, es preciso, analizar la capacidad de las pruebas de software de forma tal que resuelvan distintos escenarios de pruebas identificados y diseñados para una familia de productos [6], y obtener cierto nivel de fiabilidad para cada componente reutilizable [7]. Por consiguiente, es vital determinar cómo probar una SPL, examinando y aprovechando la variabilidad presente en los activos, para que las pruebas de software tengan los mismos beneficios del desarrollo de una SPL. La variabilidad es una meta que se alcanza a través del nivel de abstracción de todas las instancias relacionadas con el conjunto de los productos que pertenecen a la línea de productos [7]. Este nivel de abstracción determina la granularidad de la descripción de las diferentes características que reflejan el comportamiento de un producto software [14].

En la construcción de una SPL, es necesario considerar un proceso de pruebas, que pueda incluir pruebas de dominio (ingeniería de dominio), para el conjunto de activos reutilizables, como también pruebas del producto (ingeniería de aplicación), para cada producto específico de la SPL [8]. Siendo las pruebas de software, un desafío determinado por la combinación e integración de especificaciones de casos de pruebas con la ingeniería de SPL [4], especialmente con la arquitectura de los productos reutilizables que configuran y soportan el proceso de pruebas y la variabilidad en el mercado de la SPL. Teniendo en cuenta que los activos comunes y variables de la SPL, afectan la planificación de las

pruebas, en la creación y administración de instancias de casos de prueba para un producto específico de un cliente. Las pruebas, identifican las condiciones del entorno y los parámetros relevantes que debe contener la especificación de los casos de prueba, siendo necesario asociar las etiquetas de variabilidad, que permiten derivar la lista y/o combinaciones de los escenarios de prueba para todos los productos de software reutilizables [7]. Normalmente, se utiliza la notación de casos de uso (base), para desarrollar los casos de pruebas (base y de aplicación), debido a que los casos de pruebas son extraídos del subflujo de los casos de uso, que son los encargados de especificar las funcionalidades y todos los escenarios posibles para un producto [7][10].

Todas las pequeñas organizaciones de software (menos de 25 empleados [12]), no funcionan de la misma manera [15][16] y varían según factores como: el tamaño, sector del mercado, tiempo en la industria, estilo de gestión, gama de productos y localización geográfica [15]. Por tanto, requieren implementar actividades de apoyo y mecanismos operativos [16], que adopten prácticas eficientes de ingeniería de software, y que permitan a estas organizaciones desarrollar productos significativos [10][17], en el marco de su tamaño y tipo de negocio [10]. Para ello, cabe señalar a la variabilidad, que es clave en la ingeniería de requisitos y es una de las tareas esenciales durante todo el proceso de desarrollo de la SPL y las pruebas de software [13]. Particularmente, en una SPL, el tema de la variabilidad en artefactos software a nivel de pruebas, se presenta por la existencia de dos dimensiones; una de activos reutilizables donde los casos de prueba del dominio se derivan a partir de los casos de uso del dominio, y por los casos de prueba de aplicación (a nivel de producto) que se derivan de los casos de uso de aplicación, pero reutilizando los casos de prueba del dominio. Donde, las características especiales de una pequeña organización, hacen que los procesos de pruebas deban aplicarse de un modo particular y visible a como se hace en las grandes organizaciones [10]. Por lo tanto, se requiere implementar e incorporar actividades de pruebas que sean fácilmente aplicables en pequeñas organizaciones y que permitan desarrollar productos de calidad [10]. Considerando las complejidades que las pruebas de software presentan en el desarrollo de una SPL en una pequeña organización, el presente proyecto plantea la siguiente pregunta de investigación: **¿Cómo facilitar la derivación de los casos de prueba específicos de un producto a partir de los casos de uso y casos de prueba de dominio en el contexto de SPL en una pequeña organización de software?**

1.3. Objetivos

1.3.1. Objetivo General

Proponer una técnica de refinamiento y derivación que permitan relacionar en forma consistente los casos de uso y los casos de prueba considerando las dos dimensiones de desarrollo de una SPL (Ingeniería de dominio e ingeniería de aplicación) en pequeñas organizaciones de software.

1.3.2. Objetivos Específicos

- Catalogar métodos y técnicas de pruebas de software, que han sido aplicados en SPL.
- Diseñar una técnica que incluya un conjunto de reglas de refinamiento y derivación, así como notaciones/etiquetas que permitan especificar y modelar la variabilidad de los casos de pruebas.
- Validar la aplicabilidad y consistencia de la técnica a través de un estudio de caso en el contexto de una pequeña organización.

1.4. Metodología de Investigación Utilizada

Para alcanzar los objetivos propuestos, se ejecutará el método de Mario Bunge [18] para el desarrollo del proyecto y el planteamiento del problema del proceso SPL, la propuesta (técnica) siguiendo el enfoque de ingeniería de método basada en ensamble de fragmentos de método y en patrones de método [17], y la exploración y la evaluación del proceso sigue el método de estudio de caso propuesto por Runeson y Höst [19]. A continuación, se describen las fases que se siguieron en el desarrollo del proyecto:

Fase 0: Formulación Preliminar

- **Planteamiento del problema y construcción del modelo teórico**
  - Reconocimiento de los hechos, descubrimiento del problema y formulación del problema: Estudio del referente teórico de SPL, métodos y técnicas; y formulación de una pregunta de investigación o hipótesis.

Producto: Anteproyecto para trabajo de grado

Fase 1: Formulación Preliminar

- **Estudio de viabilidad**
  - Identificación de los factores que soportan las pruebas de software en el contexto de una SPL, y que generen resultados en nuestro proyecto.
- **Exploración del enfoque SPL**
  - Estudio de algunos métodos y técnicas utilizados en la construcción de casos de pruebas en el contexto de SPL.
  - Examinar las técnicas que han adaptado las pequeñas organizaciones de software para planear, diseñar y ejecutar las pruebas de software, en el contexto SPL.
  - Determinar las técnicas o formas básicas de cómo las organizaciones construyen casos de pruebas derivados.

Producto: Análisis comparativo de la construcción de casos de pruebas en el contexto de SPL, en organizaciones de software.

Fase 2: Formulación Inicial

- **Ensamble de la técnica para la construcción de casos de pruebas funcionales en SPL**
  - Inclusión de factores identificados en el estudio de viabilidad mediante puntos de adaptabilidad del proceso.

- Definición de una técnica que permita construir casos de prueba funcionales basados en casos de uso en el contexto de una SPL en pequeñas organizaciones de software.

- **Ensamble de las técnicas particulares (fragmentos de la técnica)**
  - Evaluación de la técnica adoptada en la construcción de casos de uso funcionales, mediante un método de investigación empírico de caso de estudio al interior de una pequeña organización de software.

Producto: Una técnica asociada para construir casos de pruebas funcionales en el contexto de SPL y un reporte de un estudio de caso exploratorio.

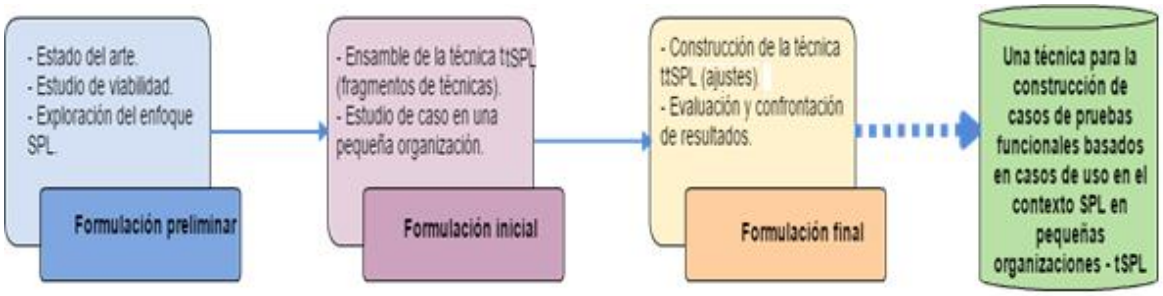
Fase 3: Formulación Final

- **Construcción de la técnica**
  - Ajustes a la técnica para construir casos de pruebas funcionales en el contexto de SPL, según los hallazgos obtenidos en el estudio de caso realizado en el interior de la pequeña organización de software.
- **Evaluación de la técnica**
  - Analizar la aplicabilidad y consistencia de la técnica para construcción de casos de pruebas funcionales basados en casos de uso en el contexto de SPL en pequeñas organizaciones de software. Donde la aplicabilidad está determinada por la compresión y el esfuerzo justificable para la estrategia de reutilización y la consistencia con los posibles errores en los casos de pruebas derivados.
- **Confrontación de los resultados**
  - Evaluación de la técnica con diferentes unidades de análisis para nuevos resultados.
- **Ajustes finales**
  - Análisis de resultados y ajustes técnicos en la construcción de casos de pruebas funcionales basados en casos de uso en el contexto de SPL en una pequeña organización de software.
  - Documentación y elaboración de la monografía del trabajo de grado en la que se consolida el proceso realizado y los resultados obtenidos.

Productos: Monografía, un artículo de investigación, reporte técnico del estudio de caso y una técnica para la construcción de casos de pruebas Funcionales basados en casos de Uso en el contexto de SPL en pequeñas organizaciones de software.

La figura 1, presenta las fases seguidas en el proceso, con los aspectos claves de cada una de las fases.

Figura 1. Metodología de investigación utilizada.



1.5. Estructura del Documento

A continuación, se describe la forma en la que se encuentra organizado el trabajo desarrollado en el presente documento:

Capítulo II – Marco Teórico y Estado del Arte: se presentan los factores relacionados con SPL, pruebas, técnicas y métodos existentes y utilizados en líneas de producto. También, se describe como las organizaciones de software adoptan el enfoque SPL y derivan los casos de pruebas funcionales. Además, se presenta el estado del arte, en donde se encuentran los trabajos relevantes para el desarrollo del presente trabajo.

Capítulo III - Técnica para la construcción de casos de pruebas funcionales basados en casos de uso en el contexto SPL en organizaciones de software - ttSPL: se construye o se ensambla la técnica propuesta, incorporando descripciones, modelos, actividades, tareas y productos de trabajo.

Capítulo IV - Evaluación de la técnica para la construcción de casos de pruebas funcionales: evidencia la aplicación y consistencia de la técnica propuesta mediante el método de caso de estudio en un entorno de trabajo real.

Capítulo V – Conclusiones, limitaciones y Trabajo Futuro.

CAPÍTULO II

2. MARCO TEÓRICO Y ESTADO DEL ARTE

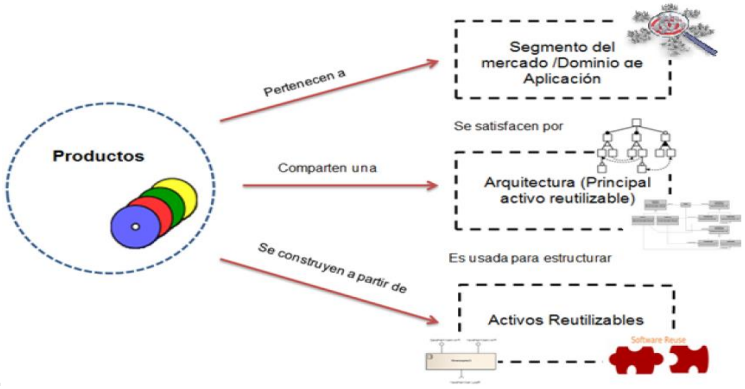
2.2. Marco teórico

En esta sección se presenta los aspectos teóricos de diversos referentes, su concepto y uso, además de otras de definiciones teóricas fundamentales para el desarrollo del trabajo, con el fin de dar claridad sobre las diferentes temáticas tratadas.

2.2.1. Líneas de Productos Software

Una SPL, consiste en un conjunto de elementos base para producir un conjunto de aplicaciones software que comparten características comunes (llamadas similitudes o comunes), pero al mismo tiempo mantienen sus características propias (llamada variabilidad) [3][7]. Una SPL pretende analizar un segmento de mercado en un segmento particular, partiendo de activos comunes (o reutilizables) que desarrollan la arquitectura de nuevos productos potenciales en el mercado, basados en características o requisitos que demanda un cliente. En la figura 2, se representa gráficamente el concepto de SPL.

Figura 2. Representación gráfica del concepto de SPL[1].



La ingeniería de SPL, se compone de tres fases esenciales, ingeniería de dominio, ingeniería de aplicaciones y gestión de SPL [4][7]. El análisis y el diseño de la arquitectura de los artefactos reutilizables (o core asset development), son considerados en el proceso de ingeniería de dominio. En este proceso la organización desarrolla un conjunto de activos centrales (o core assets) que forman una arquitectura genérica para una SPL, que incluye requisitos, diseño, código y pruebas [14], en otras palabras, comprende el análisis del dominio del mercado tanto para el conjunto de productos de la SPL, especificando el alcance (requisitos), la variabilidad y el plan de producción para derivar productos en forma sistemática [13]. En los procesos de ingeniería de aplicaciones, se aplica el plan de producción para desarrollar los productos a partir de la reutilización de activos. El proceso

[1] Tomada de [2].

gestión de SPL, es responsable de mantener y evolucionar la arquitectura de los activos reutilizables paralelo a la ejecución de los otros dos procesos [4].

Un concepto esencial en el desarrollo de una SPL, es la reutilización, un término relevante en la ingeniería de software, que hace uso de un conjunto de procesos reutilizables, que se pueden combinar, con el fin de implementar nuevos sistemas a partir de uno existente, permitiendo ahorrar costos y esfuerzos [10]. Sin embargo, se han adoptado tres enfoques para la reutilización en SPL como son: (i) el enfoque proactivo, que es similar al enfoque en cascada de ingeniería de software convencional, donde se analiza las variaciones del producto software, se diseñan y se implementan aquellas variaciones a través del ciclo de desarrollo, (ii) el enfoque reactivo, que es similar al espiral o programación extrema (XP), en el que se analizan, se diseñan y se implementan las variaciones de uno o varios productos en cada desarrollo de la espiral y de allí se generaliza y construye para la reutilización, y (iii) el enfoque extractivo, que reutiliza la base inicial de los productos existentes, para desarrollar una SPL y derivar nuevos productos software [3].

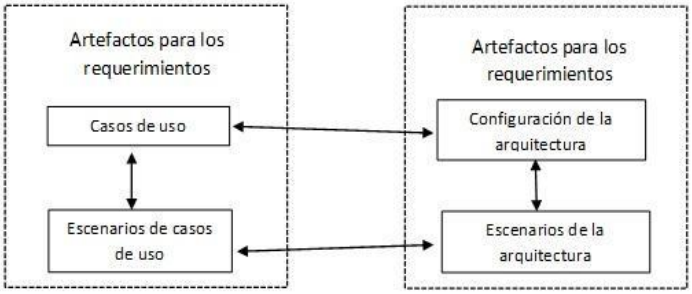
Las características (“features” en inglés) de una SPL, se determinan de las particularidades de un producto y de la identificación de un conjunto de características de productos viables para el mercado y para diversas partes interesadas, como desarrolladores y expertos en el producto, de esta manera representan requisitos funcionales y no funcionales que definen el alcance de la SPL [6][7]. Por ende, un producto de una SPL está desarrollado a partir de las combinaciones de sus características que pueden ser, (i) las características obligatorias, que son comunes para todos los productos de una SPL, (ii) las características opcionales, que representan las particularidades de un dominio dado y (iii) las características alternativas, que permiten la elección de una característica desde un conjunto de las mismas [4]. Facilitando, la posibilidad y habilidad de cambio o de personalización de un producto de la SPL, es decir, poder administrar la variabilidad dentro de la línea de productos [20]. A causa de que la variabilidad en una SPL se enfrenta a dos retos esenciales que son: la representación de las características comunes y variables de la SPL y la construcción de nuevos productos que contengan tanto características comunes, como un subconjunto de las características variables, a partir de la reutilización de componente que se conocen como activos [21]. Además, cabe mencionar que la variabilidad se divide en dos grupos que como: (i) los productos personalizables, donde interviene el usuario seleccionando las características de su producto; y (ii) la línea de productos, donde un conjunto de productos prácticamente similares se unen para reutilizar la parte común de cada uno [20].

2.2.2. Ingeniería de Requisitos en Líneas de Productos Software

La parte más importante para la construcción de software, son los requisitos, pues definen las funcionalidades, restricciones y responsabilidades (actores) que deben ser diseñadas e implementadas para alcanzar los objetivos del producto requerido [22]. En la especificación de los requisitos, se delimita el alcance y se describen las características de los activos core de la SPL [23]. En la mayoría de los casos, en la ingeniería de dominio, se deben definir los requisitos del dominio (requisitos comunes y variables de la SPL) [24]. Uno de los artefactos más utilizados para capturar mecanismos de variabilidad, son los casos de uso [25]. Por

eso, serán la herramienta principal que facilitara la construcción de casos de pruebas funcionales en el contexto de SPL, para nuestra investigación. En la figura 3, se presenta un modelo de información basado en la propuesta [24], para el manejo de requisitos en una técnica para la construcción de casos de pruebas funcionales basados en caso de uso en el contexto de pequeñas organizaciones - ttSPL.

Figura 3. Modelo de información para los requisitos de ttSPL[2].



La ingeniería de requisitos, ha sido reconocida como un elemento verdaderamente crítico dentro del desarrollo de software, debido a su criticidad y que requiere un esfuerzo significativo [22]. En el desarrollo de una SPL, dicha complejidad aumenta debido a las diferentes partes interesadas, el aumento de los productos y la atención requerida por la variabilidad y la reutilización [26][27]. Teniendo como actividades esenciales: (i) la elicitación, la cual se centra en la definición del alcance y la captura de variaciones previstas explícitamente durante la vida útil de la SPL, involucrando mayoritariamente las partes interesadas [26], (ii) el análisis y negociación, que consiste en encontrar las variaciones, similitudes, analizar el impacto de los nuevos requisitos en la arquitectura de la SPL, identificar oportunidades para la reutilización y resolver los conflictos no solo desde el punto de vista lógico sino también desde el económico y de mercadeo, (iii) la especificación, consiste en documentar un grupo de requerimientos a nivel de línea de producto y requerimientos específicos del producto, (iv) la validación, consiste en comprobar la coherencia, la integridad y la exactitud de los requisitos comunes y variables [26][28][29]; y (v) la gestión, consiste en apoyar la evaluación sistemática de cómo los cambios propuestos afectaran la SPL y verificarán los vínculos de trazabilidad entre los requisitos y sus principales activos asociados [27].

La reutilización requiere trazabilidad, entre los componentes o activos de software que forman un producto [30]. La trazabilidad de requisitos, es un factor reconocido en la calidad de los procesos de desarrollo de software [31]. Cuando los requerimientos se gestionan bien, la trazabilidad puede establecerse desde el requerimiento base hasta sus requerimientos de más bajo nivel. Particularmente, cuando se lleva a cabo la evaluación del impacto de los cambios de los requerimientos sobre las actividades y los artefactos de un proyecto. Además, es útil para la consistencia entre los diferentes modelos, etapas del desarrollo y mantenimiento de las líneas de productos [31].

[2] Basada en [24].

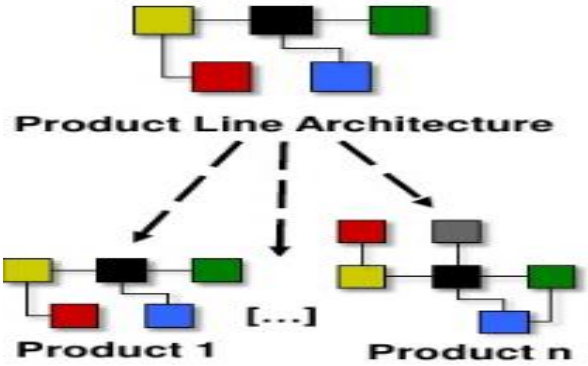
Realizando una revisión de la literatura acerca de los métodos para ingeniería de requisitos dentro del contexto SPL se tienen diferentes enfoques: (i) basado en puntos de vista: es un mecanismo explícito que tiene en cuenta los problemas del sistema y diferentes perspectivas de los grupos de interés [27], (ii) basados en objetivos: el objetivo significa la dirección, propósito y objetivo de la organización, (iii) escenarios basados en objetivos, consistente en la creación de escenarios que muestran el comportamiento del sistema [32], y (iv) basado en casos de uso, el cual cuenta con un modelo basado en UML como el expuesto por Gomma en [33].

Un problema en la ingeniería de requisitos es que los requisitos son continuamente cambiantes, por eso, es imposible capturar todos los requisitos para un sistema en una etapa inicial del desarrollo [9]. Adicionalmente, los requisitos en SPL, abarcan varios productos, es decir, ciertos requisitos deben escribirse como puntos de variación, con el fin de capturar variaciones entre productos individuales de la línea de productos [34]. Por tanto, es necesario identificar los puntos donde se diferencian los productos, los cuales se conocen como puntos de variación y cada implementación de los puntos de variación se denomina variante [2][35].

2.2.3. Manejo de la Variabilidad en una SPL

Indicar las características comunes y variables entre productos de software de una SPL, se conoce como "*gestión de la variabilidad*". Donde el propósito es construir un modelo con las distintas características y relaciones de la línea de productos, considerando la variabilidad desde la especificación de requisitos hasta la ejecución de las pruebas de software [6].

Figura 4. Representación de la arquitectura una SPL.



La especificación de cada producto de software en una SPL, se realiza a través de características, que indican un incremento en la funcionalidad. Pues, un modelo de características agrupa la parte común y variable de una línea de productos, siendo útil para construir la gran mayoría de producto de la SPL [36]. Es importante, resaltar la relación que existe entre características y requisitos. Porque, un requisito se puede aplicar a un conjunto de características funcionales, y de la misma forma una característica puede satisfacer más de un requisito. Dentro de los tipos de características para una SPL, se encuentran las

características externas (no forma parte directamente del producto), características obligatorias (identifican o define un activo core), características opcionales (añaden valor a las características externas y obligatorias de un producto) y las características variantes (son una abstracción de un conjunto de características que relaciona las características opcionales, obligatorias o externas). Que al utilizarlas es posible hacer que la SPL, sea lo suficientemente flexible para ajustarse a un nuevo requisito [37].

La identificación de la variabilidad, en la fase inicial del desarrollo de una SPL, admite que se puedan especificar los requisitos que construyen una serie de productos y también los requisitos que serán incorporados en productos futuros. Pues, la funcionalidad de la variabilidad es unir de alguna manera los requisitos que definen la SPL [37]. Es así, que la variabilidad se define como: "*la capacidad de cambiar o de personalizar un producto*" [37][30]. La metodología más utilizada para analizar la variabilidad basada en características es el "*Análisis del Dominio Orientado a las Características*" (Feature Oriented Domain Analysis - FODA), el cual, se basa en identificar las características que los usuarios esperan que contenga el producto de software, que corresponde a un dominio de la SPL, también define las etapas del método y los resultados obtenidos en cada una de las etapas, soportando el descubrimiento, análisis y la documentación de los aspectos comunes y diferentes de la ingeniería de dominio [31].

En el contexto de línea de productos, existen varios niveles para diseñar la variabilidad como: a nivel de línea de productos, que establece cómo diferentes productos de la línea varían, a través de la selección de componentes que construyen diferentes productos, a nivel de producto, que define la arquitectura y la selección de componentes para construir un producto en particular, a nivel de componentes, que especifica la implementación de los componentes seleccionados para un producto y a nivel de subcomponente, que puntualiza las características para formar un componente de un producto en particular [38].

González-Baixauli, Laguna, Sampaio do Prado Leite, exponen en su propuesta [20], la existencia de otros enfoques para representar la variabilidad, según el criterio de otros autores. Inicialmente, nombran a Jacobson, que utiliza casos de uso tradicionales y por medio de la relación <<extends>> expresan gráficamente la variabilidad, enmarcada por el tipo (múltiple u opcional) y la cardinalidad, definiendo de esta manera los puntos de variación. Seguidamente y finalmente, nombran a Halmans y Pohl, quienes extienden la notación de casos de uso para expresar gráficamente los puntos de variación. En ambos casos las técnicas fueron desarrolladas para la especificación de requisitos desde la perspectiva del cliente. Pero presentan inconvenientes en la representación global de la variabilidad. Ya que representan por separado la variabilidad, es decir, construyen múltiples casos de uso según la existencia de puntos variables, y no globalmente como lo haría un modelo de características.

2.2.3.1. Modelo de Características

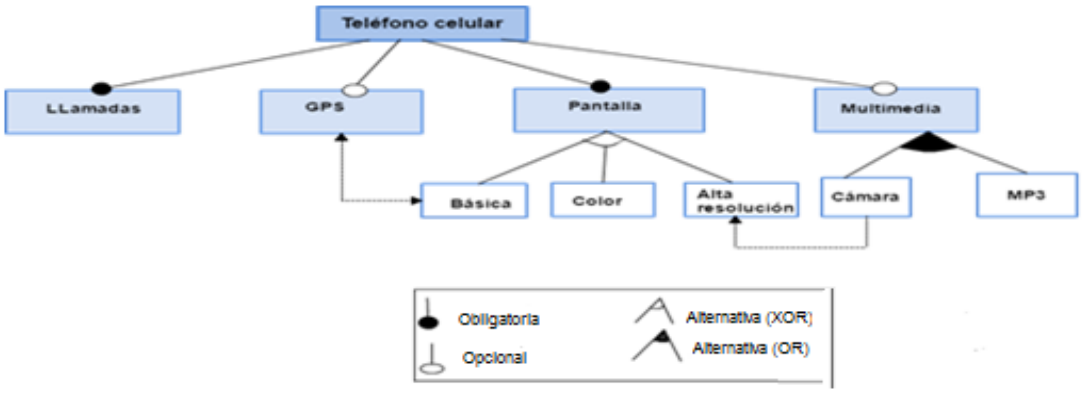
Usualmente, un modelo de características es utilizado para representar una SPL [39], pues delimita los elementos (comunes y variables) de la línea, a través de las features, estructurándolas en: forma de diagramas jerárquicos And/Or [20], con relaciones que

requiere/incluye, excluye [34] obligatorias, opcionales y/o alternativas [4]. Donde AND representa la parte común, OR la parte variable [20], una relación requiere indica que la función es dependientemente de otra función para poder existir y una relación excluyente entre dos características, indica que características no se pueden incluir en el mismo producto [34]. Aunque, no son un procedimiento de especificación de requisitos [20]. Por tal razón, nuestra propuesta utilizará casos de uso para la elicitación de requisitos funcionales de software y los modelos de características para presentar la variabilidad de los productos de la SPL.

Un modelo basado en características facilita el desarrollo, la parametrización y la configuración de activos reutilizables [40]. Van Der Vegt, Westera, Nyamsuren Georgiev y Ortiz en [41], construyen una técnica que define la arquitectura de activos reutilizables, llamada *RAGE,* la cual se basa en un modelo de componentes, dirigida a la construcción de la estructura del funcionamiento de los activos que el cliente solicite, además requiere el cumplimiento de requerimientos de calidad básicos como: la interoperabilidad, la nomenclatura, la extensibilidad, abordar las dependencias de la plataforma y del hardware, la portabilidad, el aseguramiento de calidad y simplicidad general. Todos estos aspectos son aplicados y evaluados en un juego serio.

Eriksson y Börstler [42] describen un enfoque que denominan "*Modelado de Casos de Uso de la Línea de Productos para Ingeniería de Sistemas de Software*" (Product Line Use Case Modeling for Systems and Software Engineering - PLUSS), este integra un modelo de casos de uso y un modelo de características, para describir y mantener un modelo de casos de usos de una línea de productos. Considerando, que el modelado de casos de uso tradicional no apoya ni permite la variabilidad. Por tal razón, desarrollan un modelo orientado hacia líneas de productos, que contiene el dominio de la SPL y usa el modelo de características FODA, para plasmar una visión general de las características y las interrelaciones entre los productos. En la técnica FODA las características del producto son abstracciones tanto para los desarrolladores como para el cliente [40].

Figura 5. Ejemplo de un modelo de características[3].



[3] Tomada de [39].

Algunas investigaciones se han dedicado a desarrollar herramientas (ver tabla 1) para personificar modelos de características e implantar su respectivo análisis. La mayoría coinciden en la creación de modelos, configuraciones gráficas y en la validación de los modelos, pero generalmente estas herramientas presentan carencia en la representación gráfica de un modelo que requiera todos los tipos (obligatoria, opcional, alternativa, O, requiere y exclusión) de características en el mismo modelo [43].

Tabla 1. Herramientas para el modelado de características[4].

Nombre	Descripción	Facilidad de uso	Plataforma	GUI	Licencia
FaMa	Permite restricciones básicas[5]	Media	Librería de Java	Si	Open. source
FeautreIDE	Es un editor gráfico como textual, permite configuraciones y creación de restricciones avanzadas[6].	Media	Plugin Eclipse	Si	Open. source
MFM	Utiliza técnicas basadas en lenguaje de modelado UML.	Alta	Plugin Eclipse	Si	Open. source
S2T2	Es el resultado de una investigación relacionada con SPL. Permite configurar y comprobar las restricciones básicas.	Media	Aplicación Java	-	Open. source
RequiLine	Tiene un editor gráfico basado en la notación FORM. Dispone de un validador de restricciones.	Baja	Aplicación Win	Si	Libre hasta el momento
Fmp	Posee muchas características que funcionan como la creación de configuraciones, atributos y restricciones avanzadas.	Media	Plugin Eclipse	Si	Open. source
SPLOT	Dispone de una base de datos con gran cantidad de modelos de características base. El usuario puede subir su propio modelo de características que debe de ser escrito en XSML.	Alta	Web	-	.
xFeature	Presenta referencias, atributos y una versión no completa de restricciones avanzadas	Baja	Plugin Eclipse	Si	Open. source
Pure::Variants	Permite gestionaras partes de los productos software con sus componentes, restricciones y términos de uso**.**	Media	Plugin Eclipse	Si	Copyright
Captain Feature	Presenta referencias, atributos y una versión no completa de restricciones avanzadas	Media	Aplicación de Java	Si	GNU

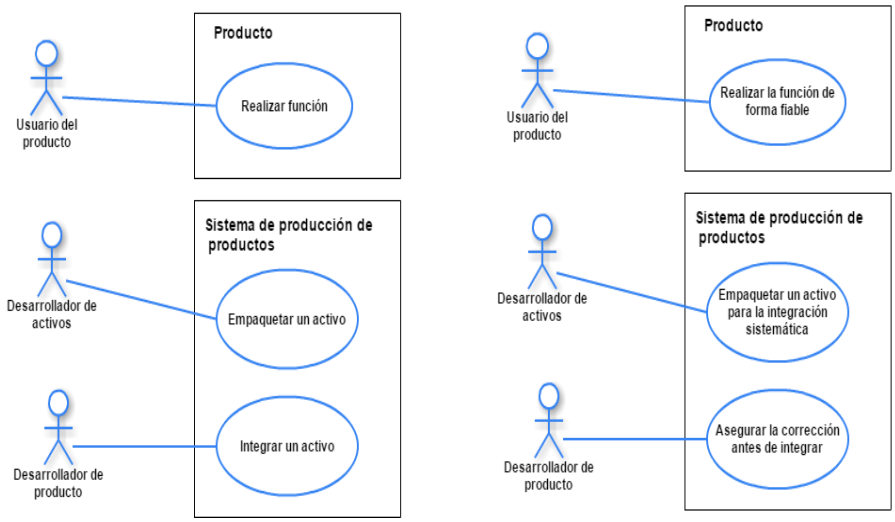
[4] Tomada de [43].
[5] Las restricciones básicas representan las características de inclusión y exclusión.
[6] Las restricciones avanzadas representan las características opcionales, or, obligatorias y alternativas.

2.2.4. Casos de Uso en SPL

Los casos de uso, son utilizados para obtener documentación y comprender las necesidades y requerimientos del cliente y stakeholders, Los casos de uso en SPL son utilizados para modelar los requisitos de una línea de productos, las propuestas presentadas adaptan los modelos y plantillas para poder describir las características comunes como también las variaciones entre productos [44]. Al mismo tiempo, los casos de uso en SPL, pueden ser etiquetados con estereotipos tales como: <<kernel>> para representar las características que son comunes a todos los productos de la SPL, <<opcionales>> para personificar las características que son requeridas únicamente por algunos productos y <<alternativos>> cuando se requieren diferentes versiones de casos de uso para construir diferentes productos [44].

Una de las formas de modelar los requisitos en una SPL, es por medio de un modelo de casos de uso, en esta propuesta se emplea como un punto de partida para derivar los casos de prueba. Pues, los casos de uso son calificados como una poderosa herramienta para capturar requisitos funcionales, que describen la interacción entre un determinado sistemas de software y su entorno desde el punto de vista del usuario [45]. En la figura 6, se presentan una técnica para capturar los requisitos de un producto y las anotaciones para describir acciones asociadas a construir un sistema software, incluyendo una arquitectura (producto y sistema de producción de productos) que proporciona el contexto para manipular activos [46].

Figura 6. Esquema de casos de uso y anotaciones para la calidad[7].



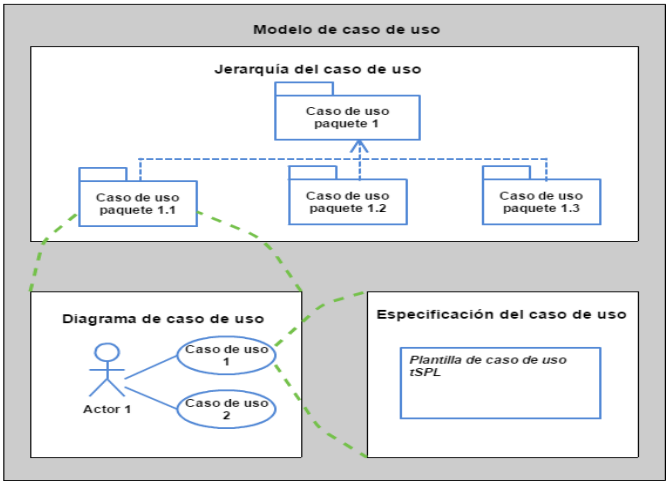
Autores como: Guedes, Silva y Castro [47], Eriksson y Börstler [42], utilizan en sus respectivas propuestas la notación de "*modelado de casos de uso de la línea de productos para ingeniería de sistemas y de software*" (Product Line Use case modeling for Systems and Software engineering - PLUSS), un enfoque para líneas de producto que combina modelos de características y casos de uso, para capturar el comportamiento común y

[7] Tomada de [46].

variable de una SPL, lo que permite para la técnica propuesta calificar a PLUSS como una estrategia inicial para el desarrollo del proyecto.

La figura 7, expone desde el punto de vista general, un modelo de casos de uso, indicando la jerarquía de ciertos paquetes, que contienen o agrupan diagramas de casos de uso para representar la funcionalidad asociada a un producto de la línea de productos.

Figura 7. Descripción general de artefactos y conceptos del modelo de casos de uso[8].



Adicionalmente, en los modelos de casos de uso para SPL, se distinguen cuatro tipos de variación: (i) un caso de uso puede o no incluirse en un determinado producto, (ii) algunos escenarios de casos de usos pueden ser excluidos en la construcción de un producto, (iii) el flujo de eventos que describe un determinado caso de uso puede variar y (iv) la definición de otros aspectos (actores, flujo de eventos, etc) que afectan el comportamiento de múltiples casos de uso. Al seleccionar exclusivamente algunos escenarios (opcional o/y alternativo) o flujo de eventos de un caso de usos, es posible distinguir los elementos arquitectónicos responsables de la funcionalidad de un producto [34]. En PLUSS, todas estas variaciones se organizan en un modelo de features, que representa el conjunto de características de un producto específico de una SPL. También se modelan en un "*caso de cambio*", que permiten a los desarrolladores de la SPL planificar y proyectar los requisitos futuros de un dominio. Asimismo, desarrollar solicitudes de cambio y propuestas de ingeniería de cambio para la línea de productos [35].

2.2.5. Pruebas de Software

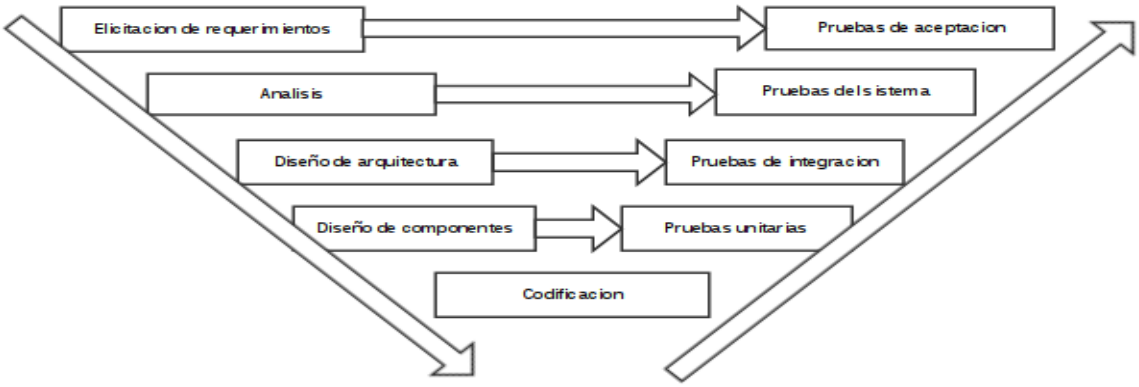
Las pruebas de software, es el proceso de planear la ejecución de un programa con la intención de encontrar, reducir y corregir los fallos relacionados con un producto de software [48]. Normalmente, la planificación de las pruebas se puede ejecutar en cualquier punto del proceso de desarrollo de software, variando el número de etapas de acuerdo al proceso aplicado para probar un producto de software. Se considera, un proceso de pruebas

[8] Tomada de [34].

formalmente compuesto, cuando por lo menos contiene las siguientes etapas: planeación de pruebas, diseño de pruebas, implementación de pruebas, evaluación de criterios de salida y cierre del proceso. Las pruebas se pueden clasificar en cuatro niveles: pruebas unitarias, pruebas de integración, pruebas de sistema y pruebas de aceptación [49]. En cada uno de estos niveles de prueba, se podrán ejecutar diferentes tipos de prueba tales como: pruebas funcionales, no funcionales, de arquitectura y asociadas al cambio de los productos. Existen tres enfoques principales para el diseño de casos de prueba: el enfoque estructural o de caja blanca, se centra en la estructura interna del programa; el enfoque funcional o de caja negra; se centra en las funciones, entradas y salidas; y el enfoque aleatorio, utilizando modelos (en muchas ocasiones estadísticos) que representen las posibles entradas al programa para crear a partir de ellos los casos de prueba [50].

Las pruebas basadas en modelos, son definidas como las pruebas de software donde los casos de pruebas son derivados de un modelo que describe algunos o todos los aspectos del sistema bajo prueba. Se distinguen diferentes procesos de desarrollo de software, en todos estos procesos, el software es desarrollado en fases. La mayoría de los procesos tienen fases similares y difieren principalmente en las condiciones y posibilidades para avanzar hacia la próxima fase o volver a visitar la fase anterior. Uno de estos procesos, el modelo en V presenta una visión integrada de la construcción y de las fases correspondientes de pruebas [51].

Figura 8. Estructura del modelo en V.



El modelo V distingue las fases de construcción y las fases de pruebas. El desarrollo del sistema comienza con la captura y definición de los requisitos. Los requisitos son capturados del cliente y de los futuros usuarios. Estos son usados en la siguiente fase para desarrollar los modelos funcionales del sistema [51].

El modelo muestra diferentes tipos de pruebas. Por un lado, están las pruebas unitarias, que consisten en la prueba de diferentes partes o módulos de software por separado, y simulando de alguna forma los componentes que interactúan con el que se está probando. Las pruebas de integración consisten en, una vez que se han probado los distintos sistemas por separado, se prueban todos juntos, especialmente las interfaces. Tras esas pruebas de integración, se realizan las pruebas del sistema completo y tras las pruebas de sistema van las pruebas de aceptación [52].

2.2.6. Pruebas de Software en Línea de Productos Software

Las pruebas de software para una SPL, abarcan la validación del modelo inicial de requisitos, el diseño y la ejecución de casos de prueba para un producto específico, para el conjunto de productos [30] y sus interacciones [4]. Teniendo en cuenta que es fundamental definir el alcance de la línea a partir del conjunto de productos, pues el alcance limita las posibles variaciones que deben estar representadas en las pruebas, además de la gama de valores que deben considerarse durante su ejecución [35].

La definición del alcance de las pruebas, permite abordar el rango de las variabilidades existentes en la SPL [30]. Pues, es necesario identificar las características que se someterán a prueba [45]. Un alcance demasiado amplio será un desperdicio de recursos de prueba al momento de probar productos que no se construirán [30]. Por ellos, se podría aplicar esta tarea, como una decisión estratégica para reducir esfuerzos de tiempo y desarrollo, que influyen en los costes de la prueba y los recursos necesarios para cumplir el propósito de la prueba [20].

En particular, existen algunas directrices para SPL, donde se involucran las pruebas de los activos fundamentales de un producto específico y su variabilidad [7]. Además, las pruebas en SPL producen artefactos (conjunto de datos de pruebas y documentos tales como planes de pruebas e informe de pruebas) que pueden ser reutilizados en los distintos productos de la línea. Por ello, las pruebas de software deben estructurarse de tal forma que soporten la reutilización de activos de pruebas (activos core) y la funcionalidad de un producto [38]. El tema de las pruebas en líneas de productos será detallado más adelante en los trabajos relacionados.

2.2.7. Pequeñas Organizaciones de Software

Las pequeñas organizaciones de software, representan la mayoría de la industria software en Colombia [12], como también en USA, Brasil, Canadá, China, Finlandia, Irlanda, Hungría y en otros países [10][53]. Además, estas organizaciones compiten a través de factores como la calidad de software, la productividad [53], tiempos de desarrollo y costos [54]. Condicionando, de esta forma el tiempo de entrega y la calidad de los productos software que ofrecen. Por consiguiente, necesitan definir un proceso de pruebas software, que le brinden la capacidad de incrementar la calidad del producto, apoyar la comunicación, ejecución y mejora continua de sus procesos, pero al mismo tiempo dichas actividades deben adaptarse a las necesidades y recursos de la organización [12], siendo flexibles en la configuración de su flujo de trabajo e introduciendo la variabilidad, que expresan las características que construyen un producto software dentro de SPL [55].

Especialmente, este tipo de organizaciones tiene características únicas, que hacen su estilo de negocio diferente a las grandes organizaciones [15]. Por lo cual, la mayoría de los procesos de gestión se llevan a cabo a través de mecanismos informales [15][56], basados en las relaciones cara a cara (comunicación, toma de decisiones, resolución de problemas, etc.) [56] y menos documentado [15]. Además, no tienen suficiente personal para

desarrollar funciones especializadas y cuentan con poco o ningún presupuesto para la adquisición de conocimientos adecuados, puesto que, cada proyecto es nuevo, así que, la creatividad y la flexibilidad son cualidades importantes. También, son económicamente vulnerables y tienen recursos económicos limitados [56].

2.3. Trabajos Relacionados

Las líneas de productos se han convertido en una estrategia valiosa para elevar la competitividad de las empresas de software [2][23]. Sin embargo, la aplicación de dicho enfoque requiere conocimiento, muchos recursos y consume tiempo [2], con mayor razón cuando se contempla el tema de pruebas, pues los activos de la línea tienen la habilidad de ser reutilizados, por tanto, se debe garantizar de cierta forma la calidad de los productos de la SPL. Por ello, es primordial considerar algunos estudios encontrados en el área de investigación de líneas de productos y pruebas, y que son relevantes en la propuesta a realizar.

2.3.1. Pruebas en Líneas de Productos Software

Las pruebas juegan un papel importante en el esfuerzo estratégico de una SPL [35], que encierra conocimientos acerca de los procesos, tecnologías y modelos necesarios para definir los distintos escenarios de pruebas, representados a través de casos de pruebas reutilizables, desarrollados a partir de la abstracción de casos de uso, que son los activos más utilizados para construir casos de pruebas y extraer datos de pruebas [3]. De esta manera construir casos de pruebas en: pruebas unitarias, pruebas de integración, pruebas del sistema y en pruebas de aceptación. Probando cada artefacto diseñado en la ingeniería de dominio y de aplicación, incluyendo los activos de un producto específico junto con sus interrelaciones [3][35], sin perder uno de los principios más importantes del enfoque SPL como es la reutilización de los casos de pruebas asociados a un producto [3]. En el análisis de los puntos comunes o variables, se logra identificar ciertas pautas para los puntos de variación necesarios en la construcción de la arquitectura de la SPL, y así apoyar la gama de requisitos de los productos de una SPL con las variaciones entre productos, lo que significa también, variación de pruebas de software, en especial en los casos de pruebas de software [35] que deben identificar y abordar las variabilidades.

Colanzi, Assunção, de Freitas, Zorzo, Vergilio [3] y Mcgregor [35], identifican atributos de calidad en el juego “Arcade Game Maker, que es una línea de productos desarrollada por el instituto de ingeniería de software (SEI), creada con fines pedagógicos para ilustrar los diferentes temas que abarca el curso que este instituto ofrece acerca de SPL, donde una de las temáticas que abarca es el diseño y aplicación de pruebas a los productos de la línea. Las pruebas de software se basan en combinar cuestiones técnicas de pruebas con la parte de gestión de pruebas de software, para producir activos principales como: un plan de pruebas, pruebas de infraestructura, casos de prueba y datos de pruebas. Para ello, Mcgregor [35], establece el manejo de una “cadena de calidad” (requerimientos, análisis, diseño arquitectónico, diseño detallado y codificación), que enlaza y establecen una variedad de actividades que determinar los artefactos incluidos y dispersos en las pruebas de software. Esta cadena permite mezclar los activos para reducir los problemas de

trazabilidad entre el producto y la prueba de activos.

Colanzi, Assunção, de Freitas, Zorzo y Vergilio [3] compararon tres estrategias de prueba denominadas: (i) estrategia de producto; que se basa en poner a prueba productos de forma individual, (ii) estrategia progresiva; que trata de poner a prueba nuevos productos, reutilizando los casos de pruebas utilizados en productos previamente probados (reutilización de casos de prueba) , y (iii) una estrategia para crear instancias de los datos de pruebas obtenidos en la ingeniería de dominio, teniendo en cuenta aspectos comunes y variabilidades de la SPL (derivación de casos de prueba). Los autores concluyen que la calidad de los productos está determinada por la exactitud y la fiabilidad de los activos y la oportunidad de reutilizar los activos de la SPL [3][35], y que "las pruebas deben estar diseñadas para la portabilidad, mediante el aprovechamiento de los puntos de variación en el software, así como en la arquitectura del sistema" [2].

El aporte principal que ofrecen estos trabajos relacionados es la identificación y definición de ciertas actividades o estrategias, que juegan un papel importante, dentro del desarrollo de pruebas de software para un juego en específico de una SPL. Nuestra propuesta se diferencia de estos trabajos en que se pretende construir un conjunto de reglas de refinamiento y derivación que facilite la construcción de casos de pruebas en el contexto una SPL, para identificar qué características se aplican a un producto de la línea y qué configuración es necesaria para el nuevo producto.

2.3.2. Casos de Uso y Casos de Pruebas en Líneas de Productos software

Bertolino, Fantechi, Gnesi, Lami [7], Bertolino y Gnesi [13][57], centran su propuesta en la planificación de pruebas software, considerando esta la parte más crítica de del proceso de pruebas y en especial en el contexto de cómo validar una SPL [58], ellos presentan una metodología desarrollada para la construcción de pruebas funcionales, llamada Optimización de Pruebas de Casos de Uso en Líneas de Producto (Product Line Use Case Test Optimization – PLUTO) [7][13][57], que se encarga de definir cada paso para derivar un conjunto de pruebas funcionales de acuerdo a las especificaciones de un producto, las cuales se encuentran descritas en lenguaje semi-estructurado [57]. Con lo que se construye, un conjunto de especificaciones de prueba para cada caso de uso relacionado con una SPL [13]. En particular, se busca identificar cómo manejar y representar la variabilidad dentro de una SPL [7][13][57], en el sentido de evitar la ambigüedad, ya que un conjunto de especificaciones debe ser lo más preciso, riguroso y encontrar formas para especificar una SPL [13]. Bertolino y Gnesi [57], definen mecanismos para abstraer y describir escenarios de pruebas, creando escenarios de uso (casos de prueba), los cuales deben ser probados para validar que las necesidades de los usuarios están completas. La derivación de casos de prueba se realiza mediante la combinación de las opciones relevantes desde los Casos de Uso de la Línea de Productos (Product Line Use Case – PLUCs) relacionados. Los casos de prueba se derivan cuando un producto específico está siendo desarrollado después de haber instanciado las etiquetas en cada PLUC a los valores adecuados. En [13], cada especificación de pruebas se relaciona con un PLUC, y un conjunto diferente de casos de prueba, el cual se basa en los requisitos descritos en

lenguaje natural, dando lugar a la derivación de los casos de pruebas que son ejecutado parcialmente de forma manual, dependiendo de los valores de variables de la SPL. Otra característica específica de casos de prueba derivados basados de casos de uso, es la presencia de varios escenarios, es decir, el escenario principal de éxito, los escenarios alternativos y excepcionales. Por supuesto, todos ellos deben tenerse durante la prueba. Pues, la especificación de prueba de PLUCs normalmente incluirá una categoría (escenarios), en el que aparecen el escenario principal y todas las extensiones especificadas [13][57].

El aporte principal que ofrecen estos trabajos relacionados es la posibilidad de adaptar un método ya formalizado, en la construcción de nuestra técnica, con la diferencia de definir aspectos (notaciones/etiquetas) que permitan medir la trazabilidad de los requisitos software y las pruebas del software.

2.3.3. Modelo Basado en Pruebas por Parejas la Cobertura de Características de Interacción en Ingeniería de Líneas de Producto Software

Lochau, Oster, Goltz, y Schu [59], plantean una relación entre modelos de características describiendo las partes comunes y variables de una SPL y un modelo de prueba reutilizable en forma de gráficos de estado, con el fin de modelar el comportamiento de una SPL. Su estudio se basa en modelos, considerando que estos son adecuados para describir la variabilidad de los productos y pueden usarse con fines de implementación y pruebas, debido a la cantidad productos que se pueden desarrollar en una misma línea. Además, utilizan pruebas por parejas para probar un subconjunto de todos aquellos productos posibles de la SPL. Para validar esta propuesta, se realizó un estudio de caso en el sector automovilístico, donde se evaluó un Sistema de Confort Corporal (BCS), que contiene 21 características, relacionadas como las funciones de seguridad. Debido a la creciente cantidad de la variabilidad en el sector automovilístico, esta situación se complica, ya que una sola Unidad de Control Electrónico (ECU) puede ser instanciado en al menos 10.000 formas diferentes. De esta manera, se busca interrelacionar criterios de cobertura, basados en modelos con criterios de pruebas de cobertura basadas en modelos de cobertura de control y flujo de datos, dando la posibilidad de tratar aspectos de pruebas por parejas. Para ellos, se desarrolló un modelo denominado 150%, que incluye puntos variables y puntos en común para derivar casos de prueba para los productos. Un modelo de prueba 150%, contiene toda la información requerida para probar cada posible producto de la SPL y se representa a través de gráficos de estado. Una ventaja importante del uso de grafos de estado es la disponibilidad de una variedad de generación de código y el uso de herramientas para generación automática de casos de prueba, en este estudio se utilizó la herramienta ParTeG. El razonamiento sobre la prueba por parejas en el contexto SPL, es basado en la detección y el análisis de interacciones entre características tanto del punto de vista sintáctico como semántico.

El aporte principal que ofrece este trabajo, es tener otra perspectiva para la generación de escenarios de pruebas, ya que se utiliza la notación de grafos de estado, nuestra propuesta se diferencia de este trabajo en que los casos de pruebas construidos serán a partir de la

notación de casos de uso ya que se ha logrado identificar por los estudios analizados, que son la herramienta software más potente para generar escenarios de casos de pruebas reutilizables y datos de pruebas.

2.3.4. Herramienta de Casos de Prueba del Sistema de Líneas de Producto Software

C. R. L. Neto, et Al. [60], identifican la carencia de herramientas software, que permitan automatizar la producción de casos de prueba en SPL. Por tanto, proponen una herramienta que apoyará las actividades de pruebas a nivel de sistema, con el objetivo de reducir el esfuerzo requerido, además de generar artefactos de pruebas reutilizables, enfocados en la elaboración de casos de pruebas para cada caso de uso que pertenezca a una SPL. Esta herramienta llamada RiPLE-TE, se basa en la definición de un proceso estructurado para ejecutar las pruebas que han sido determinadas para SPL, a través de un plan de pruebas que incluye activos base, activos de productos específicos y sus interacciones, de acuerdo a esta consideración, RiPLE-TE construye los artefactos de prueba en paralelo al desarrollo de los activos del producto. Así, es posible mantener la trazabilidad entre estos artefactos, y en consecuencia, facilitar el proceso de desarrollo y la reconstrucción de ellos, si es necesario. Cada caso de prueba creado en la base del proceso de RiPLE-TE puede representar variabilidad en la forma en que la reutilización es fomentada, describiendo fragmentos del modelo de prueba para ser manipulados, ya que cada caso de uso de una SPL es apoyado por uno o más casos de prueba, donde un caso de uso se puede descomponer de forma recursiva en sub-objetivos, consecuentemente para crear casos de prueba para cada uno de los fragmentos asegurando la cobertura de los casos de uso y considerando la variabilidad para construir el modelo de prueba que incluye en este caso la planificación y la gestión de activos de pruebas. Para poder manejar la variabilidad, todos los puntos de variación están asociados con los requisitos especificados en el documento detallado de requisito de SPL.

Este trabajo resulta importante para nuestra investigación ya que comparte ciertos retos propuestos en nuestro trabajo, como son la gestión de la variabilidad. Enriqueciendo la trazabilidad a partir de los requerimientos para la implementación de los activos de prueba, creando una completa distinción entre los activos pertenecientes a la SPL y los pertenecientes a un producto específico. De otro lado, este trabajo expone en las conclusiones como trabajo futuro, que la propuesta sea validada experimentalmente, para analizar la efectividad de la herramienta. Nuestra propuesta pretende llegar a construcción una técnica para la obtención de casos de prueba funcionales los cuales se ejecutarán de forma manual, con el fin de validar la aplicabilidad y consistencia de la técnica a través de un estudio de caso en el contexto de una pequeña organización. Donde la aplicabilidad está determinada por la compresión y el esfuerzo justificable para la estrategia de reutilización y la consistencia con los posibles errores en los casos de pruebas derivados.

2.3.5. Pruebas en Líneas de Productos Software basado en Especificaciones

Mishra [61] utiliza un proceso de especificación formal para generar los casos de pruebas de una SPL, a partir de la utilización de casos de uso. El documento propone la reutilización de los casos de prueba, que han sido generados por las características comunes a través de la especificación de los axiomas que se van agregando de acuerdo con el refinamiento de los respectivos casos de prueba para enfatizar en la reutilización de los casos de pruebas y datos de pruebas. Para aprovechar los beneficios del enfoque de línea de producto, se sigue una planificación sistemática y la generación del conjunto de pruebas, útil para comprobar el producto derivado de la especificación existente.

En este trabajo, muestra la creación de paquetes de prueba en una SPL, mediante la Especificación Formal del Proceso del Sistema en Lenguaje Algebraico de Especificación (Communicating Sequential Processes – Common Algebraic Specification Language – CSP-CASL), que es un lenguaje para describir sistemas concurrentes. Generalmente los sistemas concurrentes, tienen complejas interacciones entre los procesos que los hacen difíciles de entender, por consiguiente, CSP proporciona el conjunto de símbolos matemáticos para analizar un sistema tan complejo con claridad y precisión que absorbe la mayoría de los enfoques anteriores de especificación algebraica. CSP-CASL es un lenguaje algebraico de especificación de procesos, para la descripción de un sistema. Los procesos se describen mediante CSP y las comunicaciones entre estos procesos son los valores de tipos de datos CASL. En este trabajo se utilizan conjuntos de pruebas que se construyen manualmente, el conjunto de pruebas es ejecutado en la implementación de un entorno de prueba de caja negra. Donde la extensión se logra mediante la adición de eventos en los procesos o por la adición de nuevos procesos en la especificación existente. Los resultados se logran mediante la ejecución de casos de prueba de la especificación(es) del producto, las características seleccionadas en la combinación de datos y la aplicación del proceso CSP-CASL. Se demuestra la aplicación de la especificación formal como una nueva teoría de prueba. La ventaja de la especificación formal está en la generación de artefactos de prueba y en el análisis de los productos que utilizan estos artefactos. Una Mejora de la relación entre el módulo común y la especificación de producto proporciona el camino para la reutilización de los paquetes de prueba. Teniendo como objetivo automatizar la generación de la suite de prueba, que pueda comprobar automáticamente el funcionamiento del sistema con el veredicto apropiado de la prueba.

2.3.6. Manejo de Pruebas de Software en Organizaciones de Software

2.3.6.1. Organizaciones de Software que utilizan el Enfoque SPL

Las grandes empresas que desarrollan software han adoptado el enfoque SPL, en el desarrollo de sus proyectos de software, al conocer los beneficios que pueden obtener al utilizarlo, además de que cuentan con los recursos humanos, económicos y de infraestructura para su adopción [23]. Proporcionándole, a la organización mayor competencia en la industria del software, al emplear la derivación y reutilización de los

activos base de la SPL. Algunos de estos beneficios son: el aumento de tiempo de salida al mercado de los productos de la línea, el uso más eficiente del talento humano, el incremento en la satisfacción del cliente, el incremento en la calidad del producto, la disminución de los riesgos del producto, la capacidad de personalizar los productos y la capacidad para mantener presencia en el mercado [2].

Sin embargo, existen pocos estudios que evidencian la adopción del enfoque SPL en las pequeñas y medianas empresas desarrolladoras de software, que expliquen en detalle; el proceso del uso de una línea de productos y la utilización del enfoque en la calidad de la una línea de productos. Pese a ello, Da Silva, Da Mota Silveira Neto, O'Leary, De Almeida y Meira [23] señalan que la ingeniería de SPL se ha aplicado especialmente en el desarrollo de software a gran escala. Por lo tanto, optan por dirigir su investigación a un caso de estudio contextualizado para una pequeña y mediana empresa (PYME) de Brasil que desarrolla sistemas en el dominio de la información para la gestión médica. Con el fin de argumentar el uso de métodos ágiles para la determinación del alcance y la disciplina de requisitos en una SPL. Incluyendo ciertos factores que consideran trascendentales en el desarrollo de un proyecto de software, como son: la comunicación, la colaboración, lo ciclos de iteración, la ausencia de adaptabilidad y flexibilidad, el aumento del esfuerzo y la reducción de la motivación en el transcurso de un proyecto. En cuanto al caso de estudio, la empresa introduce el alcance y la disciplina de requisitos SPL y durante su ejecución, los expertos en el dominio validan los productos tradicionales (de la empresa) y los artefactos producidos por un equipo encargado de líneas de productos. Donde la delimitación del alcance del producto les permitió la identificación de funciones de revisión, la identificación de productos, la construcción y validación del mapa de productos. Lo cual, les suministro la actividad de inspección que evalúa la calidad de los artefactos construidos. En la disciplina de requisitos SPL, la construcción de casos de uso para describir los escenarios principales y las variabilidades, utilizando etiquetas que permitieran seguir la trazabilidad de los artefactos y los requisitos. La razón por la cual los autores seleccionan el alcance y la ingeniería de requisitos es por determinan que son dos disciplinas importantes que capturan la esencia del negocio y el dominio de la organización.

Para Miguel A. Laguna [31] "el desarrollo de una SPL supone un reto considerable para las pequeñas organizaciones". Es así, que este autor, toma la iniciativa de adaptar el Proceso Unificado (UP) de desarrollo de software para agrupar las técnicas adelantadas para la ingeniería de línea de productos dentro de la ingeniería de aplicación. Para ello, inicia con la búsqueda de los modelos que puedan representar la parte común y variable de los activos, siguiendo la definición del cómo registrar la trazabilidad entre el modelo, la arquitectura y la implementación de la SPL, para facilitar la derivación de cada aplicación concreta. La propuesta [39] es evaluada mediante un caso de estudio que realiza la parte de análisis del dominio de un sistema web de comercio electrónico. Inicia con la búsqueda de un modelo de características para expresar la variabilidad, es así que utiliza FODA (Feature Oriented Domain Analysis) o FORM como modelo. Consecuente, propone un mecanismo de combinación de paquetes (package) mezcla (merge) de UML 2 para representar de forma ortogonal las variaciones arquitectónicas de la línea de productos, su relación directa con las características opcionales e incluso la utilización de paquetes de clases parciales, con la estructura del código que implementa el diseño. El mecanismo

“package merge”, consiste básicamente en añadir detalles de forma incremental, para formar una relación entre dos paquetes que luego pueden ser combinados e ir localizando las modificaciones necesarias del modelo de diseño asociado a las características. La esencia de este estudio es organizar la variabilidad de una línea de productos utilizando únicamente modelos y lenguajes de programación estándar. Además, de guiar la trazabilidad entre niveles que permiten derivar la implementación de cada aplicación concreta a partir de la configuración de características deseada.

2.3.6.2. Proceso de Pruebas de Software Orientado a Pequeñas Organización de Software

Entrando, un poco más en detalle, a continuación, se describen dos estudios que han sido aplicados a pequeñas organizaciones desarrolladoras de software en el contexto de pruebas y líneas de productos.

Para Gruner en su propuesta [62], recoge una serie de problemas de las pequeñas empresas al momento de desarrollar software y busca mejorar dichos problemas implementando un proceso de pruebas que permita garantizar la calidad del producto final. Primeramente, el autor identifica las áreas problemáticas alrededor de los procesos de pruebas en una determinada empresa. Donde, el primer problema fue la falta de conocimiento de los empleados sobre las pruebas de software, debido a la falta de experiencia en la industria, por el hecho de ser graduados “recientemente”. Otro problema, fue la falta de compromiso por parte de la empresa hacia las pruebas, ya que no presentaba una documentación relacionada con las pruebas y los estándares de calidad para la ejecución de un proyecto, tales como planes de pruebas o plantillas de pruebas. Otro problema que identificó, fue la falta de ambientes adecuados para pruebas. Pero, el principal problema que presentaba dicha empresa era que el entorno de producción nunca podría ser replicado correctamente en un entorno de prueba.

Con el ánimo de dar soluciones el autor plantea un modelo formal para el desarrollo de pruebas funcionales de software en base al *Modelo de Madurez de Pruebas Integrado* (TMMI) y los *Procesos de Pruebas Críticos* (CTP), ya que son modelos de mejora de procesos de prueba seleccionados como línea base para el proceso de pruebas. El modelo CTP contiene 12 “procesos críticos” que pueden ser aplicados en cualquier software organización y TMMI es un modelo de “peso pesado” (en comparación con el más ágil o “ligero”), estructurado más fuertemente y exhaustivamente, que sigue el mismo enfoque por etapas de la *Integración de Modelos de Madurez de Capacidades* (CMMI), para áreas de proceso. Estas áreas de proceso están creadas de metas específicas y genéricas. Este trabajo solo se ha centrado en niveles TMMI dos y tres, ya que puede tomar hasta dos años para alcanzar un nivel de madurez. Estos dos niveles incluyen áreas como la planificación de las pruebas, la política de prueba y la estrategia, supervisión de pruebas y control, diseño de pruebas y ejecución y organización de pruebas.

Ahora Lucía, José, Mauricio, Rojas-montes, Pino-correa y Martínez en [12], proponen un proceso liviano definido para guiar y apoyar la realización de las pruebas en pequeñas organizaciones desarrolladoras de software. La incorporación de técnicas de pruebas funcionales en el proceso propuesto ofrece una manera organizada y sistemática de llevar

a cabo actividades requeridas para evaluar un producto software, especialmente aquellas labores que tienen que ver con el diseño y la ejecución de las pruebas.

Este proceso fue definido a partir del análisis de algunos procesos de pruebas existentes en modelos de referencia y en la literatura. Además, fue aplicado “exitosamente” en una pequeña empresa caleña de desarrollo software, que se enfoca en crear soluciones para el gobierno en línea. Los autores, considerando pertinentes las siguientes etapas: (i) la clasificación y análisis de los procesos de pruebas existentes, (ii) la identificación, análisis e integración de técnicas de pruebas funcionales, y (iii) la identificación, observación y análisis de las pruebas de acuerdo con los procesos y formas de trabajo para el desarrollo de software en una pequeña empresa; para la construcción de dicho proceso.

El proceso de pruebas propuesto describe 6 fases: (i) inicio, (ii) planeación, (iii) diseño, (iv) ejecución, (v) monitoreo y control, y (vi) finalización; en cada una de las cuales se especifican sus actividades, tareas, productos de trabajo y roles. Para validar la propuesta los autores llevaron a cabo un proyecto piloto, el cual permitió identificar la necesidad de mejorar la especificación de requisitos para obtener una buena calidad en el software. De ahí que los beneficios de las pruebas no solo se vieron en el área de desarrollo de proyectos, sino que también se dejó notar en el resto de las áreas de negocio y especialmente en la satisfacción del cliente final.

Los dos casos mencionados anteriormente buscan proponer un proceso de pruebas aplicable por una pequeña empresa, el primero de ellos basado en TMMI y el segundo una propuesta ágil, ambos consideran áreas como la planeación, diseño, ejecución y gestión de las pruebas los cuales buscan organizar el proceso de pruebas en la empresa que aplique alguno de los procesos, al tiempo que sea lo suficientemente menudo para ser aplicable por una pequeña empresa.

2.3.6.3. Estudios Relacionados con la Derivación de Casos de Pruebas

El proceso de derivar casos de pruebas en SPL, multiplica significativamente las funcionalidades a probar, debido a la variabilidad de los requisitos, cada punto de variación multiplica el número de pruebas que se requieren [63]. La idea central de la derivación, es preservar las variabilidades que pueden encontrar en los casos de uso y casos de pruebas construidos en la ingeniería de dominio y aplicarlas dentro de los casos de pruebas en la ingeniería de aplicación [63].

La derivación de un producto que pertenece a la SPL, depende en gran medida de expertos, ya que la participación en este proceso, debe dominar el conocimiento implícito en la gestión de la variabilidad y la reutilización de activos [64]. Aunque, la variabilidad que proporciona un producto se comunica a clientes, expertos de dominio, gestores del proyecto y desarrolladores, que son por lo general los usuarios que intervienen en una SPL. Pero, desde la perspectiva de derivar un producto, la variabilidad es difícil de usar si los modelos no están debidamente estructurados y modularizados [64], por tal motivo un experto con su experiencia mejora la calidad del diseño en los modelos y permite la derivación más fácilmente.

2.3.6.3.1. Estrategias de Derivación en SPL

Kamsties, Pohl, Reis y Reuys en su propuesta [63], exponen varias estrategias, orientadas a la derivación de casos de pruebas en el contexto de líneas de productos, las cuales se mencionan seguidamente:

- La “derivación de casos de pruebas basado en escenarios” (Scenario based TEst case Derivation - ScenTED) [63][65], es una técnica que se basa en modelos, orientada a la reutilización de los casos de pruebas [65], para derivar sistemáticamente y automáticamente los casos de pruebas dentro de la ingeniería de aplicación [63].
- Estrategia pragmática, evita el desarrollo de casos de pruebas en la ingeniería de dominio y construye casos de pruebas únicamente en la ingeniería de aplicación, limitándose a copiar los casos de pruebas que cumplen con las especificaciones del producto a probar. Pues, los casos de pruebas pueden ser reutilizados directamente si sólo si el caso de uso relacionado es implementado por un nuevo producto que tiene las mismas variantes. El inconveniente de esta estrategia es que si se realizan cambios en los activo core no se puede mantener el caso de pruebas.
- Estrategia de abstracción y parametrización, en esta los pasos de pruebas definidos para las variantes particulares (producto), se derivan y se añaden a los casos de pruebas del dominio. El inconveniente, se presenta en que si se eligen las mismas variantes para diferentes productos se deben probar indistintamente.
- Estrategia de segmentación, la idea es derivar varias veces los casos de pruebas del dominio, captando variantes en cada caso de pruebas. Pues, un punto de variación se presenta en un caso de prueba a través de un conjunto de segmentos de pruebas. Los casos de pruebas de aplicación se derivan seleccionando los segmentos que prueban las variantes elegidas. El inconveniente de esta estrategia, es si dos casos de pruebas comparten un punto de variación, los resultados serían redundantes.
- Estrategia de fragmentación, la parte común y la parte variable de los casos de pruebas del dominio son derivados por separado, hasta que se llega a una especificación de caso de pruebas completa, es decir, un caso de prueba se deriva por fragmentos, con el fin de cubrir el caso de uso y los fragmentos puedan ser reutilizados en otros casos de pruebas. También, en la ingeniería de aplicación los casos de pruebas se concatenan con los casos de pruebas previamente fragmentados. El inconveniente, está en el aumento del esfuerzo en la mantención de estos fragmentos de pruebas.

Los autores, realizaron una práctica basada en diagramas de secuencia y un caso de uso base para aplicar la estrategia de fragmentos y como resultando obtuvieron, que los fragmentos de casos de pruebas resultantes necesitan una notación que indiquen que fragmentos pueden ser concatenados para construir un caso de pruebas completo. Sin embargo, de manera general, concluyen que cada estrategia tiene sus fortalezas y debilidades, pero que no hay una sola buena estrategia para derivar casos de pruebas en una línea de productos.

2.3.6.3.2. Metodología para la Derivación de Casos de Pruebas

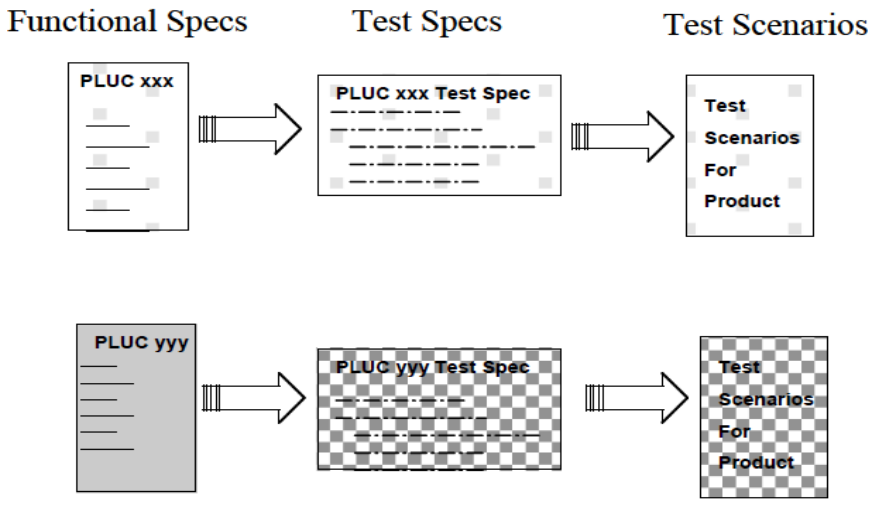
Últimamente, los procesos de desarrollo software se construyen a partir de artefactos software reutilizables, pues las aplicaciones pueden compartir requisitos de funcionalidad y al mismo tiempo usuarios similares, pero con características específicas que los hacen diferentes uno de otros [13].

Para Bertolino y Gnesi [13], es posible construir una SPL desde cero; es decir sin la necesidad de tener un activo core, o desde sistemas heredados. Facilitando, la tarea de pasar de nivel de familia (línea de productos) a nivel de producto a través de una instancia del proceso e igualmente desde el nivel de producto a nivel de familia por medio de una abstracción del proceso de desarrollo software [13][7]. Un proceso relevante para esta investigación [13] es el proceso de pruebas de software, el cual se considera un problema, en el sentido de cómo probar una SPL, ya que, por lo general, las pruebas se centran en la planificación de las mismas y no en la derivación de los casos de pruebas. Dado que las pruebas para una SPL, se obtiene de forma iterativa e incremental, porque los requisitos de una línea de productos se consideran como partes comunes y variables. Para abordar esta cuestión, tanto Bertolino, Fantechi, Gnesi y Lami [7] y Bertolino y Gnesi [13], proponen una metodología de ensayo para SPL denominada PLUTO, que a través de casos de uso para líneas de productos (PLUCs), logran construir casos de pruebas que contienen elementos comunes y permiten derivar un conjunto de escenarios, destacando que estos casos de pruebas, no son construidos con el mismo sentido usual, es decir, indicando que se espera una entrada precisa para ejecutar la prueba, y una secuencia de acontecimientos y la salida esperada, más bien representan la abstracción de los posibles escenarios de pruebas que en realidad son los escenarios de uso que necesitan estar probados para validar los requisitos de los usuarios finales.

Conociendo, que un escenario de ejecución de pruebas sigue la trayectoria de un caso de uso, donde el flujo principal (o escenario principal de éxito) se representa por medio de una secuencia indexada, que describe una secuencia única de éxito de acciones del sistema. Incluyendo etiquetas que personifican la variabilidad de un escenario, estas etiquetas pueden ser de tres tipos: (i) etiquetas paramétricas, su instancia está asociada a valores reales de un parámetro para desarrollar y probar los requisitos de un producto específico, (ii) etiquetas alternativas, expresan la posibilidad de crear un conjunto de posibles opciones (variantes) para después ser seleccionadas para un producto específico y finalmente (iii) etiquetas opcionales, la creación de instancias se logra mediante la selección en un conjunto de valores para un producto derivado [13].

La figura 9, presenta el esquema que la metodología PLUTO utiliza para construir casos de pruebas funcionales, partiendo de la especificación de casos de uso (PLUCs), hasta lograr la obtención de escenarios de pruebas, que se refinan a medida que se selecciona qué escenario (escenario de éxito) se va a ejecutar para la prueba. Sin dejar a un lado la variabilidad que se expresa a través de etiquetas predefinidas [13].

Figura 9. Esquema de pruebas de la metodología PLUTO[9].

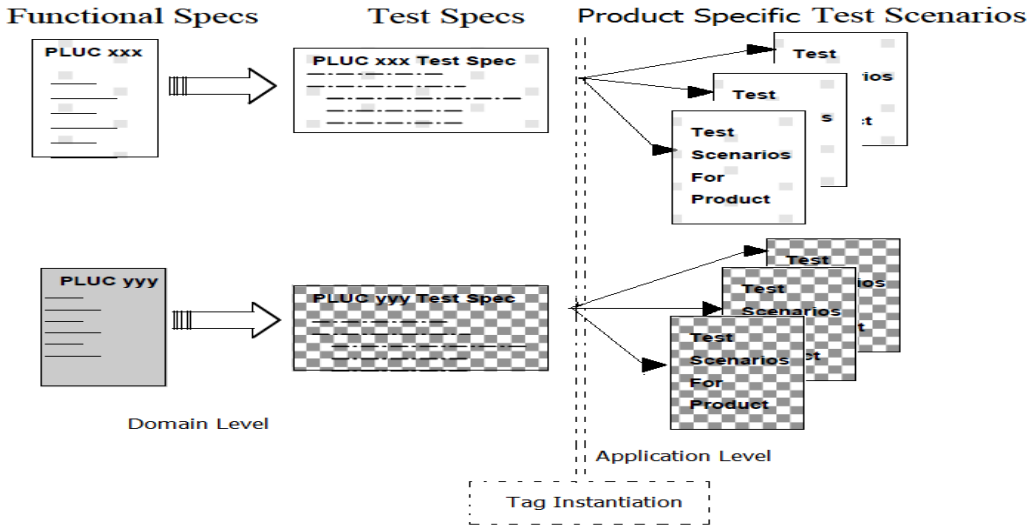


Para evitar la construcción redundante de escenarios, las características pueden ser anotadas como limitaciones, que pueden ser de dos tipos: (i) propiedades, que se establecen para ciertas particularidades, y expresiones de selección, modeladas en forma de condiciones simples como el IF o (ii) condiciones especiales, este tipo de restricciones es útil para reducir el número de casos de prueba, ya que se establecen algunas marcas (“error” y “única”). Donde, las posibilidades marcadas con “error” y “única” se refieren a condiciones erróneas o especiales respectivamente. Estas marcas son manipuladas para probar los artefactos de software y no necesitan ser combinadas con todas las opciones [13].

En general, una especificación de prueba para una SPL incluye una directiva hacia la derivación de casos de prueba, en esta investigación los autores, realiza la derivación por medio de la combinación de las opciones relevantes de dos PLUCs relacionados, analizando los escenarios obligatorios y las variantes de pruebas necesarias, para formar el activo de casos de pruebas para la línea de productos. Pues, los casos de prueba se derivan cuando un producto específico está siendo desarrollado después de tener fijas las etiquetas en cada PLUC, con los valores apropiados. Luego, cada especificación de prueba, desarrolla un conjunto diferente de casos de prueba correspondientes a cada producto de la SPL. En la figura 10, se presenta el esquema que utiliza PLUTO para construir algunos de los casos de prueba y conjunto de escenarios de pruebas, obtenidos para diferentes productos, es decir, para diferentes asignaciones de etiquetas [13].

[9] Tomada de [13].

Figura 10. Esquema de derivación de la metodología PLUTO[10].



2.3.6.4. Actividades Claves para Derivar Productos Software

Los autores Rabisera, O'Leary y Richardson en [9], consideran que existen pocos estudios, que se han ocupado de experimentar una forma válida de derivar productos software, pues en la mayoría de los casos, la industria de software construye sus productos con otros fines, sin pensar en la derivación de los mismos, y sin establecer cuáles son los elementos claves para la reutilización. Dentro de la ingeniería de aplicación, la derivación es un proceso importante que permite seleccionar y personalizar los activos de software que definen a una SPL, pues el cliente o el mercado puede requerir productos a partir de otro, de tal forma que se logre maximizar la reutilización a gran escala. Por tanto, Rabisera, O'Leary y Richardson presentan en su estudio algunas actividades que consideran claves para derivar productos dentro de una SPL. Partiendo, del análisis de dos enfoques existentes como son: el modelo de referencia del proceso de derivación del producto (Pro-PD) y la toma de decisiones orientada a la ingeniería de líneas de producto para una reutilización efectiva: configuración centrada en el usuario (DOPLER^{UCon}).

Primeramente, se considera el enfoque Pro- PD, con el fin de explorar qué actividades se han determinado para derivar productos. Señalando, que este enfoque emplea patrones de proceso para coordinar y sincronizar productos de una multiplataforma, a través de la construcción de bloques para la creación de instancias de proceso, que definen un producto software. La guía del modelo de referencia, que construye Pro-PD se presenta en la figura 11, donde se despliegan las actividades claves para la derivación de productos individuales de software en este enfoque

[10] Tomada de [13].

Figura 11. Modelo de Referencia Pro-PD[11].

Los autores proponen la actividad "Pre-Derivación" para la incorporación de prácticas que se ejecutan antes de realizar la derivación del producto software como tal, pues en primera medida, esta actividad apunta a la construcción de los requisitos específicos del producto requerido y de forma iterativa realizar ciertas sub-actividades que se muestran en la tabla 2.

Tabla 2. Actividades Pre-Derivación, Pro-PD[12].

Sub-actividades de Pre-Derivación	Propósito
Traducir al cliente el análisis de los requisitos de cobertura	"Traducir" al lenguaje del dominio los requisitos del cliente. Determinar el nivel de satisfacción de los requisitos, mediante la configuración base y el documento de requisitos asignado/sin asignar.
Negociación con el cliente.	Negociar los requisitos (sin asignar) del cliente y comprobar su viabilidad.
Crear los requisitos específicos del producto	Implica la fusión de los requisitos mapeados y negociados
Implementación del alcance de los requisitos	Los requisitos funcionales y no funciones para el sistema se especifican en el alcance. Los requisitos son designados para la implementación de plataformas o productos específicos.
Crear los casos de prueba específicos del producto	Creación de casos de prueba, utilizando los requisitos específicos del producto.

[11] Tomado de [9].
[12] Tomada de [9].

Sub-actividades de Pre-Derivación	Propósito
Asignar los requisitos	Asignar requisitos a las disciplinas pertinentes, por ejemplo, la disciplina de hardware, algoritmos. Dar prioridad a la iteración para requisitos particulares del producto.
Crear una guía para la toma de decisiones	La guía está orientada a los requisitos específicos del producto. La variabilidad restante deberá ser explicada de acuerdo a la complejidad del problema, en la representación de la variabilidad de la línea de productos.

Dentro de las actividades claves, los autores han definido otras acciones para el desarrollo y las pruebas de un producto específico de una SPL. Las cuales, permiten la configuración de dicho producto hasta que este cumpla con las expectativas del cliente. La tabla 3 presenta y describe estas acciones.

Tabla 3. Desarrollo y pruebas de un producto – Pro-PD[13].

Pro- PD desarrollo de un producto y actividades de pruebas	Propósito
Identificar el desarrollo del producto requerido.	Satisfacer necesidades a través de la reutilización de la plataforma existente.
Desarrollo / adaptación de componentes	Se desarrolla a nivel de producto, el código fuente para implementar nuevas funcionalidades o para adaptar un componente de la plataforma existente.
Componentes de pruebas unitarias	Cuando un componente está construido o adaptado, puede ser sometido a pruebas rigurosas a través de pruebas unitarias o versiones de pruebas anteriores
Integrar y crear la compilación del producto	Los componentes desarrollados o adaptados están integrados en la configuración de productos integrados.
Ejecutar prueba del sistema	El producto tiene que ser probado para verificar el cumplimiento de los requisitos específicos del producto. Las pruebas utilizadas a nivel de plataforma pueden ser reutilizadas.
Evaluar los resultados	Se determina el proceso de éxito o de fracaso del producto. Se discuten las mejoras que se pueden hacer el proceso de entrega de un producto.
Proporcionar información al equipo de la plataforma	El resultado del uso de los activos base se da al equipo del proyecto. Además, el equipo de producto identifica los componentes específicos del producto que la plataforma podría beneficiarse potencialmente en la adopción.

Posteriormente, DOPLER^{UCon} es una actividad de la ingeniería de línea de productos orientada a la toma de decisiones, que fue creada con el objetivo de ser una herramienta de soporte para al enfoque de derivación del producto centrado en el usuario, utilizando modelos de variabilidad y DOPLER^{VM} es otra actividad que soporta el modelado y la gestión de la variabilidad. Pero, la esencia de esta propuesta [9], es la definición de actividades y

[13] Tomada de [9].

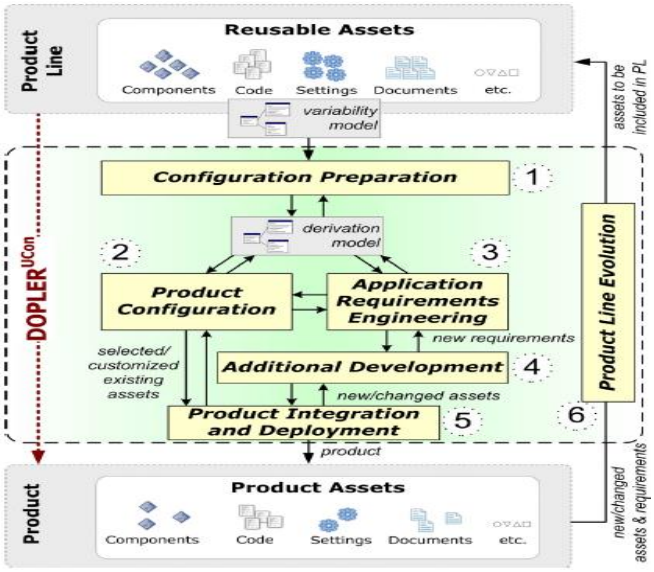
sub-actividades claves para la derivación de productos de software en SPL. Por tal razón, se abarca en su totalidad el contexto DOPLER^{UCon}, tal como se presenta en la tabla 4.

Tabla 4. Actividades DOPLER^{UCon}

Actividades DOPLER^{UCon}	**Descripción**	**Sub-Actividades**
Preparar la configuración	Los directores del proyecto preparan los modelos de variabilidad para un proyecto/cliente concreto, a partir de los requisitos conocidos desde el principio. Esta actividad tiene soporte en la herramienta "projectKing"	● Definir el proyecto. ● Revisar el modelo de variabilidad. ● Crear y gestionar una guía. ● Adaptar la variabilidad. ● Modelar y definir las funciones y tareas.
Configuración del producto	Introduce las decisiones del usuario, que dependen de sus funciones y tareas definidas en la derivación. Es una actividad crítica para la derivación de un producto debido a que las actividades posteriores dependen de ella.	● Determinar valores para la variabilidad. ● Comunicar la variabilidad. ● Establecer las decisiones y personalizar activos. ● Generar la configuración.
Ingeniería de requisitos	Propone: reproducir, negociar y gestionar los requisitos que no pueden ser cumplidos por la SPL. Los requisitos específicos incluyen la variabilidad disponible.	● Obtener y capturar los requisitos específicos del producto. ● Negociar los requisitos específicos del producto.
Durante el desarrollo	Se dirigen requisitos específicos del producto, teniendo en cuenta los activos existentes y las relaciones.	Ninguna
Integración y distribución de productos	Se preparan medios de integración de activos derivados y nuevos desarrollos para el despliegue.	Ninguna
Evolución de la ingeniería de línea de productos	Averiguar cuáles activos y/o modificaciones debería formar la línea de productos	● Analizar nuevos activos ● Analizar las nuevas necesidades

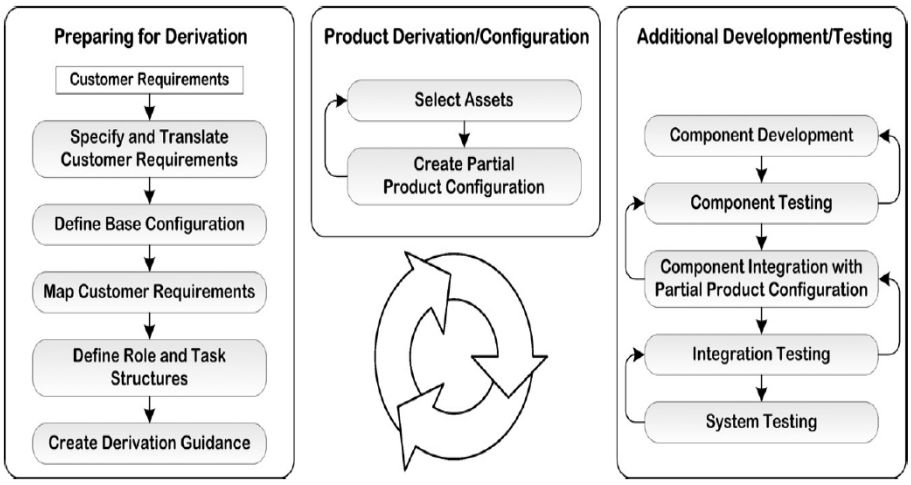
La estructura de las actividades que DOPLER^{UCon}, utiliza para la derivación y que anteriormente se especificaron en la tabla 4, se muestra en la figura 12. Donde, se puede observar algunos activos reutilizables como: los componentes, el código, ajustes y documentos, que pertenecen a la línea de productos. Una vez, identificados los activos intervienen las actividades para la derivación de que DOPLER^{UCon} ha instaurado para crear un producto software, el cual, a su vez contiene activos (base y nuevos) reutilizables, e inicia el proceso nuevamente para la derivación.

Figura 12. Estructura DOPLER^{UCon}[14].



Ya estudiados los enfoques Pro-PD y DOPLER^{UCon}, los autores de este estudio, consideran que cuenta con las bases necesarias para establecer sus propias actividades claves para la derivación de un producto de software. Es así, que la figura 10, se presenta la combinación de los enfoques y una nueva propuesta orientada a la derivación de productos. Además, en la tabla 5, se describen estas "nuevas" actividades y sus sub-actividades correspondientes.

Figura 13. Actividades claves para derivar productos software[15].



[14] Tomada de [9].
[15] Tomada de [9].

Tabla 5. Actividades Claves para Derivar Productos Software[16].

Actividades claves para la derivación	Descripción	Sub-actividades	Descripción sub-Actividad
Preparación para la derivación	Los requisitos de los clientes se especifican en un conjunto de documentos, a través de diversas tareas donde se analizan los requisitos potenciales para su reutilización y luego se asignan a un ente responsable.	Especificar y traducir los requerimientos del cliente.	Los requisitos deben estar claramente especificados, con el fin de: prevenir la confusión en la terminología y la descripción de los activos específicos para la obtención de un producto.
		Definir la configuración básica.	Tener como punto de partida una configuración base para derivar. Ejemplo: un conjunto existente de configuraciones de plataformas y/o experiencias de proyectos anteriores.
		Mapa de requisitos de los clientes.	Los requisitos del cliente se asignan a la configuración base y los requisitos que no se pueden asignar deben ser negociados con el cliente y que no afecte la mantención de los activos para la línea de productos.
		Definir la estructura de roles y tareas.	El objetivo es resolver quién es el responsable de gestionar la variabilidad que satisfaga los requisitos del producto derivado.
		Crear una guía de derivación.	La guía es importante para explicar la variabilidad en el caso de que exista algún problema en su representación dentro de la SPL.
Derivación del producto/ configuración	El objetivo de esta actividad es la derivación de productos software mediante la reutilizar de todos los artefactos disponibles, y de esta manera reducir al mínimo la cantidad de desarrollo que el producto requiera. El arquitecto del producto inicia seleccionando/	Seleccionar los activos.	Se basa en la estructura definida en la guía de derivación. Donde, los activos son seleccionados tienen que ser seleccionados y personalizados, a partir de la línea de productos, mediante la toma de decisiones o la selección de características y herramientas de apoyo necesarias para esta actividad. Existen dependencias y limitaciones en la descripción de la variabilidad y en los activos evaluados por la herramienta, ya que debe asegurarse la corrección del conjunto seleccionado y personalizado de los activos.

[16] Tomada de [9].

Actividades claves para la derivación	Descripción	Sub-actividades	Descripción sub-Actividad
	personalizando un conjunto de activos de la SPL. Posteriormente, determina los posibles desarrollos y las pruebas requeridas.	Crear configuración parcial del producto.	Una configuración parcial es creada paso a paso de manera iterativa ya que toda la variabilidad requiere un análisis de compatibilidad.
Desarrollo adicional/ pruebas	Los productos o componentes de software se desarrollan, se prueban, se integran y luego se despliegan.	Desarrollo de componentes.	Los nuevos componentes se deben desarrollar con la posibilidad de adaptarse a un activo, considerando el valor de la reutilización.
		Pruebas de componentes.	Los componentes son sometidos a pruebas rigurosas a través de pruebas unitarias que son construidas o adaptadas de otros proyectos.
		Integración de componentes con la configuración parcial del producto.	Los activos o componentes nuevos deben ser incorporados a la configuración base de los productos, detallando los cambios arquitectónicos para facilitar la integración de los activos desarrollados y adaptados.
		Pruebas de integración.	Las pruebas de integración son importantes para conocer si los activos desarrollados y adaptados interactúan correctamente con la arquitectura existente.
		Pruebas del sistema.	En las pruebas del sistema se revisan el cumplimiento de los requisitos específicos de un producto.

A manera de conclusión, el enfoque Pro-PD, gestiona los requisitos del proceso cuando se trata de equipos grandes y distribuidos, pues la comunicación en tales organizaciones tiende a ser demasiado dependiente de la documentación. Por tanto, deciden que sea excesivamente detallada, aunque presenta el inconveniente de disminuir la trazabilidad, al no identificar correctamente los artefactos para la reutilización.

La esencia de Pro-PD y DOPLER^{UCon}, no es precisamente la calidad de software, el interés de los autores es adoptar con madurez los elementos importantes para adquirir la derivación de un producto de una SPL. Sin embargo, se preocupan que los productos satisfagan los requisitos del cliente o el mercado. En Pro-PD, en la fase de desarrollo y pruebas, los requisitos propuestos son probados, con el fin de establecer si satisfacen la línea de productos y la variabilidad proporcionan. En cambio, el enfoque DOPLER^{UCon} en

las actividades destinadas para el desarrollo del producto y la fase de pruebas se producen en la ingeniería de aplicación de requisitos, y se componen de un desarrollo adicional, de la integración de productos y despliegue y de la fase de evolución de la línea de productos, para averiguar cuáles activos pertenecen a la SPL. Y las actividades definidas para la derivación de activos reutilizables, requieren un grado de variabilidad que implica la selección y personalización de los activos o componentes de una SPL, que corresponde a un contexto, dominio u organización particular.

2.3.7. Tabla de Comparación de los Trabajos Relacionados

La Tabla 6, presenta una comparación de las propuestas citadas en los trabajos relacionados de este documento (sección 2.3), con el fin de identificar como dichos autores contextualizan el tema de las pruebas de software de una SPL. Por tanto, se definen algunos aspectos como: (i) activo de prueba, presenta al artefacto utilizado para generar los casos de prueba, estos activos pueden ser: casos de uso (CU), casos de pruebas (CP), datos de pruebas (DP), informe de pruebas (IP), grafos de estado (GE), y/o un plan de pruebas (PP), además se considera si estos artefactos fueron reutilizados (R), (ii) aplicabilidad de la prueba, consiste en determinar la forma por la cual se ejecutan los casos de pruebas construidos, en este caso existen dos forma como son: automática, que indica que los casos de pruebas fueron ejecutados por sí solos o que realizan total o parcialmente un proceso sin ayuda humana; y manual: indica la descripción de los pasos de pruebas que realizará un evaluador, (iii) refinamiento, indica el nivel de especificación de algunos activos de prueba y (iv) la derivación, describe de donde proviene las variaciones o escenarios de los casos de prueba.

Tabla 6. Comparación de trabajos relacionados con SPL.

Trabajos Relacionados	Activos de Prueba		Aplicabilidad de la Prueba	Refinamiento	Derivación
A. Bertolino, A. Fantechi, S. Gnesi, y G. Lami [7]	CU	R	Manual	Las relaciones que se logran identificar para CU y CP son soportadas por el estándar UML, y describen notaciones gráficas de extensión, generalización y asociación. Por tanto, las interacciones básicas entre usuario y sistema se descomponen en una o varias instrucciones detalladas	La derivación de CP se realiza mediante la combinación de las opciones relevantes desde los CU de línea de productos.
	CP	R			
T. E. Colanzi, W. K. G. Assunção, D. de Freitas Guilhermino Trindade, C. A. Zorzo, y S. R. Vergilio. [3]	CU	R	Manual		
	DP				
	CP	R			
	PP				
A. Bertolino y S. Gnesi [13]	CU	R	Manual		
	CP	R			
J. D. Mcgregor [30],	CU	R	Automática		
	CP	R			
	DP				
	PP	R			
	IP				
M. Lochau, S. Oster, U. Goltz, y A. Schu [59]	CP	R	Automática		
	GE	R			

Trabajos Relacionados	Activos de Prueba		Aplicabilidad de la Prueba	Refinamiento	Derivación
A. Bertolino y S.Gnesi [57]	CU	R	Automática	dependiendo del dominio del producto.	
	CP	R			
	DP				
	Documento de CU	R			
C. R. L. Neto, I. D. C. Machado, I. Do Carmo MacHado, P. a. M. S. Neto, E. S. De Almeida, y S. R. De Lemos Meira [60]	CU	R	Automática		La derivación de CP se realiza mediante la combinación de las opciones relevantes de los GE de la línea de productos.
	CP	R			
	IP				
Mishra, S.[61]	CP		Manual		La derivación parte de la especificación algebraica de los CP particulares hacia los genéricos.
M. Granda y E. Hurtado.	CU	R	Manual	A medida que se logran especificar los CP, se detalla el modelo de CP para el producto.	La derivación de CP para un producto se realiza a partir de los CP del dominio y los casos de uso.
	CP	R			
	DP				

2.3.8. Catálogo de algunos Métodos y Técnicas de Pruebas de Software, que han sido aplicados en SPL

El propósito de realizar un catálogo es analizar algunos métodos, técnicas u otros enfoques utilizados en la construcción de casos de pruebas en el contexto de SPL, también examinar las técnicas que han adaptado las pequeñas organizaciones de software para planear, diseñar y ejecutar las pruebas de software, en el contexto SPL y determinar las técnicas o formas básicas de cómo las organizaciones construyen casos de pruebas derivados.

Se recopilan algunos estudios que describen la construcción o aplicación de métodos técnicas, modelos, enfoques o actividades claves relacionadas con pruebas de software y el enfoque SPL. Elementos que componen la estructura del catálogo: (i) tipo de contribución, se refiere especialmente a indicar si la investigación analizada es un método, técnica, modelo, enfoque o actividades claves , con lo cual se lograría formar un conjunto sistemático de reglas que permitan construir la técnica “deseada”, (ii) tamaño de la empresa,

este aspecto es básico, ya que nuestra propuesta está dirigida a organizaciones de software, en particular a pequeñas organizaciones, pero considerando la poca evidencia referenciada y aplicada (documentación y caso de estudio) en la industria de software, se extiende el tamaño de la empresa con el fin de tener más información que permita cumplir los objetivos de la técnica a desarrollar, y finalmente se considera si dicho estudio (iii) parte de casos de uso (CU), pues se ha determinado que específicamente la construcción de casos de pruebas funcionales debe partir de casos de uso, motivo por el cual, aquellos estudios tendrían mayor interés para nuestra propuesta de investigación.

Clasificación de métodos, técnicas, modelo, metodologías, herramientas actividades claves para las pruebas de software en SPL.

Tabla 7. Clasificación de estudios relacionados con pruebas de software en SPL**.**

Tipo de contribución	Estudios relacionados
Métodos	1. Specification Based Software Product Line Testing: A case study (CSP-CASL). 2. Improving the Testing and Testability of Software Product Lines (FIG Basis Path) 3. Elicitation of Use Cases for Product Lines (PLUs) 4. UML-Based Statistical Test Case Generation (UML- casos de pruebas) 5. An Automated Model-based Testing Approach in Software Product Lines Using a Variability Language (M-B testing) 6. An Automated Model-based Testing Approach in Software Product Lines Using a Variability Language (CVL-Tool) 7. Modeling Dependable Product-Families: from Use Cases to State Machine Models (MBT-SPL) 8. 12 System Testing of Product Lines: From Requirements to Test Cases (12-sTest) 9. Towards Software Product Line Testing using Story Driven Modeling (t-SMD)
Técnicas	1. Testing and inspecting reusable product line components (Test PL-C) 2. ScenTED technique (Scenario based TEst case Derivation) 3. Análisis De Propuestas Para Generación De Casos De Prueba Para El Control De Calidad. (generación CP)
Modelos	1. Test Overlay in an Emerging Software Product Line – An Industrial Case Study (test overly –SPL). 2. Generación automática de casos de prueba para Líneas de Producto de Software (QVT- CP)
Herramientas	1. Software Product Lines System Test Case Tool: A Proposal (RiPLE-TE) 2. SPLMT-TE: A Software Product Lines System Test Case Tool 3. Testing de Servicios Web para Líneas de Productos Software (W- S test)
Enfoques	1. Testing Variabilities in Use Case Models (T-UCv)
Actividades claves	1. Key activities for product derivation in software product lines

Tipo de contribución	Estudios relacionados
	(Derivación –PSPL)

A continuación, se describen algunos detalles de cada estudio presentados en la tabla 7.

Métodos

Tabla 8. Métodos relacionados con pruebas en SPL.

Nombre	CSP-CASL
Denominado por:	Los autores
Autores	S. Mishra
Tipo de contribución	Método
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Manual
Tamaño de la empresa (orientado el estudio)	No indica
Utiliza casos de uso	No
Descripción	Propone demostrar la capacidad de mantenimiento de los conjuntos de pruebas en una SPL que se puede lograr con un enfoque basado en especificaciones.
Fuente	https://www2.informatik.huberlin.de/~hs/Aktivitaeten/2006_CSP/CSP06_23.pdf
Nombre	FIG Basis Path
Denominado por:	Los autores
Autores	Isis Cabral, Myra B. Cohen, y Gregg Rothermel
Tipo de contribución	Método
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	Pequeña
Utiliza casos de uso	Si
Descripción	Proponen un gráfico basado en el método que selecciona productos y

	características para pruebas basadas en un gráfico de dependencia de características.
Fuente	http://cse.unl.edu/~grother/papers/splc10.pdf
Nombre	PLUs
Denominado por:	Para este catálogo
Autores	A. Fantechi, S. Gnesi, G. Lami y j. Dörr
Tipo de contribución	Método
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	No indica
Tamaño de la empresa (orientado el estudio)	No indica
Utiliza casos de uso	Casos de uso en líneas de productos (PLUs)
Descripción	Describe un enfoque para la derivación en SPL, utilizando el análisis de la documentación existen de usuario de sistemas para realizar los casos de uso (PLUs). Además, utilizan una disciplina para integrar la información heredada que se encuentra en la documentación existente de la línea de productos.
Fuente	https://pdfs.semanticscholar.org/91eb/4ff8c8d31074ede4a51e4510c4cfc51ba3b8.pdf
Nombre	UML- casos de pruebas
Denominado por:	Para este catálogo
Autores	Matthias Riebisch, Ilka Philippow y Marco Götze
Tipo de contribución	Método
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: no ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Automática o manual
Tamaño de la empresa (orientado el estudio)	No indica
Utiliza casos de uso	Si
Descripción	Presenta un enfoque para la generación de casos de prueba basados en modelos de casos de uso y refinados por diagramas de estado. Los casos de prueba resultantes son adecuados para ser llevados a cabo de manera convencional, es decir, bien manualmente o utilizando herramientas de prueba.

Fuente	https://pdfs.semanticscholar.org/70e6/3aceb04e78cf7bad59aa8fc721b3654b2ff1.pdf
Nombre	CVL-Tool
Denominado por:	Para este catálogo
Autores	Boni García, Rodrigo García-Carmona, Álvaro Navas, Hugo A. Parada G., Félix Cuadrado y Juan C. Dueñas
Tipo de contribución	Método
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: si
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	No indica.
Utiliza casos de uso	No
Descripción	Presenta la aplicación de un enfoque de pruebas automatizado para SPL impulsado por su estado-máquina y modelos de variabilidad para lograr la automatización de pruebas.
Fuente	http://oa.upm.es/6945/1/INVE_MEM_2010_75704.pdf
Nombre	MBT- SPL
Denominado por:	Para este catálogo
Autores	L. Erazo y E. Martins
Tipo de contribución	Método
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	No indica
Tamaño de la empresa (orientado el estudio)	No indica
Utiliza casos de uso	Si
Descripción	Presenta un enfoque de modelado de casos de uso que permite la extracción automática del modelo de máquina estados para un producto.
Fuente	
Nombre	12 –sTest
Denominado por:	Para este catálogo
Autores	C. Nebut, Y. Le Traon, and J.-M. Jezequel

Tipo de contribución	Método
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	No indica
Utiliza casos de uso	Si
Descripción	El enfoque que presenta se basa en la automatización de la generación de pruebas de aplicación para cualquier producto elegido, de los requisitos del sistema de una SPL Los requisitos se modelan utilizando casos de uso UML mejorados que son la base para la generación de pruebas.
Fuente	https://hal.inria.fr/inria-00512533/document
Nombre	t-SMD
Denominado por:	Para este catálogo
Autores	Sebastian Oster, Andy Schürr, y Ingo Weisemöller
Tipo de contribución	Método
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	No indica
Utiliza casos de uso	No
Descripción	Ofrece un enfoque para pruebas de software para SPL basado en modelos, utilizando el modelo impulsado por historia. Identifica un pequeño conjunto de instancias de un producto para que la verificación de este conjunto de pruebas garantice la exactitud de todas las posibles instancias de producto o incluso línea de productos.
Fuente	http://www.fujaba.de/fileadmin/Informatik/Fujaba/Resources/Publications/Fujaba_Days/tud-fi08-09.pdf#page=58

Técnicas

Tabla 9. Técnicas relacionadas con pruebas en SPL.

Nombre	Test PL-C
Denominado por:	Para el catálogo
Autores	Christian Denger y Ronny Kolb

Tipo de contribución	Técnica
Documentación	● Artículo: 1 ● Guías:no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Manual
Tamaño de la empresa (orientado el estudio)	Pequeña
Utiliza casos de uso	No
Descripción	Realiza un estudio empírico comparando las dos técnicas de detección de defectos “Inspecciones” y “pruebas funcionales”, en el contexto de la línea de productos en técnicas sobre reutilización de componentes de software.
Fuente	http://bit.ly/2nKiOrz
Nombre	ScenTED
Denominado por:	Por lo autores
Autores	A. Reuys, E. Kamsties, K. Pohl, y S. Reis
Tipo de contribución	Técnica
Documentación	● Artículo: 1 ● Guías: si ● Un ejemplo: si ● Documentación detallada disponible: si
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	Grande
Utiliza casos de uso	Si
Descripción	Tiene como objetivo reducir el esfuerzo en las pruebas de la SPL de productos. Es una técnica basada en modelos y orientada a la reutilización para derivar de casos de prueba en línea de producto.
Fuente	ftp://ftp.inf.puc-rio.br/pub/docs/FomularioSolicitacoes/LeandraMara-09-08-25167.pdf
Nombre	Generación CP
Denominado por:	Para este catálogo
Autores	M. Mejías, J. Torres, M. J. Escalona, J. J. Gutiérrez, J. A. Álvarez
Tipo de contribución	Técnica
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no

Aplicabilidad de la Prueba	Automática/sistemática
Tamaño de la empresa (orientado el estudio)	Grande
Utiliza casos de uso	No
Descripción	En este artículo se describen, se analizan y se comparan varias propuestas existentes para la generación sistemática de un conjunto de pruebas del sistema a partir de los requisitos funcionales para garantizar la calidad del sistema software.
Fuente	http://bit.ly/2nDkpie

Modelos

Tabla 10. Modelos relacionados con pruebas en SPL.

Nombre	Test Overlay –SPL
Denominado por:	Para este catálogo
Autores	Emelie Engström y Per Runeson
Tipo de contribución	Modelo
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Manual
Tamaño de la empresa (orientado el estudio)	Grande
Utiliza casos de uso	No
Descripción	Este estudio evalúa la cantidad y el tipo de prueba de integración de forma manual y la prueba del sistema en el contexto SPL. Además, explora las causas de la redundancia potencial en las pruebas.
Fuente	https://lup.lub.lu.se/search/ws/files/3196983/3738208.pdf
Nombre	QVT- CP
Denominado por:	Para este catálogo
Autores	Beatriz Pérez Lamancha
Tipo de contribución	Modelo
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado	No indica

el estudio)	
Utiliza casos de uso	No
Descripción	Los casos de prueba se generan automáticamente mediante el lenguaje de transformación QVT a partir de diagramas de secuencia extendidos para representar la variabilidad en la SPL.
Fuente	http://www.redalyc.org/html/922/92217153004/

Herramientas

Tabla 11. Herramientas relacionadas con pruebas en SPL.

Nombre	RiPLE-TE
Denominado por:	Por lo autores.
Autores	C. R. L. Neto, I. D. C. Machado, I. Do Carmo MacHado, P. a. M. S. Neto, E. S. De Almeida, y S. R. De Lemos Meira
Tipo de contribución	Herramienta
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: no ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	No lo indica
Utiliza casos de uso	si
Descripción	Evalúa cómo reducir el esfuerzo durante el SPL Proceso de Pruebas y, en consecuencia, cómo hacer la variabilidad De los activos de prueba manejable-
Fuente	http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.467.3171&rep=rep1&type=pdf
Nombre	SPLMT-TE
Denominado por:	Por los autores.
Autores	Crescencio Rodrigues Lima Neto, Ivan do Carmo Machado, Paulo Anselmo Mota Silveira Neto, Eduardo Santana de Almeida y Silvio Romero de Lemos Meira
Tipo de contribución	Herramienta
Documentación	● Artículo: 1 ● Guías: si ● Un ejemplo: si ● Documentación detallada disponible: si
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	No indica.

Utiliza casos de uso	Si
Descripción	Define los requisitos, diseño e implementación de una herramienta de prueba para SPL, dirigida a la creación y gestión de activos de prueba.
Fuente	http://www.lbd.dcc.ufmg.br/colecoes/sbqs/2012/034.pdf
Nombre	W-S test
Denominado por:	Para este catálogo
Autores	Alberto Barbosa León
Tipo de contribución	Herramienta
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: si
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	No indica
Utiliza casos de uso	Si
Descripción	Describe los aspectos de testing de servicios Web que han sido desarrollados dentro de una SPL y en particular aquellos referidos al testing de servicios Web.
Fuente	https://eciencia.urjc.es/bitstream/handle/10115/4507/Memoria-AlbertoBarbosaLeon.pdf?sequence=1&isAllowed=y

Enfoques

Tabla 12. Enfoques relacionados con pruebas en SPL.

Nombre	T-UCv
Denominado por:	Para este catálogo
Autores	E. Kamsties, K. Pohl, S. Reis, y A. Reuys
Tipo de contribución	Enfoque
Documentación	● Artículo: 1 ● Guías: no ● Un ejemplo: si ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	No indica.
Tamaño de la empresa (orientado el estudio)	No indica
Utiliza casos de uso	si
Descripción	Propone un enfoque para el desarrollo de casos de prueba utilizando casos de uso del dominio con variabilidad y de esta manera obtener casos de pruebas de aplicación y lograr la derivación.

Fuente	https://pdfs.semanticscholar.org/9ca3/4cc99a5a8c2b75750a02cdc94bac4c531ff4.pdf

Actividades claves para la derivación

Tabla 13. Actividades claves relacionadas con pruebas en SPL.

Nombre	Derivación -PSPL
Denominado por:	Para este catálogo
Autores	Rick Rabiser, Pádraig O'Leary y Ita Richardson
Tipo de contribución	Actividades claves
Documentación	● Artículo: 1 ● Guías: gráficos ● Un ejemplo: no ● Documentación detallada disponible: no
Aplicabilidad de la Prueba	Automática
Tamaño de la empresa (orientado el estudio)	No indica.
Utiliza casos de uso	No
Descripción	Presenta un informe donde se comparan dos enfoques de derivación de productos desarrollados. Ambos enfoques buscaron de manera independiente identificar las actividades de derivación de productos, uno a través de un modelo de referencia de procesos y el otro a través de un enfoque de derivación apoyado por herramientas.
Fuente	https://www.researchgate.net/publication/223064423_Key_activities_for_product_derivation_in_software_product_lines

A partir de los estudios del catálogo se puede concluir que los estudios analizados no evidencian detalladamente el proceso desarrollado y aplicado para construir caso de pruebas funcionales, estos mencionan el propósito de su investigación y pocos señalan el tamaño de la empresa de software, a la cual se puede aplicar con éxito la propuesta planteada, además son carentes en la documentación que soporte y refleje en gran manera el caso de estudio realizado. También, se puede notar que las empresas aplican diferentes formas para probar los productos de una línea, bien sea utilizando métodos, modelos, actividades o técnicas para la ejecución de dichos casos de pruebas y dependiendo de su organización interna realizan la especificación o representación de casos de uso y casos de pruebas. Asimismo, las organizaciones prefieren utilizar herramientas de software para ejecutar casos de pruebas, pues les ahorra tiempo y esfuerzo en el diseño de pruebas y datos de pruebas, aunque en la mayoría de casos no se puntualiza la herramienta de pruebas que fue utilizada, solamente se menciona la automatización (prototipos o herramientas) de las pruebas.

En la mayoría de estos estudios analizados los casos de prueba que se derivan de los casos de uso, utilizan la especificación existente de casos de uso para establecer la

cobertura de pruebas funcionales del producto. De tal manera que el equipo de desarrollo y el equipo de pruebas puede aprovechar el formato de casos de uso para derivar casos de pruebas [66].

2.3.9. Las Pequeñas Empresas de Software y SPL

Las pequeñas empresas de software representan la mayoría de empresas en varios países. Sin embargo, para persistir y crecer, las pequeñas empresas de software necesitan, soluciones eficaces de ingeniería de software. Muchas empresas consideran que las buenas prácticas y los enfoques de ingeniería de software son costosas, consumen mucho tiempo y que se orientan más hacia las grandes organizaciones, lo que es real, es que grandes y pequeñas empresas de desarrollo de software enfrentan desafíos similares, necesitan administrar y mejorar sus procesos de software, lidiar con los rápidos avances de la tecnología, operar en un entorno de software global y mantener su competitividad en el mercado a través de los productos y servicios que ofrecen. Sin embargo, a menudo las pequeñas empresas requieren enfoques diferentes que consideren sus modelos de negocio, objetivos, nicho de mercado, disponibilidad de recursos (financieros y humanos), capacidad de gestión y otras diferencias organizacionales y operativas, pero no todas las condiciones tienen porque ser negativas muchas de ellas tienen una estructura plana, una gestión orgánica que fomenta la innovación, por lo general se centran en un nicho de mercado que las grandes empresas han ignorado, construyen componentes para productos de otras empresas, u ofrecer servicios de soporte de productos que otros producen, tienen limitaciones de personal no solo en cantidad sino en conocimiento especializado y experiencia, como también sufren de finanzas apretadas [67].

Los investigadores están trabajando en adaptaciones de soluciones de ingeniería de software para pequeñas organizaciones, y el número de informes de experiencia sobre tales aplicaciones está aumentando, como ocurre también con el enfoque de producción SPL, la mayoría de reportes de casos exitosos corresponden a grandes empresas con experiencia en nichos de mercado, y algunos como Linder, Bosch, Kamsteries, Känsälä y Obbink [68], afirman que las pequeñas empresas no pueden permitirse los costos ni el engorroso proceso ni las estricta organización y planeación que requiere la adopción de SPL, otros como Gacek, Knauber, Schmid y Clements [69] consideran que las SPL ofrecen una oportunidad de éxito y competitividad a las pequeñas empresas, porque beneficios de SPL como reducción del tiempo de comercialización y la economía para la producción, son más críticos para una empresa pequeña que para las grandes. Las pequeñas empresas de software constantemente se enfrentan al reto de la supervivencia, trabajan de una manera ad-hoc, y su éxito se basa en desarrollar productos en tiempos cortos con poco personal. Se ha observado que a veces las pequeñas empresas tienen productos similares, pero no se desarrollan a partir de una plataforma común, utilizando familias de productos sin saberlo [70]. Y en base a esa observación se han realizado propuestas de enfoques de SPL orientados a pequeñas empresas, Nazar y Rakotomahefa [71], consideran que primero que se debe establecer una visión general de la empresa que permita determinar su capacidad de adoptar la tendencia de reutilización planificada de software antes de implementar plenamente el enfoque de SPL.

El aporte principal que ofrecen estos trabajos, es poder entender cómo las pequeñas organizaciones desarrolladoras de software operan dentro de la industria de software. Además, conocer la perspectiva sobre la adopción del enfoque SPL en pequeñas empresas desde varios autores. A partir de ello, nuestra propuesta pretende construir una técnica que pueda ser aplicada en una pequeña organización de software, teniendo en cuenta los factores o limitaciones que involucran a estas organizaciones y los beneficios que ofrecen el enfoque SPL.

CAPÍTULO III

UNA TÉCNICA PARA LA CONSTRUCCIÓN DE CASOS DE PRUEBAS FUNCIONALES BASADOS EN CASOS DE USO EN EL CONTEXTO SPL EN PEQUEÑAS ORGANIZACIONES - ttSPL

Este capítulo presenta en detalle la técnica ttSPL que se propone en este proyecto, la cual se encuentra orientada a la construcción de casos de pruebas funcionales a partir de la especificación de casos de uso para la ingeniería de dominio de SPL y la derivación de casos de prueba funcionales para los productos de la línea a partir de casos de uso (del dominio y del producto) y de casos de prueba del dominio.

Para presentar la técnica ttSPL se estructuran las siguientes secciones:

En la sección 3.1: Construcción de la técnica ttSPL, presenta el enfoque de ingeniería de método basada en ensamble de fragmentos de método y en patrones de método [24], empleando algunos de los referentes indicados en el capítulo II del presente documento, adaptados a la técnica ttSPL.

En la sección 3.2: Descripción general de la técnica ttSPL, presenta una visión general de un proceso SPL definido. El cual ttSPL utiliza para enmarcar su proceso en el contexto de líneas de producto

En la sección 3.3: Restricciones de la técnica ttSPL, indican las limitaciones que contiene la técnica ttSPL para su aplicación.

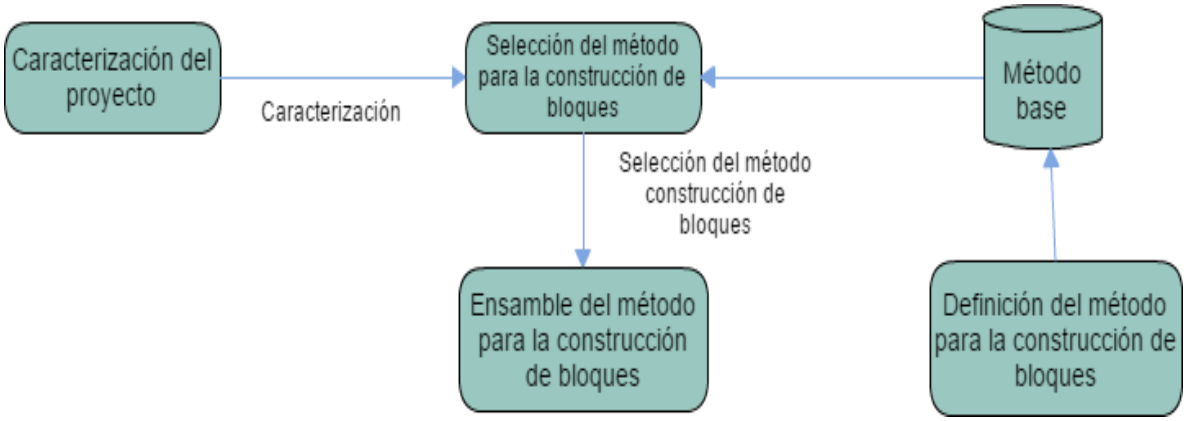
En la sección 3.4: Fragmentos de métodos para la construcción de la técnica ttSPL, presenta los fragmentos considerados para ensamblar la técnica ttSPL.

En la sección 3.5: Técnica ttSPL: específica los elementos que definen la técnica ttSPL, como roles y actividades que componen tanto el procedimiento ttSPL en ingeniería de dominio como el procedimiento ttSPL en ingeniería de aplicación.

3.1. Construcción de la Técnica ttSPL

Es necesario, indicar el proceso seguido para la construcción de la técnica ttSPL propuesta en esta tesis. En la figura 14, se presenta el proceso de ensamble de bloques que Fazal-baqaie, Luckey y Engels [17] propone como guía general para construir una técnica. La técnica ttSPL, es una técnica de refinamiento y derivación que permite relacionar en forma consistente los casos de uso y los casos de prueba, considerando las dos dimensiones de desarrollo de una SPL (Ingeniería de dominio e ingeniería de aplicación) y está dirigida a una pequeña organización de software.

Figura 14. Proceso general para la construcción de una técnica[17].



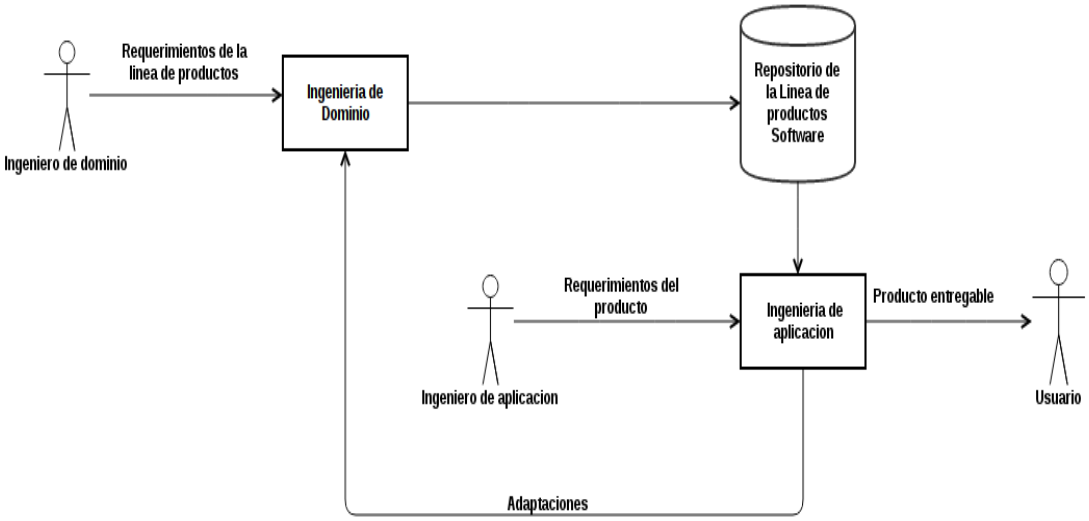
Para caracterizar el proyecto se considera las características y limitaciones típicas que las pequeñas organizaciones tienen para adaptarse a enfoques diferentes y así sostenerse y crecer en el mercado, teniendo en cuenta que las pequeñas empresas son calificadas como una versión reducida de una gran empresa, tanto así que son extremadamente sensibles y flexibles, lo que a su vez es una ventaja competitiva [67].

3.2. Descripción General de la Técnica ttSPL

La técnica ttSPL, plantea una estrategia para que pequeñas organizaciones productoras de software puedan derivar casos de prueba funcionales para un producto a partir de casos de prueba creados en el marco de una línea de productos software, además permite el refinamiento de los productos con nuevos comportamientos funcionales que pertenecen a un producto en particular. La figura 15, muestra una visión general de los procesos de desarrollo SPL utilizados en el método PLUS [72], donde un ingeniero SPL crea elementos reutilizables como modelos de requisitos, modelos de análisis y arquitectura durante la ingeniería SPL y luego los almacena en un repositorio SPL. A continuación, un ingeniero de aplicaciones reutiliza y configura estos modelos para una aplicación derivada del SPL. El proceso es iterativo, lo que significa que cualquier requisito, errores y adaptaciones no satisfechos descubiertos durante la ingeniería de aplicaciones se devuelven al ingeniero de línea de productos, que desarrolla los modelos SPL y actualiza el repositorio SPL.

[17] Tomada de [17].

Figura 15. Proceso SPL[18].



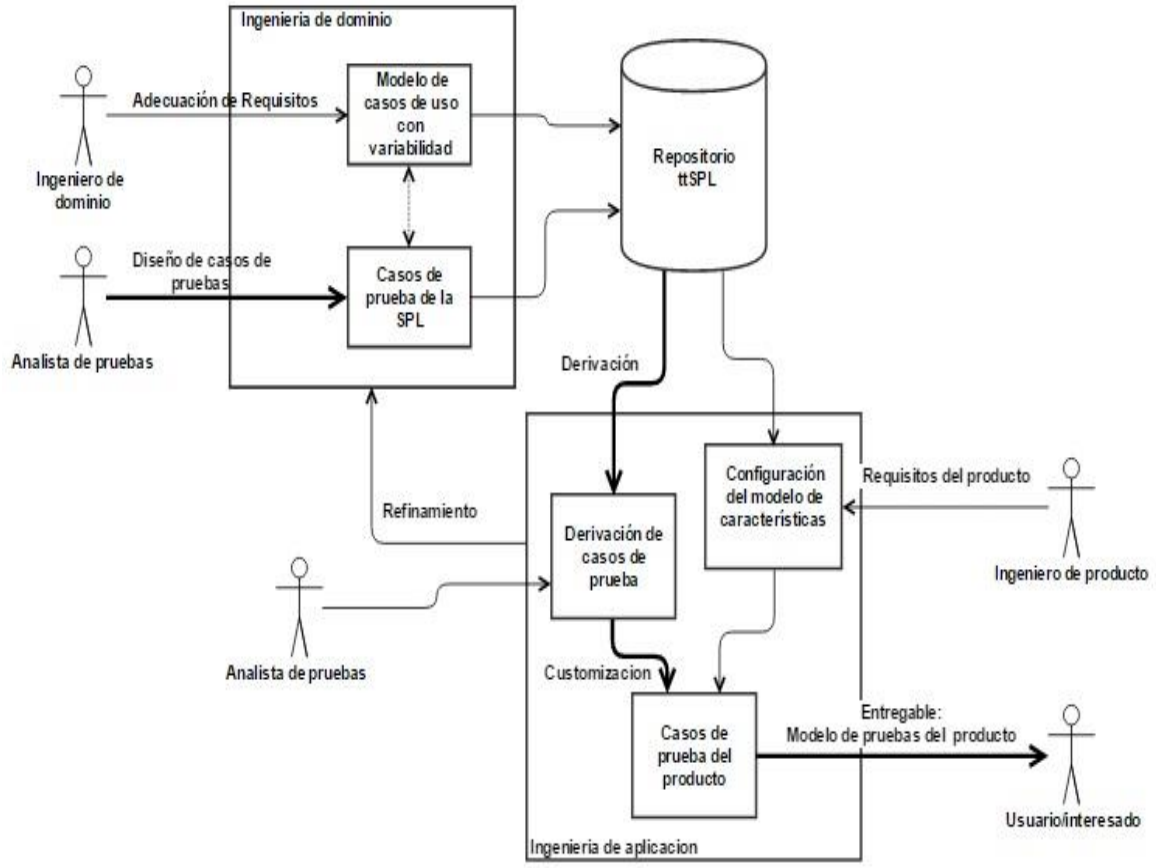
Con base en este proceso SPL (ver figura 15), ttSPL pretende partir de casos de uso ya que al obtener casos de prueba derivados directamente de casos de uso, se tiene la ventaja de contar con una especificación funcional clara necesaria para las pruebas y así garantizar una base sólida que permita verificar la trazabilidad del proceso de diseño de casos de prueba.

El manejo de la variabilidad de la SPL representada a través de un modelo de características de la SPL, permite elaborar un conjunto de casos de prueba que corresponden a un conjunto de productos, es decir el modelo de características de la SPL permite identificar los posibles productos que pueden configurarse y que necesitan unos casos de prueba que son almacenados en un repositorio. Entonces derivar un producto para la técnica ttSPL, es personalizar los productos compartidos dentro de la ingeniería de aplicación, por medio de la toma de decisiones para seleccionar los activos que constituyen un nuevo producto de la SPL y después personalizarlo (agregar y/o eliminar requisitos funcionales) para que cumplan una funcionalidad en particular. En la Figura 16, se puede observar el esquema general de ttSPL que permite observar la técnica bajo el proceso de SPL.

El propósito de ttSPL no es formalizar un reporte de pruebas, por tanto, el usuario no interviene en el proceso de aceptación de los casos de pruebas diseñados. Por esta razón, no se tiene en cuenta la ejecución de los casos de pruebas contenidos en el modelo de pruebas del producto, que constituye el producto/entregable de ttSPL.

[18] Tomada de [72].

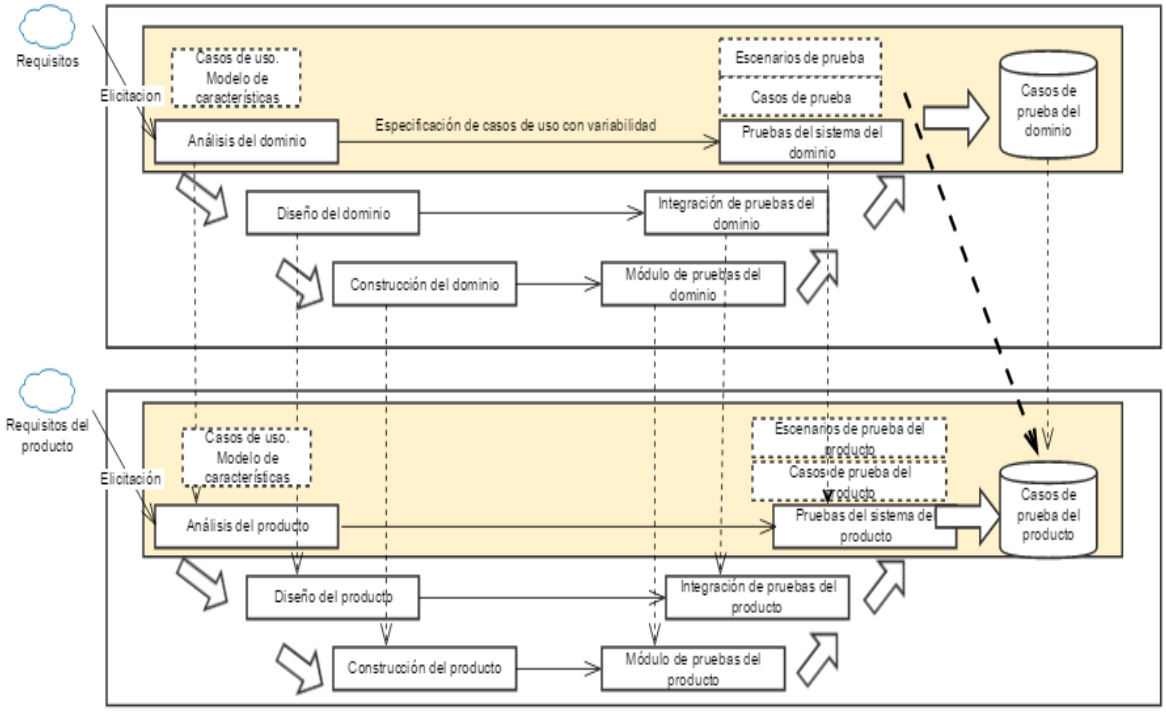
Figura 16. Esquema general de ttSPL, basado en el proceso SPL (ver. figura 15).



La técnica ttSPL permite acoplarse a las pequeñas organizaciones al contar con tareas claramente definidas con el ánimo de hacer fácil la aplicación de la técnica, enfocadas a la reducción del esfuerzo y el tiempo para el proceso de pruebas y permitir el aseguramiento de la calidad del producto software.

Cuando se habla de derivación de pruebas es preciso tener en cuenta un proceso de software definido que facilita el desarrollo y permita que pueda considerarse un plan de pruebas formalmente compuesto (planificación, diseño, configuración, ejecución, evaluación y cierre, seguimiento) [73]. Para ttSPL se toma el modelo en V teniendo en cuenta que es un proceso en cascada que desarrolla en paralelo la fase de construcción lo que facilita verificar la trazabilidad a medida que se avanzan en las pruebas y tiene posibilidades para avanzar hacia la próxima fase o volver a visitar la fase anterior. En la figura 17, la parte sombreada se definen las fases que intervienen del modelo en V para ttSPL,

Figura 17. Modelo en V para ttSPL.



3.3. Restricciones de la Técnica ttSPL

ttSPL es una técnica para usar en el desarrollo de líneas de productos, considera la ingeniería de dominio y la ingeniería de productos. Es posible que la técnica ttSPL, pueda ser usada en una línea de productos o en otros enfoques de reutilización, pero sólo ha sido validada en el contexto de la construcción de una línea de productos.

La técnica ttSPL emplea casos de uso como insumo, por lo tanto, no puede usarse cuando el proceso de desarrollo no considere casos de uso, o se debería adaptar el proceso incluyendo este artefacto y las tareas asociadas a su construcción en el contexto de líneas de productos.

La técnica ttSPL emplea el modelo de características para especificar la variabilidad de la línea de productos.

ttSPL se enfoca en pruebas funcionales y no considera otros tipos de pruebas.

La técnica ttSPL podrá ser aplicada con éxito en una pequeña organización de software: (i) si está tiene una SPL definida. Puesto, que ttSPL no indica la forma como se construye una línea de productos, sino que aplica la estrategia de reutilización y derivación de productos de software, para construir los casos de pruebas de los productos que pertenezcan a la

SPL y (ii) si la organización gestiona sus requisitos a través de la especificación de casos de uso, pues ttSPL no contiene pasos definidos para adaptar otro artefacto de especificación de requisitos para aplicar la técnica.

3.4. Fragmentos de Método para la Construcción de la Técnica ttSPL

En la construcción de la técnica ttSPL se ensamblaron partes de técnicas, métodos, y actividades que otros autores han propuesto y que son relevantes para la propuesta. A continuación, en la tabla 14, se listan los fragmentos de métodos seleccionados, y la justificación de la inclusión de dichos fragmentos.

Tabla 14. Fragmentos de bloques utilizados para la técnica ttSPL.

Fuente	Actividad	Fragmento de método seleccionados	Criterios de selección	Observaciones	Seleccionado	Aplicación en ttSPL
Modeling Dependable Product-Families from Use Cases to State Machine Models	Plantilla de casos de uso	Segmentos de la plantilla de casos de uso	Estudio reciente dirigido a pruebas de software con el enfoque SPL		SI	Se refino la plantilla de casos de uso
Templates for textual use cases of software product lines: results from a systematic mapping study and a controlled experiment	Plantilla de casos de uso		Estudio reciente para SPL Disponible Esfuerzo proporcional conocimientos previos	La estructura de la plantilla es amplia y compleja	NO	No aplica
PLUTO: A Test Methodology for Product	plantilla de casos de uso	Segmentos de la plantilla de casos de	Poca disponibilidad Enfoque similar Difícil de entender		NO	No aplica

Fuente	Actividad	Fragmento de método seleccionados	Criterios de selección	Observaciones	Seleccionado	Aplicación en ttSPL
Families		uso	Difícil de aplicar Conocimientos previos			
PLUSS	Tabla PLUSS	Tabla PLUSS del resumen de las relaciones entre características y caso de uso y los puntos variantes	Disponible Enfoque similar Difícil de entender Difícil de aplicar Conocimientos previos		SI	Tabla PLUSS. Tarea: "Asociar casos de uso con características" (ver tabla 17)
Goals and scenarios to software product lines: The GS2SPL approach	refinamiento	Enfoque de refinamiento	Disponibilidad Enfoque similar Aplicable		SI	Tarea: "Refinar modelo de casos de prueba del producto." (ver tabla 28)
Testing a Software Product Line	Variabilidad	Enfoque Variación entre puntos de variación Variación entre productos de software	Disponibilidad Enfoque similar Difícil de entender Conocimientos previos	A través de aquellos segmentos se identifican oportunidades para la reutilización para los artefactos de pruebas	SI	Se representa en el modelo de características
Herramienta para el modelado y configuración de modelos de característi	FeautreIDE	Editor gráfico como textual, permite configuraciones un	Disponibilidad Enfoque similar Fácil de entender Fácil de aplicar	Plugin Eclipse	SI	Graficar el modelo de características

Fuente	Actividad	Fragmento de método seleccionados	Criterios de selección	Observaciones	Seleccionado	Aplicación en ttSPL
cas		modelo de características				
Product derivation in software product families: A case study	Actividades para configurar el modelo de características	Selección de configuración Añada puntos de variación Añadir o cambiar la variante Elimine una variante o un punto de variación	Disponibilidad Enfoque similar Conocimientos previos		SI	Tarea: "Análisis de requisitos y configuración de características del producto." (ver tabla 23)
Evolving feature model configurations in software product lines	Personalizar una SPL a partir de un modelo de características	Análisis de Requisitos y Selección de características	Enfoque similar Conocimientos previos	Se basa en FORM (Método de Reutilización Orientado a las Funciones)	SI	Agregar y/o eliminar características en el modelo de características
Software Evolution: A Features Variability Modeling Approach	Actividades para configurar un modelo de características	Agregar una función compuesta Eliminación de funciones compuestas Cambiar las funciones compuestas	Poca disponibilidad Enfoque similar Conocimientos previos		SI	Configuración de características del producto." (ver tabla 23)
EPF composer	Modelar la técnica propuesta		Disponibilidad Fácil de entender		SI	Modelar el procedimiento ttSPL en ingeniería

Fuente	Actividad	Fragmento de método seleccionados	Criterios de selección	Observaciones	Seleccionado	Aplicación en ttSPL
			Fácil de aplicar			de dominio y el procedimiento ttSPL en ingeniería de aplicación
Key activities for product derivation in software product line	Actividades necesarias para la derivación de un producto	Ingeniería de requisitos Definir la configuración básica Seleccionar los activos	Disponibilidad Enfoque similar		SI	Estrategia de derivación
Testing a Software Product Line	Testing del producto	Derivación de casos de pruebas	Disponible Enfoque similar Fácil de entender Fácil de aplicar		SI	Diseño de casos de pruebas
Models in Software Architecture Derivation and Evaluation: Challenges and Opportunities	Derivar la arquitectura de software	configuración del producto y la instanciación de la arquitectura	Enfoque similar Esfuerzo proporcional		SI	Aplica en concepto para la derivación en SPL
Adoption of Software Product Line from Extreme Derivative Development Process	proceso de desarrollo derivado	proceso de desarrollo derivado	Enfoque similar Esfuerzo proporcional	Describe un proceso de desarrollo derivado (XDDP), que modifica el producto base para	NO	No aplica

Fuente	Actividad	Fragmento de método seleccionados	Criterios de selección	Observaciones	Seleccionado	Aplicación en ttSPL
				construir un nuevo producto, en lugar de Combinando los activos principales		

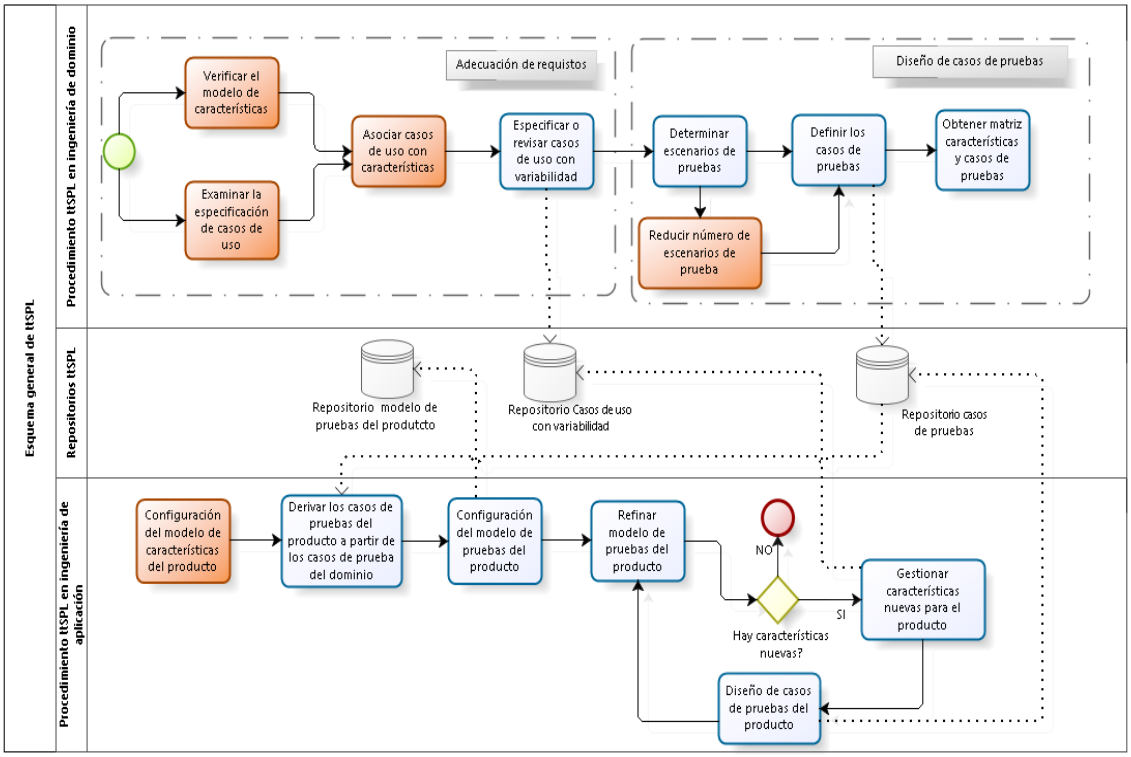
3.5. Técnica ttSPL

Una técnica consiste en “un sistema de reglas orientadas a conseguir un resultado útil”, teniendo en cuenta la habilidad para aplicar dichas reglas [74]. Entonces, para ttSPL una técnica comprende el conjunto de pasos o acciones utilizadas para realizar una determinada tarea de ttSPL, brindando la información necesaria para establecer los pasos específicos y bajo qué condiciones se realizar una determinada tarea de ttSPL. Cabe resaltar que ttSPL no puede ser entendida como un método, debido a que establecer un método requiere de más fases de investigación que comprueben la veracidad de una propuesta, "la ciencia busca la verdad, la técnica la eficacia" [74] y contiene un conjunto de herramientas, técnicas y procesos. Aunque las tareas de ttSPL estén descritas en una secuencia de pasos predefinidos que se pueden entender como un procedimiento, ttSPL no es un procedimiento debido a que un procedimiento se caracteriza por establecer varias formas de cumplir un objetivo (tarea) estableciendo diferentes etas y estructuras diferentes, lo cual ttSPL no tiene dicha característica, pues los pasos deben seguir secuencialmente y tiene una única forma de desarrollar una tarea.

ttSPL se entenderá como una forma para obtener los casos de prueba del dominio y derivar a partir de estos los casos de prueba funcionales para los productos que pertenecen a una línea de productos, utilizando los casos de uso. Aunque, ttSPL utilice estrategias de variabilidad y reutilización que son elementos no funcionales, pero conceptos esenciales del contexto SPL, se considera que ttSPL es una técnica de casos de pruebas funcionales porque tiene como insumo principal los casos de uso que especifican el comportamiento funcional de un producto de software, y cuya finalidad es construir casos de pruebas funcionales a partir de los ellos. También, los casos de uso y casos de pruebas están estrechamente unidos, que tanto el desarrollo de software como el desarrollo de pruebas se encuentran sincronizados si se producen cambios en la especificación de requisitos [66].

La figura 18, presenta el esquema general de ttSPL, la cual se incluye dos procedimientos, el primer procedimiento que se emplea en la ingeniería de dominio, y el segundo procedimiento se emplea en la ingeniería de aplicación.

Figura 18. Esquema de la técnica ttSPL propuesta.



La técnica presentada a continuación corresponde a la primera versión (1.0) del desarrollo de la misma, obtenida como resultado de los refinamientos de versiones preliminares, realizadas a través de los estudios de caso realizados para su validación.

La primera versión preliminar (V0.3) correspondió a una estructura general de la propuesta (ver anexo III: Ejecución del estudio de caso par la línea “arcade game”)

La segunda versión (V0.7) se obtuvo de detallar y unir los fragmentos de método seleccionados, con esta versión se realizó el estudio de caso experimental (ver anexo II: Guías para estudio de caso, sección B) y se evaluó la plantilla de casos de uso. (ver anexo II: Guías para estudio de caso, sección A).

3.5.1. Roles para la Técnica ttSPL

- Ingeniero de dominio: Interactúa con usuarios potenciales y expertos de dominio para establecer el alcance de la línea, identifica requisitos, características y restricciones de los productos que conforman la línea.
- Analista de requisitos: identifica requisitos, características y restricciones de los productos que conforman la línea.
- Ingeniero de producto: es el responsable de la configuración del producto a partir de

la arquitectura, empleando el núcleo base, activos reutilizables y los componentes específicos desarrollados, y de probar que la integración de estos elementos funcione correctamente, para la generación de, las versiones del producto.

- Analista de pruebas: es el responsable de diseñar el modelo de pruebas del producto empleando casos de pruebas del dominio, y verifica que la integración de casos de pruebas funcione correctamente para generar nuevos productos.

3.5.2. Procedimiento ttSPL en Ingeniería de Dominio

En la ingeniería de dominio se desarrollarán los activos reutilizables para los productos de la línea. ttSPL en este apartado describe el procedimiento para la creación de artefactos que permiten el diseño de casos de prueba funcionales que pueden ser reutilizados a través de la derivación y el refinamiento, en la ingeniería de aplicación para un producto específico.

Para describir el plan de producción de ttSPL, se toma como base el propuesto por el Instituto de ingeniería de software (SEI)[19], el cual, es muy genérico y se aplica a todos los productos construidos a partir de los activos del domino seleccionados, en la medida en que se identifican, se definen, se analizan, se diseñan y se prueban los productos. A partir de ellos, ttSPL considera un plan mínimo de producción que contiene la gestión de los activos en cuanto al almacenamiento (repositorio) y documentación (casos de uso y casos de pruebas) de los mismos, también incluye la lista de productos y los modelos de pruebas de los productos construidos. Resaltando que elaborar un plan de producción no está en el alcance del proyecto, este puede servir para gestionar los activos y los productos que se desarrollen en la aplicación de ttSPL.

Figura 19. Actividades a realizar en la ingeniería de dominio.

[19] http://www.sei.cmu.edu/productlines/ppl/production_plan.html

ADECUACIÓN DE REQUISITOS (OPCIONAL).

Los requisitos definen las funcionalidades de los productos de la línea, particularmente sus características comunes y variables, como punto de partida para la derivación de casos de prueba que incluyan en su especificación la variabilidad propia de las SPL. ttSPL toma como fuente la especificación de casos de uso para las funcionalidades y el modelo de características para el manejo de la variabilidad. Sin embargo, de acuerdo a como se hayan definido estos artefactos en el proceso SPL, puede ser que sea necesario adecuarlos para que la técnica aquí propuesta pueda desarrollarse por lo cual estas tareas son opcionales de acuerdo al nivel de semejanza entre los artefactos requeridos por ttSPL y los obtenidos en el proceso de desarrollo de la línea de productos.

La figura 20 muestra el procedimiento general de esta actividad y la figura 21 muestra en detalle cada una de las actividades, los roles responsables de la tarea y los artefactos de entrada y de salida de cada tarea.

Figura 20. Diagrama de actividad de la adecuación de requisitos.

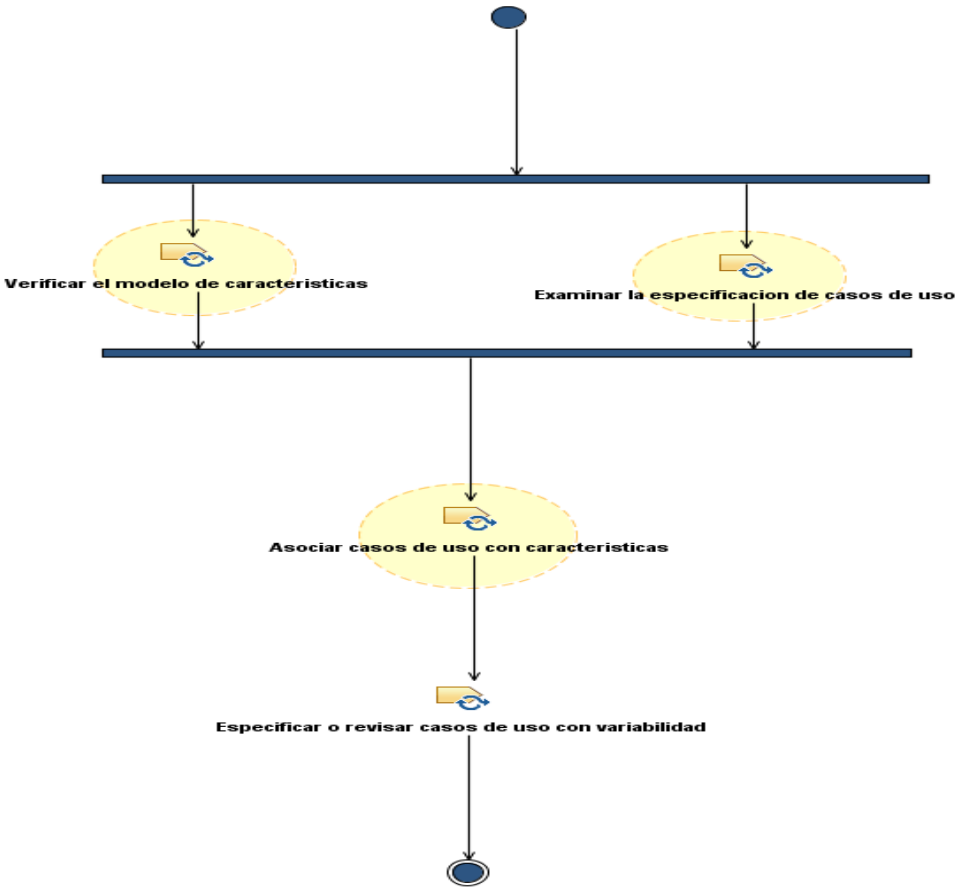
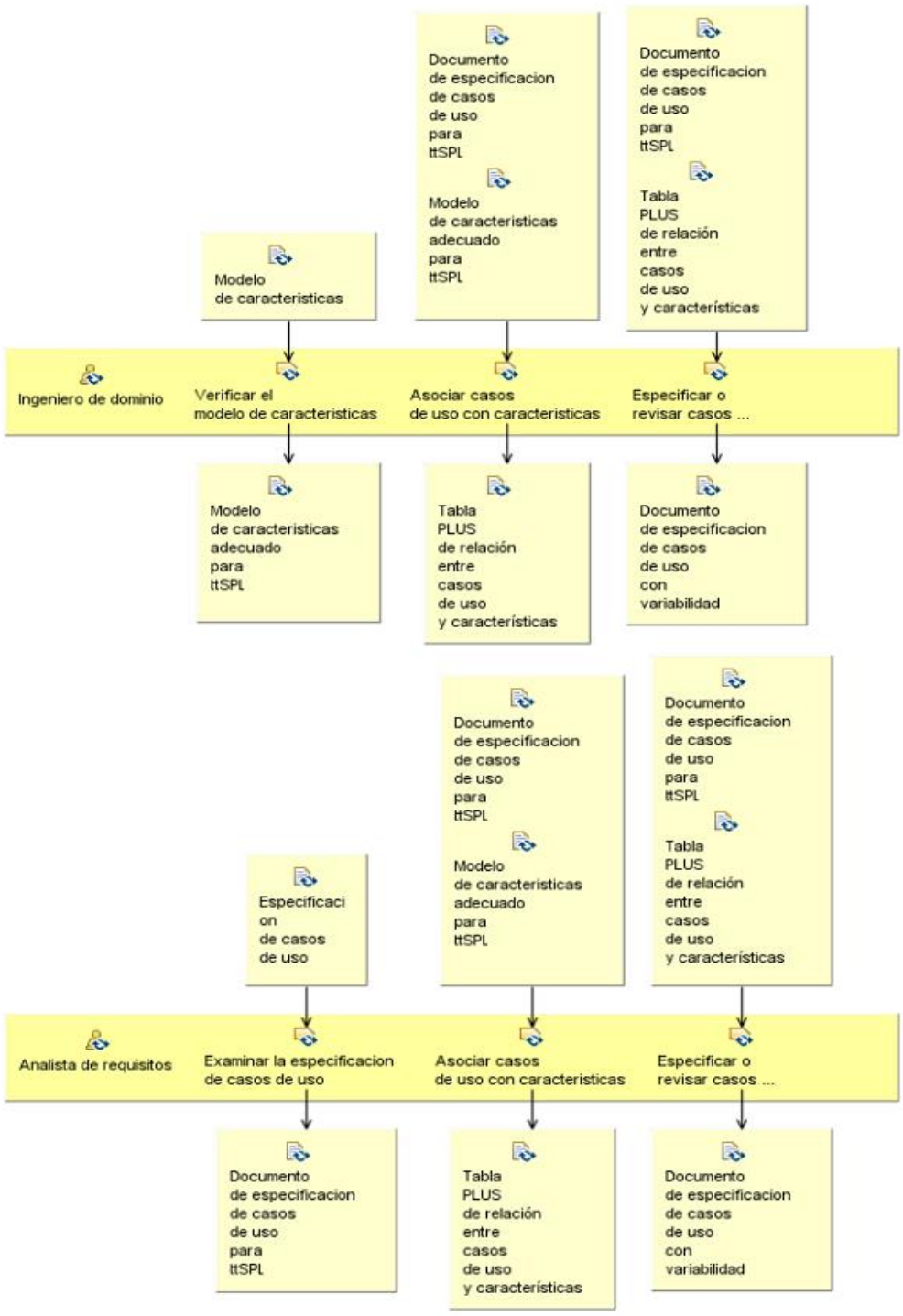


Figura 21. Diagrama detallado de roles, artefactos y tareas para la adecuación de requisitos.



A continuación, se detallan cada una de las tareas pertenecientes a esta actividad adecuación de los requisitos de acuerdo a las necesidades de ttSPL:

1. Verificar el modelo de características (opcional)

El modelo de características es una entrada para ttSPL la cual permite identificar servicios, funcionalidades y requerimientos comunes y variables dentro de una SPL, que permite la configuración de la variabilidad de una manera práctica y comprensible al analista de requisitos.

Tabla 15. Tarea verificar modelo de características.

Tipo:	Tarea opcional.
Nombre:	Verificar modelo de características.
Descripción:	La descripción del modelo de características para el desarrollo de ttSPL toma la versión propuesta en FODA como se describe en [80], que cuenta con una descripción en árbol jerárquico de características. Tomando como base de la representación para las características las siguientes notaciones o etiquetas: **Característica Común:** Representada por un color distinto a las demás características, e identificada por ser la raíz del modelo de características. **Característica Obligatoria**: Representada por un círculo relleno encima del nombre de la característica. Estas características son las que deben estar siempre presentes en los productos de la SPL. **Característica Opcional:** Representada por un círculo blanco encima del nombre de la característica. Esta puede o no ser parte de un producto. **Característica alternativa OR:** Representada por un arco relleno. Si se selecciona la función principal de un grupo alternativo, se puede elegir una o más característica para la derivación del producto. **Característica alternativa XOR:** Representada por un arco sin relleno. Si se selecciona la función principal de un grupo alternativo, se debe elegir solamente una característica para la derivación del producto. La estructura jerárquica de los modelos de características, las relaciones y las restricciones determinan qué combinaciones de características se pueden ensamblar a los productos.
Objetivo:	Utilizar un mecanismo para manejo de la variabilidad de SPL para ttSPL.
Roles:	Analista de requisitos. Ingeniero de dominio.
Artefactos de Entradas:	Modelo de características.
Artefactos de Salidas:	Modelo de características adecuado para ttSPL.
Pasos:	1. Verificar que el modelo de características se encuentre acorde a la definición FODA. 2. Si no se encuentra acorde a la definición: 2.1. Eliminar relaciones y elementos no definidos (como pueden ser cardinalidades, inclusiones y exclusiones empleadas en otros modelos

	de características). 2.2. Cambiar las etiquetas a lo acordado en la definición según su correspondencia entre la descripción del modelo disponible y la definida para ttSPL.

2. Examinar la especificación de casos de uso (opcional)

Los casos de uso son una entrada para ttSPL, la cual permite definir los requisitos funcionales que se han de verificar en los elementos de prueba del sistema, mediante la descripción de una secuencia completa de acciones. Esta tarea busca que la especificación de los casos de uso se ajuste a los parámetros necesarios para aplicar la técnica ttSPL.

Tabla 16. Tarea examinar la especificación de casos de uso.

Tipo:	Tarea opcional.
Nombre:	Examinar la especificación de casos de uso.
Descripción:	La especificación de casos de uso para ttSPL permite identificar de manera completa los actores, condiciones previas, flujos de eventos, y flujos alternativos de un caso de uso.
Objetivo:	Consolidar la especificación de los casos de uso del sistema para que se adapte a la técnica ttSPL.
Roles:	Analista de requisitos.
Artefactos de Entradas:	Documento de especificación de casos de uso.
Artefactos de Salidas:	Documento de especificación de casos de uso para ttSPL.
Pasos:	1. Leer la especificación de casos de uso de la SPL. 2. Tomar el documento de especificación de casos de uso para ttSPL de los artefactos de ttSPL. 3. Para cada uno de los casos de uso identificados en el modelo se debe completar la plantilla de casos de uso.

3. Asociar casos de uso con características (opcional)

Para obtener casos de pruebas completos que representen cada uno de los escenarios de la SPL es necesario asociar los casos de uso con las características de la SPL, identificando la variabilidad asociada a los casos de uso. Una característica puede corresponder a un único caso de uso, un grupo de casos de uso o un punto de variación dentro de un caso de uso.

Tabla 17. Tarea asociar casos de uso con características.

Tipo:	Tarea opcional.
Nombre:	Asociar casos de uso con características.
Descripción:	Para relacionar las características con los casos de uso, ttSPL emplea la tabla descrita en la propuesta PLUS, donde se obtiene una matriz que relaciona cada una de las características descritas en el modelo de características con los casos de uso. A partir de esta matriz se realiza la tabla PLUS que contiene un resumen compacto de la relación entre características y casos de uso. ttSPL, en la ingeniería de dominio, se complementa con etiquetas, para modelar lo común y lo variante en los casos de uso asegurando la trazabilidad entre características y casos de uso, para esto se consolida una nueva definición, categoría de reutilización de los casos de uso, utilizando como etiquetas de identificación: 1. **Común:** para los casos de uso que se relacionan con la característica raíz del modelo de características. 2. **Opcional.** Para los demás casos de uso.
Objetivo	Identificar puntos de variación asociados a un caso de uso e identificar las relaciones entre casos de uso y características.
Roles:	Ingeniero de dominio.
Artefactos de Entradas:	Documento de especificación de casos de uso para ttSPL. Modelo de características adecuado para ttSPL.
Artefactos de Salidas:	Tabla PLUS de relación entre casos de uso y características.
Pasos:	1. Abrir documento de creación de la tabla PLUS de los artefactos de ttSPL. 2. Identificar todas las características pertenecientes a la SPL. 3. Identificar todos los casos de uso pertenecientes a la SPL. 4. Relacionar casos de uso con características, para hacerlo se diligenciará la matriz de características vs casos de uso (ver Anexo I: Guía ttSPL, sección A). 5. Concretar la relación entre casos de uso y características identificando las categorías de las características. Las categorías de reutilización de los casos de uso y los puntos variantes, para realizar este paso se presenta las indicaciones para su diligenciamiento en la guía para Asociar casos de uso con características (ver Anexo I: Guía ttSPL, sección A).

4. Especificar o revisar casos de uso con variabilidad.

Después de identificar la variabilidad asociada a los casos de uso y las relaciones entre estos y las características se precisa formalizar dicho contenido. La característica común hallada en la tabla PLUS de la línea de productos está dada por el núcleo del caso de uso. La variabilidad del caso de uso se maneja con la noción de punto de variación. Un punto de variación es un lugar en un caso de uso donde un cambio puede tener lugar. Por lo tanto, parte de la descripción del caso de Uso captura la parte común de productos (la secuencia principal y las alternativas del caso de uso), y parte de ella captura la variabilidad de la línea de productos (la descripción de los puntos de variación). Esto es distinto de casos de uso opcional, donde la descripción es totalmente de la variabilidad de la línea de productos.

Tabla 18. Tarea especificar casos de uso con variabilidad.

Nombre:	Especificar casos de uso con variabilidad.
Tipo:	Tarea.
Descripción:	La especificación de casos de uso con variabilidad para ttSPL permite identificar de manera completa los casos de uso de la SPL teniendo en cuenta que cada uno de ellos se ha relacionado de alguna manera a las características que representan la variabilidad dentro de SPL. En cada especificación de caso de uso con variabilidad, deben considerarse las características que afectan al caso de uso y sus puntos de variación. Una especificación de casos de uso con variabilidad que está asociada con dos o más características puede describir una dependencia implícita o una interacción de características. Una dependencia implícita es una dependencia de características que no se describe en el modelo de características. Una dependencia en los requisitos funcionales ocurre cuando la selección de una característica permite o excluye el comportamiento funcional asociado con la selección de otra característica. Una interacción de características es un comportamiento funcional habilitado para una combinación de características seleccionada para una aplicación derivada de la SPL, pero que no está habilitada cuando se selecciona por separado alguna característica de la combinación.
Objetivo:	El objetivo de esta tarea es capturar las funcionalidades comunes y variables de manera concreta a partir la descripción de los escenarios configurados en la tabla PLUS y la interacción entre casos de uso y características con el fin de realizar una correcta especificación que asegure la integridad de las pruebas derivadas en el dominio de manera que representen la variabilidad de la SPL.
Roles:	Analista de Requisito. Ingeniero de Dominio.
Artefactos de Entradas:	Documento de especificación de casos de uso. Tabla PLUS de relación entre casos de uso y características.
Artefactos de Salidas:	Documento de especificación de casos de uso con variabilidad.
Pasos:	1. Abrir el documento de especificación de casos de uso con variabilidad que se encuentra en los artefactos de ttSPL. 2. Identificar en el documento de especificación de casos de uso la descripción, los campos generales del caso de uso y el flujo básico y los alternativos que corresponden. 3. Identificar en la tabla PLUS las características asociadas al caso de uso, su categoría de reutilización y cada uno de los puntos variantes que tiene asociados el caso de uso. No todos los casos de uso tienen asociados puntos variantes. 4. Diligenciar la plantilla propuesta en el documento de especificaciones de caso de uso con variabilidad para cada una de las especificaciones de casos de uso.

DISEÑO DE CASOS DE PRUEBA

Los casos de prueba en la técnica ttSPL incluye procedimiento y resultados esperados para la ejecución de una prueba funcional del software, donde los casos de uso son claves para el proceso de prueba, debido a que identifican y dan a conocer las condiciones bajo las que deben ser implementadas las pruebas, y son necesarios para verificar que se han implementado satisfactoriamente y con calidad los requisitos del producto resultante.

En la figura 22 muestra el procedimiento general de esta actividad y en la figura 23, muestra en detalle cada una de las actividades, los roles responsables de las tareas.

Figura 22. Diagrama de actividad del diseño de casos de prueba.

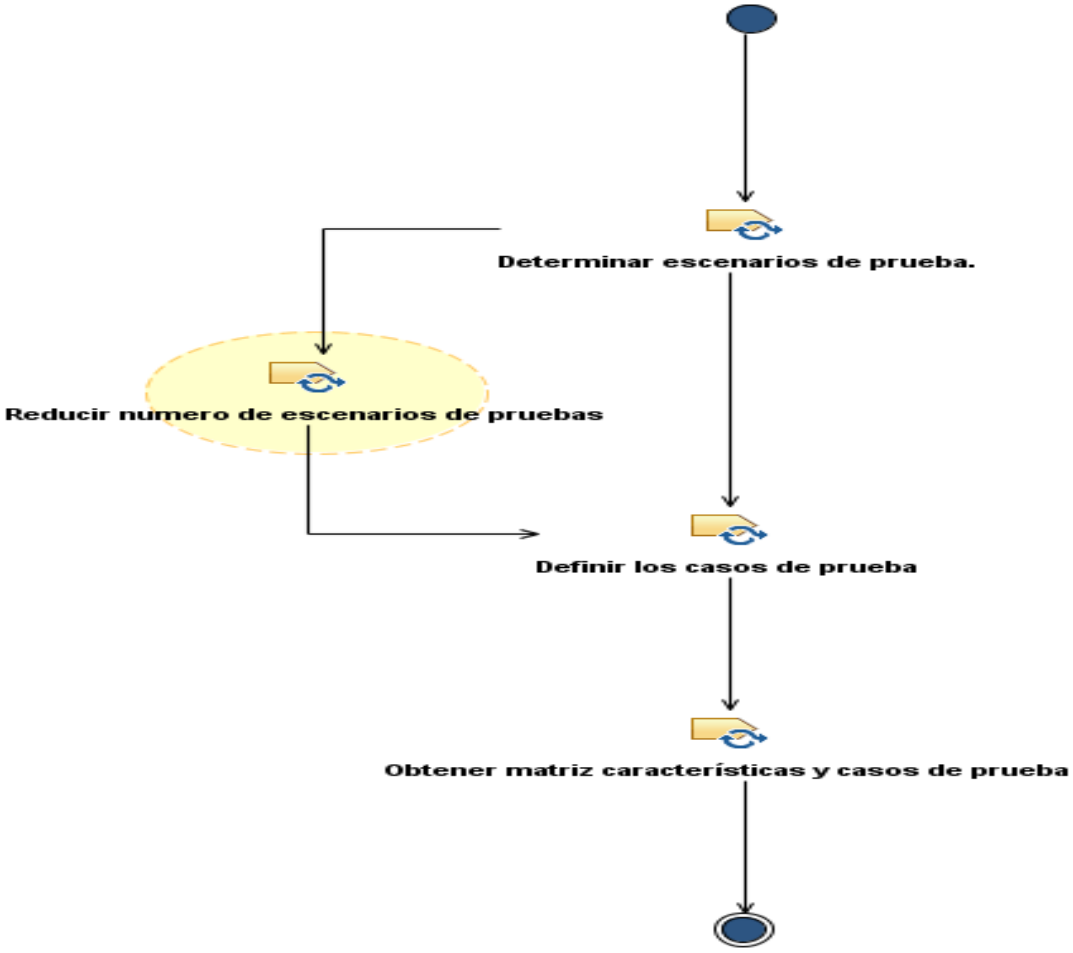
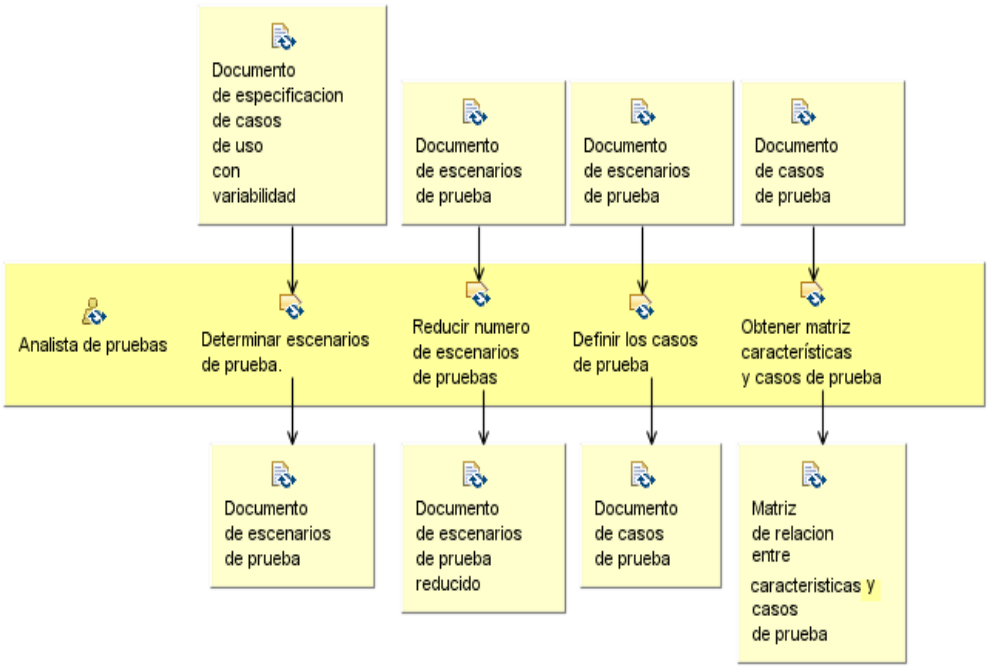


Figura 23. Diagrama detallado de roles, artefactos y tareas para el diseño de casos de prueba.



A continuación, se detallan cada una de las tareas pertenecientes a esta actividad diseño de pruebas.

1. Determinar Escenarios de prueba.

Los escenarios de prueba en ttSPL representan cada uno de los posibles flujos a partir de los cuales se pueden obtener casos de prueba.

Tabla 19. Tarea determinar los escenarios de prueba.

Tipo:	Tarea.
Nombre:	Determinar los escenarios de prueba.
Descripción:	En esta tarea se identifican los escenarios de prueba posibles a partir de la especificación de casos de uso con variabilidad para cada uno de los flujos que determinan el comportamiento funcional del sistema.
Objetivo:	El objetivo de esta tarea es identificar cada uno de los escenarios posibles del sistema con el fin de garantizar la completitud y trazabilidad de las pruebas.
Roles:	Analista de pruebas.
Artefactos de Entradas:	Documento de especificación de casos de uso con variabilidad.
Artefactos de Salidas:	Documento de escenarios de prueba.

Pasos:	1. Abrir el documento escenario de pruebas que es un artefacto de ttSPL. 2. Utilizar el documento de especificación de casos de uso con variabilidad. 3. Tomar cada una de las plantillas del documento de casos de uso con variabilidad e identificar los flujos. Estos se representan en una serie de pasos que se especifican en el documento de escenarios de prueba. Cada flujo se identifica de la siguiente manera: a. El flujo básico se identifica en la plantilla como flujo básico y se toma tal y como esta descrito en la especificación de casos de uso con variabilidad. b. Los flujos alternativos se componen del flujo básico y el(los) paso(s) del flujo alternativo unidos por un identificador del paso del cual se desprende el flujo alternativo. c. Los flujos opcionales se componen del flujo básico y la funcionalidad del punto variante unidos por un identificador del paso del cual se desprende el punto variante. Estos a diferencia de los alternativos solo representan una funcionalidad y al tomar dicha funcionalidad el flujo regresa a los pasos del flujo básico. Por cada punto variante se crea un flujo. 4. Se diligencia el documento de escenarios de pruebas con cada uno de los campos propuestos, describiendo los flujos con las mismas palabras presentes en la especificación de casos de uso con variabilidad y tomando la acción del actor y la respuesta del sistema.

2. Reducir número de escenarios de pruebas (opcional)

La lista de escenarios de prueba obtenida resulta ser extensa, por tanto, con el ánimo de optimizar la aplicación de ttSPL se permite la reducción de dichos escenarios de prueba.

Tabla 20. Tarea reducir los de escenarios de prueba.

Nombre:	Reducir los de escenarios de prueba.
Tipo:	Tarea opcional.
Descripción:	ttSPL en busca de optimizar, en la práctica, el tiempo y recursos disponibles para el desarrollo de la prueba, define una estrategia de reducción del conjunto de pruebas. La ITU-T propone una serie de estrategias que permiten la reducción de pruebas en la Recommendation Z.500 Framework on formal methods in conformance testing (1998), esta recomendación se tomó como base para proponer esta tarea.
Objetivo:	El objetivo de esta tarea es reducir la cantidad de escenarios de prueba para optimizar la técnica en recursos y tiempo.
Roles:	Analista de pruebas.
Artefactos de Entradas:	Documento de escenarios de prueba.
Artefactos de Salidas:	Documento de escenarios de prueba reducido
Pasos:	A partir de la recomendación se tomaron las siguientes prácticas: 1. Crear una copia del documento de escenarios de pruebas.

	2. Identificar y marcar flujos con comportamientos que pueden considerarse similares. 3. Realizar la unificación de los comportamientos que pueden considerarse similares en unos solo. 4. A partir de los escenarios resultantes es posible seleccionar un subconjunto de escenarios de prueba aleatorio del conjunto de escenarios de prueba posible.

3. Definir los casos de prueba

En esta sección se formalizan los escenarios de prueba identificados, con el fin de obtener el caso de prueba completo y listo para la ejecución.

Tabla 21. Tarea definir los casos de pruebas.

Nombre:	Definir los casos de prueba.
Tipo:	Tarea.
Descripción:	El analista de pruebas en esta tarea debe diseñar todos los casos de prueba definidos, empleando la plantilla de casos de prueba, de tal forma que los casos de prueba puedan derivarse para la posterior ejecución en la ingeniería de producto.
Objetivo:	Diseñar casos de prueba.
Roles:	Analista de pruebas.
Artefactos de Entradas:	Documento de escenarios de prueba. Documento de escenario de pruebas reducido (si este ha sido realizado)
Artefactos de Salidas:	Documento de casos de prueba.
Pasos:	1. Abrir el documento de escenarios de prueba o el documento reducido de escenarios de prueba, según corresponda. 2. Abrir el documento de casos de prueba de los artefactos de ttSPL. 3. Para cada uno de los escenarios del documento de escenarios de prueba: a. Crear una copia de la plantilla de formalización de casos de prueba en una nueva hoja del documento. b. Crear un identificador un nombre para el caso de prueba relacionado a su funcionalidad y definir la versión del caso de prueba. c. Renombrar la hoja con el identificador del caso de prueba. d. Copiar los pasos del escenario a la plantilla de acuerdo al formato propuesto. 4. Renombrar el documento de manera que permita identificar que son los casos de prueba de determinada SPL.

4. Obtener matriz de relación entre características y casos de pruebas

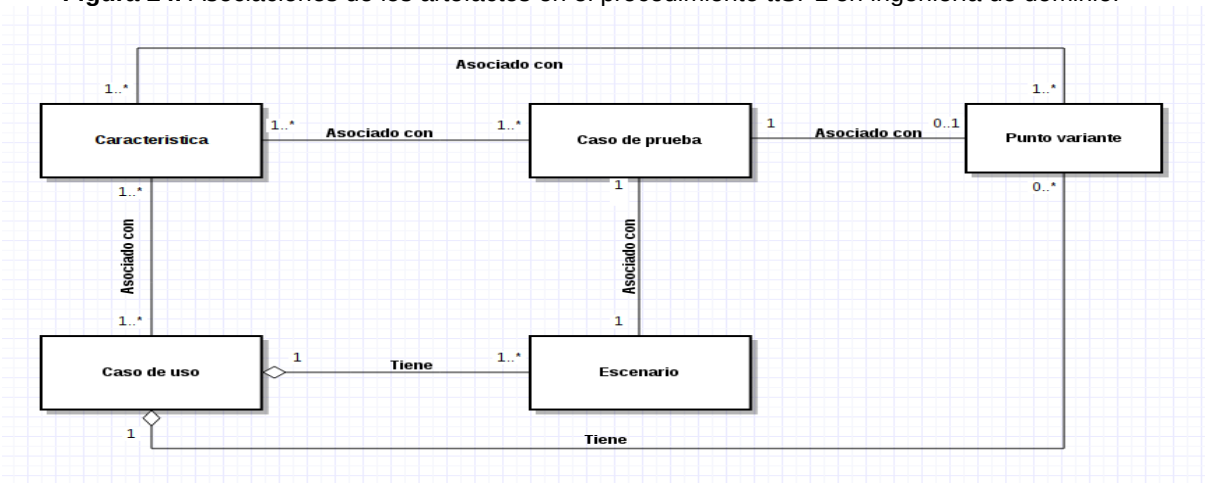
En esta tarea se analizan las relaciones de las características con las especificaciones de prueba para reducir el número de configuraciones de aplicación a probar, sin omitir ninguna combinación de características relevantes.

Tabla 22. Tarea obtener matriz de relación entre características y casos de prueba.

Nombre:	Obtener matriz de relación entre características y casos de prueba.
Tipo:	Tarea.
Descripción:	Las asociaciones de enlace entre características y casos de prueba durante la ingeniería de ttSPL, permiten que estos casos de prueba se seleccionen, agrupados en la especificaciones de prueba a la que pertenecen, durante la derivación de pruebas basada en características para una aplicación de ttSPL; siendo, las asociaciones de enlace entre características y puntos de variantes, durante la derivación de pruebas basadas en características, los que permiten personalizar las especificaciones de prueba para una aplicación específica.
Objetivo:	Este procedimiento permite al análisis de interacciones de las características con los casos de prueba, la especificación de prueba asociada y el con un escenario de caso uso con variabilidad del cual hace parte.
Roles:	Analista de pruebas.
Artefactos de Entradas:	Documento de casos de prueba.
Artefactos de Salidas:	Matriz de relación entre características y casos de prueba.
Pasos:	1. Abrir el documento de casos de prueba. 2. Crear un documento con una matriz de relación entre características y casos de prueba. 3. Para cada caso de prueba identificar la(s) característica(s) asociada(s). 4. Realizar una matriz con los identificadores (id) de los casos de prueba en las filas y las características de todo el modelo de características de SPL en las columnas. 5. Identificar con una marca las características asociadas a cada caso de prueba.

En la figura 24, se muestran las diferentes asociaciones entre los artefactos que configuran la ingeniería de dominio.

Figura 24. Asociaciones de los artefactos en el procedimiento ttSPL en ingeniería de dominio.

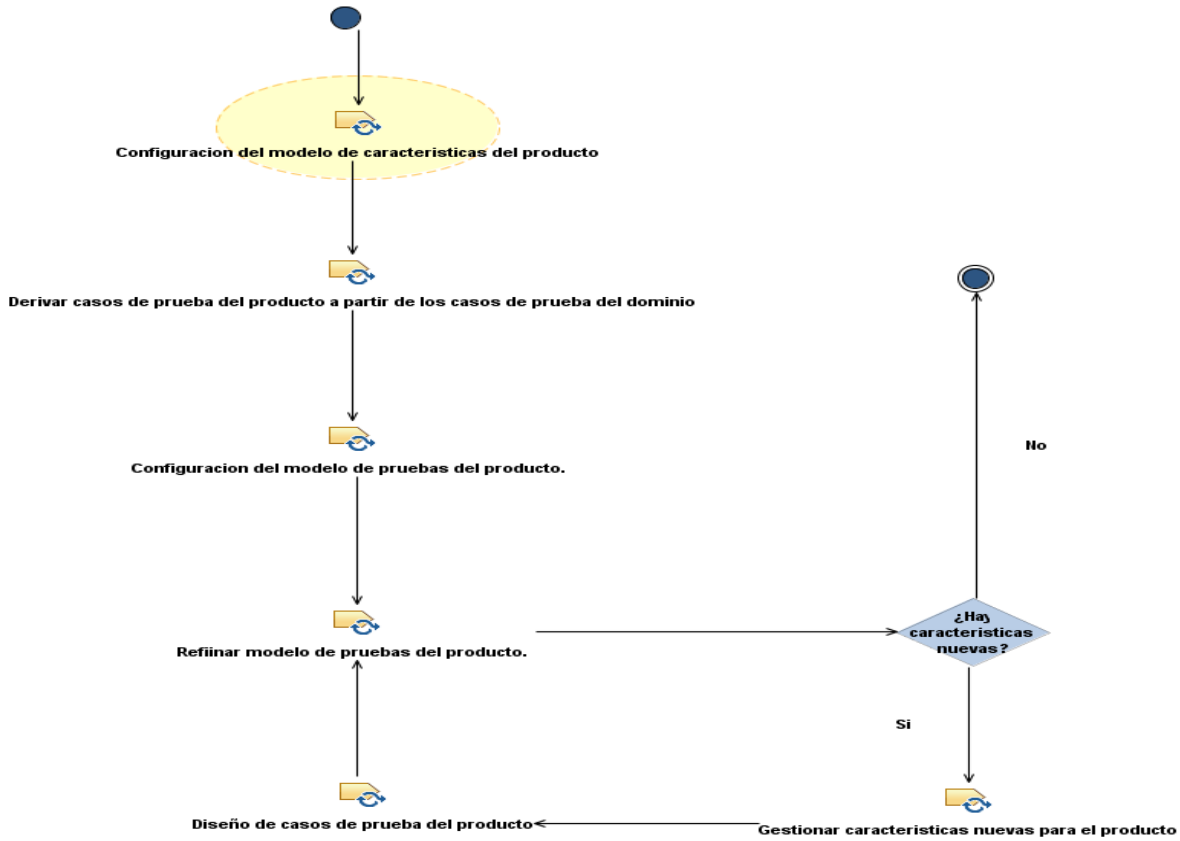


3.3.3. Procedimiento ttSPL en Ingeniería de Aplicación

La esencia de la ingeniería de aplicación es construir un producto que pertenezca a la SPL, mediante la reutilización de tantos activos del dominio como sea posible, utilizando las similitudes y variabilidades de la línea para desarrollar un producto en particular [67][81]. Es así, que en este apartado se presentará la técnica de ttSPL para facilitar la derivación de casos de pruebas funcionales específicos de un producto, a partir de los casos de prueba del dominio, y de los casos de uso del producto desarrollados desde los casos de uso del dominio.

A continuación, se presenta en la figura 25, la estrategia de derivación y refinamiento que se empleará en la ingeniería de aplicación para obtener los casos de pruebas funcionales que facilitarán las pruebas de software un producto de una SPL. Para ello, la técnica ttSPL propone dos opciones de derivación para construir los casos de pruebas del producto, una de esas opciones es derivar a partir de la configuración del modelo de características el modelo de casos de pruebas del producto, y la segunda opción es a partir de la configuración del modelo de características crear un modelo de casos de uso del producto y posteriormente crear los casos de pruebas del producto.

Figura 25. Estrategia de derivación y refinamiento para la ingeniería de aplicación de ttSPL.



1. Configuración del modelo de características (opcional)

En general, los productos de la línea son construidos a partir de artefactos reutilizables que se construyeron en la ingeniería de dominio de la SPL, por lo tanto, el modelo de características de un producto se obtiene a partir del modelo de características de la línea que captura los puntos de variación, que permite agregar, eliminar o seleccionar características en la configuración del producto y los requisitos del producto.

Esta tarea no hace parte directa de la técnica de pruebas, la organización puede hacerlo con el procedimiento que incluya su proceso de desarrollo de líneas de productos, se incluye para apoyar el proceso de ttSPL

Consecutivamente, se definen las siguientes actividades para obtener la configuración del modelo de características:

Tabla 23. Tarea análisis de requisitos y configuración de características del producto.

Nombre	Análisis de requisitos y configuración de características del producto.
Tipo	Tarea opcional.
Descripción	El análisis de requisitos procede de la indagación de las necesidades del usuario y la selección de características depende del modelo de características del dominio, una forma de facilitar la selección es realizarla de abajo hacia arriba ya que el modelo de características tiene forma de árbol o indica jerarquía. Lo anterior, permite identificar el modelo de características base, para completar el desarrollo del modelo de características en la ingeniería de aplicación. Se toma como entrada el modelo de requisitos que en las actividades previas a la prueba de software fueron establecidas para el producto. A cada configuración del modelo de características se le debe dar un identificador para asociar la configuración a un producto de la SPL. Etiqueta: A cada modelo de características del producto se le asigna una etiqueta llamada “id”: <<MF_Pr_Número>>, donde Pr_Número corresponde al “id” del producto. Ejemplo: Id:MF_Pr01.
Objetivos	Configurar el modelo de características del producto.
Roles	Ingeniero del producto.
Artefactos de entrada	Modelo de características para ttSPL del dominio. Requisitos del producto.
Artefactos de salida	Modelo de características del producto.
Paso	1. Con el análisis de los requisitos y partiendo del modelo de características para ttSPL del dominio, seleccionar características que represente el producto. 2. Si existen funcionalidades que no se pueden tomar del modelo de características del dominio, agregar características nuevas al modelo de

	características del producto, identificando bien cada uno de las relaciones jerárquicas que representan los modelos de características. 3. Eliminar características y/o sub-características que no representen el producto. (limpiar el modelo de características, eliminado aquellas que no representen el producto). Nota: En la guía "para ttSPL" (ver Anexo I: Guía ttSPL, sección B), se explica con más detalle la descripción de cada paso.

2. Derivar casos de pruebas del producto a partir de casos de prueba del dominio

El modelo de características del producto comparte su estructura con el modelo de características del dominio, por tanto, es posible encontrar el conjunto de casos de pruebas asociado a la especificación del producto de las características que comparte con el modelo de características del dominio. Para ello, se reutilizan los casos de pruebas desarrollados en la ingeniería de dominio identificando las características asociadas del producto dadas por el modelo de características del producto.

Tabla 24. Tarea derivar caso de pruebas del producto a partir de caso de pruebas del dominio.

Nombre	Derivar caso de pruebas del producto a partir de caso de pruebas del dominio.
Tipo	Tarea.
Descripción	Las SPL permiten derivar los artefactos del producto de la parte común y de aquellas características compartidas entre el modelo de características del dominio y el modelo de características del producto. Teniendo en cuenta que la SPL contiene un número indeterminado de productos posibles, y estos están dados por la selección de características. Es posible que un nuevo producto se encuentre completamente inmerso en las características que se han definido en el modelo de características del dominio, por tanto, los casos de prueba obtenidos del dominio serían suficientes para representar los casos de prueba del producto.
Objetivos	Identificar los casos de prueba del dominio que pueden ser reutilizados en el nuevo producto configurado.
Roles	Analista de pruebas.
Artefactos de entrada	Modelo de características del producto. Matriz de características y casos de prueba.
Artefactos de salida	Lista con identificadores de los casos de prueba derivados para el producto.
Paso	1. Identificar las características del modelo de características del producto y buscar en la matriz de características y casos de prueba los casos de prueba asociados. Si una característica no se encuentra sola significa que debo buscar por grupos de características; el caso de prueba me servirá siempre y cuando el producto contenga las características que están en el grupo de características asociadas al caso de prueba. 2. Realizar una lista con los identificadores de los casos de casos de prueba identificados con las características del producto.

3. **Configurar el modelo de casos de pruebas del producto**

El modelo de casos de pruebas permite definir las pruebas que se realizarán sobre los elementos a verificarse, en este caso las pruebas funcionales del producto.

Tabla 25. Tarea configurar el modelo de casos de pruebas del producto.

Nombre	Configurar el modelo de casos de pruebas del producto.
Tipo	Tarea.
Descripción	Esta tarea permite la especificación formal de los casos de prueba identificados para un producto, obtenidos mediante la derivación y desarrollados en la ingeniería de dominio. Obteniendo un entregable que permite identificar al tester o ejecutor de pruebas cuales casos de prueba debe ejecutar.
Objetivos	Especificar el modelo de pruebas del producto.
Roles	Analista de pruebas. Ingeniero del producto.
Artefactos de entrada	Lista con identificadores de los casos de prueba derivados para el producto.
Artefactos de salida	Modelo de casos de prueba del producto.
Paso	1. Realizar una copia de la plantilla del modelo de casos de prueba. 2. Abrir la plantilla de modelo de casos de prueba del producto. a. Agregar un nombre y un identificador para el producto. b. Realizar una descripción del producto. c. Agregar el identificador del modelo de características del producto. d. Incluir la lista de identificadores de casos de prueba derivados para el producto en la sección correspondiente.

4. Refinar el modelo de caso de pruebas del producto

El refinamiento en ttSPL permite obtener modelos de prueba completos que representan el producto, teniendo en cuenta cada una de las nuevas características o funcionalidades que son particulares para el producto.

Tabla 26. Tarea refinar modelo de casos de prueba del producto.

Nombre	Refinar modelo de casos de prueba del producto.
Tipo	Tarea.
Descripción	Esta tarea permite actualizar el modelo de casos de pruebas de un producto con las nuevas características agregadas.
Objetivos	Refinar el modelo de casos de pruebas del producto.
Roles	Analista de pruebas.
Artefactos de entrada	Lista de identificadores de casos de prueba nuevos para el producto. Modelo de casos de prueba del producto.
Artefactos de salida	Modelo de casos de prueba del producto actualizado.
Paso	1. Agregar la lista de identificadores de casos de prueba nuevos para el producto a la sección del modelo de casos de pruebas del producto donde

	se identifican los casos de prueba asociados al producto.

5. Gestionar características nuevas para el producto

Los productos en la ingeniería de aplicación cuentan con activos reutilizables especificados en el dominio, pero además estos productos pueden contar con características propias del producto, estas configuran nuevos activos que dependiendo de la gestión de la SPL pueden o no llegar a ser activos reutilizables, en este caso se reutilizan los artefactos necesarios para la configuración de nuevos activos.

Tabla 27. Tarea gestionar características nuevas del producto.

Nombre	Gestionar características nuevas del producto.
Tipo	Tarea
Descripción	Las nuevas características pueden ser identificadas como características funcionales, que son aquellas que representan una nueva funcionalidad completa dentro del comportamiento del producto y características de puntos variantes que son aquellas que se adhieren a un flujo ya determinado en el dominio como una nueva alternativa a las ya propuestas y desarrolladas en los puntos variantes de la línea. Las características funcionales configuran un nuevo caso de uso, este debe ser especificado correctamente, en una nueva plantilla de especificación de casos de uso con variabilidad, así no posea variabilidad, donde el nombre de la característica será su característica asociada y opcional será su categoría de reutilización. Para las características de puntos variantes es necesario identificar mediante las relaciones propias de la ingeniería del dominio, representadas en la figura 24, la especificación de casos de uso con variabilidad asociada. Esta especificación de casos de uso con variabilidad será la partida para la creación de los nuevos activos necesarios para el producto, en este caso los casos de prueba. Mediante esta tarea se identifica y se clasifica cada una de las características nuevas y propias del producto y que determinan un caso de uso con variabilidad para la obtención de los activos propios de la característica.
Objetivos	Determinar el caso de uso con variabilidad para la nueva característica.
Roles	Analista de pruebas. Ingeniero del producto.
Artefactos de entrada	Modelo de características del producto. Documento de especificación de casos de uso con variabilidad. Modelo de características para ttSPL.
Artefactos de salida	Especificación de caso de uso con variabilidad para el producto.
Paso	1. Contrastar el modelo de características del producto y el modelo de características para ttSPL para identificar la característica nueva. 2. Clasificar la nueva característica como característica funcional o punto variante. a. Si es una característica funcional se toma el documento de especificación de casos de uso con variabilidad y se agrega una

	nueva especificación (crear y llenar nueva plantilla). i. La característica asociada será la nueva característica. ii. La categoría de reutilización del caso de uso será opcional. iii. Completar los demás campos de la plantilla de acuerdo a los requisitos del producto. b. Si es una característica de punto variante se debe identificar el caso de uso asociado al árbol que pertenece la nueva característica y se debe seleccionar dicha especificación de caso de uso con variabilidad. i. Se agrega a la especificación de casos de uso con variabilidad un nuevo punto variante llenando cada uno de los campos. ii. El tipo de característica para el punto variante será el tipo de característica alternativa al que pertenecen los demás puntos variantes que se derivan de dicho punto de variación. 3. Obtener una copia de especificación de caso de uso con variabilidad identificada.

6. Diseño de caso de prueba del producto

La reutilización en ttSPL está representada por los casos de prueba del dominio utilizados en productos, pero también en el uso de artefactos o actividades necesarias para obtener los casos de prueba refinados o completos para cada producto, por lo tanto, esta tarea es la reutilización de la actividad destinada al diseño de casos de prueba y de sus artefactos.

Tabla 28. Tarea diseño de caso de pruebas del producto.

Nombre	Diseño de caso de pruebas del producto.
Tipo	Tarea.
Descripción	La especificación de casos de uso con variabilidad me permite ejecutar la actividad de diseño de las pruebas del dominio para obtener nuevos casos de prueba correspondientes al producto. Esto permite refinar el modelo de pruebas del producto haciendo el conjunto de pruebas más completo y de esta manera garantizar la calidad del producto.
Objetivos	Diseñar casos de prueba para las nuevas características del producto.
Roles	Analista de pruebas.
Artefactos de entrada	Especificación de caso de uso con variabilidad para el producto. Documento de escenarios de prueba. Documento de casos de prueba.
Artefactos de salida	Lista de identificadores de casos de prueba nuevos para el producto.

Paso	1. Actualizar el documento de escenarios de pruebas con los nuevos escenarios de prueba identificados en la especificación de casos de uso con variabilidad. Para identificar de manera correcta los escenarios de prueba se recomienda ver la tarea correspondiente en especificada en la ingeniería de dominio. 2. Actualizar el documento de casos de prueba creando la definición de casos de prueba para los nuevos escenarios identificados. Se debe tener en cuenta la tarea para definir casos de prueba del dominio. 3. Actualizar la matriz de casos de prueba y características con los nuevos casos de prueba y las nuevas características. 4. Crear una lista con los identificadores de los casos de prueba nuevos para el producto.

CAPÍTULO IV

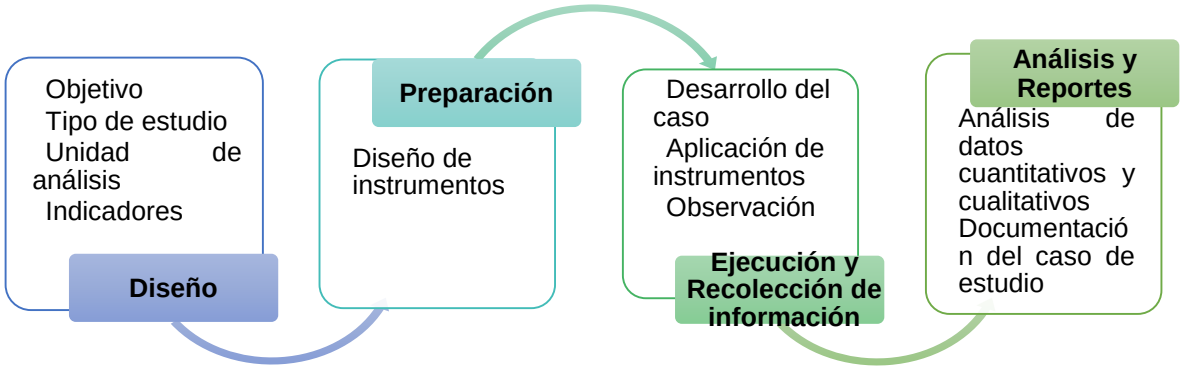
EVALUACIÓN DE LA TÉCNICA PROPUESTA

En este capítulo se presenta la aplicabilidad de una técnica para la construcción de casos de pruebas funcionales basados en casos de uso en el contexto SPL en pequeñas organizaciones - ttSPL, elaborada en el capítulo III de este documento. Ya que, al proponer una técnica como es el caso de este proyecto, es necesario evaluarla tanto la integridad de su especificación, así como su aplicabilidad y consistencia práctica en proyectos de desarrollo reales.

Para realizar una evaluación de la técnica propuesta, que permita prever si es o no aplicable en pequeñas organizaciones, se utilizó el método de investigación de estudio de casos, con aplicaciones a diferentes contextos. La investigación en la ingeniería de software mediante estudios de caso, tiene como objetivo estudiar cómo el proceso de desarrollo es realizado por los ingenieros de software y las partes interesadas en su entorno [26]. Para el diseño de la investigación se partió de varias preguntas de investigación muy relacionadas. Primero, la pregunta básica de investigación planteada en el proyecto ¿Cómo facilitar la derivación de los casos de prueba específicos de un producto a partir de los casos de uso y casos de prueba de dominio en el contexto de SPL en una pequeña organización de software?, y de ésta se han derivado las preguntas concretas para cada uno los tres estudios de caso ejecutados.

Para la realización de los estudios de caso se siguieron los pasos presentados en la figura 26, que ilustra los pasos generales seguidos en los estudios de caso empleados en la evaluación de la técnica propuesta.

Figura 26. Proceso general seguido en los estudios de casos[20].



[20] Basado en [19].

Se consideraron 3 estudios de caso y una revisión con un experto en pruebas de software:

- Estudio de caso 1 – Evaluación plantilla de casos de uso con variabilidad

Holístico[21], exploratorio[22], unidad de análisis[23].

El objetivo de este estudio de caso fue explorar la forma de aplicación de conceptos y prácticas del enfoque de producción de SPL, a través de la plantilla de casos de uso para la técnica propuesta, por la importancia de este artefacto por ser el punto de partida para la implementación de la técnica ttSPL.

- Estudio de caso 2- Caso exploratorio: Aplicación de ttSPL V0.3 en la línea de productos Pedagógica "Arcade Game[24]"

Holístico, Exploratorio

El objetivo de este estudio fue observar y contrastar los lineamientos de la técnica ttSPL con los lineamientos del instituto de ingeniería de software (SEI), que maneja una línea de productos denomina "Arcade game", a través de la exploración sobre un juego o producto de dicha línea.

- Estudio de caso 3: Aplicabilidad y consistencia de la técnica propuesta en una pequeña organización.

Descriptivo[25], unidad de análisis

El objeto de este estudio fue determinar si la técnica ttSPL es aplicable y consistente dentro de una pequeña organización de software, utilizando la línea de productos desarrollada por la organización.

- Revisión con un experto en pruebas de software: Evaluación de los artefactos que pertenecen a la técnica ttSPL

El objetivo de este estudio es validar la aplicabilidad y consistencia de los artefactos de ttSPL con un agente experto en pruebas de software.

Estudio de caso 1: Evaluación plantilla de caso de uso con variabilidad

Para iniciar con la especificación de la técnica propuesta, en proyectos reales de una pequeña organización desarrolladora de software, se consideró el proyecto "micro-juegos" en el marco de juegos serios, con el fin de abordar situaciones de recursos humanos de la empresa, la cual se especializa en la fabricación y comercialización de equipos de medición de flujo de alta precisión y prestación de soporte técnico, según lineamientos de la norma internacional ISO 9001 y la mejora continua de sus procesos de desarrollo.

---

[21] Un estudio de caso holístico, estudia el caso en conjunto [86].

[22] Exploratorio, se emplea para descubrir lo que está sucediendo, en busca de nuevas ideas y la generación de ideas e hipótesis para nuevas investigaciones [87].

[23] "En ingeniería de software la unidad de análisis puede ser un proyecto de desarrollo, un equipo o una persona individual" [2].

[24] http://www.sei.cmu.edu/productlines/ppl/

[25] Descriptivo, detalla una situación o fenómeno [87].

1. **Pregunta de investigación:** A partir de la pregunta de investigación del proyecto, era necesario analizar la elección de la plantilla de caso de uso que sería utilizada durante la ejecución de la técnica propuesta, con la posibilidad real de que una pequeña organización verificará si la selección era la más adecuada para la propuesta a desarrollar. La pregunta para este estudio de caso es: ¿La plantilla para la especificación de casos de uso para la técnica propuesta, es apropiada para facilitar el diseño y la derivación de los casos de pruebas funcionales?
2. **Objetivo del estudio de caso:** El objetivo de este estudio es determinar la aplicabilidad de la plantilla de caso de uso para el proyecto de investigación, dentro de una pequeña organización.
3. **Selección del estudio caso:** de acuerdo a Runeson y Höst [19] este estudio de caso es holístico y descriptivo, con una unidad de análisis, la cual corresponde a una práctica de casos de uso, la aplicación de la plantilla por un grupo de desarrolladores de un proyecto de una línea de micro-juegos de entrenamiento
4. **Contexto del caso:** El proyecto de micro-juegos para una empresa de la ciudad, fue desarrollado por estudiantes de IX semestre de ingeniería informática de la Institución Universitaria Colegio Mayor del Cauca. El propósito de este proyecto, era construir micro-juegos, para las distintas áreas (línea interna fase I, línea interna fase II, línea externa, laboratorio de gas, laboratorio de aguas, bodega y talento humano) de la empresa. La finalidad de cada micro-juego fue enseñar a los empleados de forma dinámica los conceptos y manejo de instrumentos relevantes de cada área. En total se construyeron 12 micro-juegos divididos en cada área y desarrollados por grupos de 2 personas. En la tabla 29, se indica la distribución de los grupos y el área a la cual se desarrolló el micro-juego.

Tabla 29. Distribución de áreas para el desarrollo de micro-juegos.

Grupo	Área
Grupo 1	Línea interna fase I – Dependencia de producción
Grupo 2	Línea interna fase II – Instrumentos
Grupo 3	Línea interna fase II – Instrumentos
Grupo 4	Línea externa – Calibración
Grupo 5	Laboratorio de aguas
Grupo 6	Laboratorio de aguas
Grupo 7	Laboratorio de gas
Grupo 8	Bodega
Grupo 9	Bodega
Grupo 10	Bodega
Grupo 11	Talento humano
Grupo 12	Talento humano

5. **Diseño del estudio**

De acuerdo al objetivo del caso de estudio se diseñó una guía (anexo II: “guías para estudios de caso”, sección A), donde se describió un poco el contexto SPL y se presentó la plantilla de casos de uso de las autoras Erazo y Martins [44], con el fin de evaluar y determinar el nivel de aplicabilidad en la técnica ttSPL de dicha plantilla. Pues, la

especificación de casos de uso que describe la plantilla comprende no sólo la representación de aspectos comunes y variabilidades, sino también la especificación del manejo de excepciones para el comportamiento tolerante a fallos.

De acuerdo al objetivo del caso de estudio, se diseñaron los indicadores, métricas e instrumentos a emplear, la tabla 30 relaciona estos elementos para este estudio de caso.

Tabla 30. Indicadores, métricas y fuentes de información de instrumentos.

Indicadores	Métricas	Instrumentos
Utilidad de un determinado Flujo (UF) UF= $(NT_{F\_X}/ NT\text{-}_{F\_X})*100$	Número Total del Flujo_X diligenciados NT_{F_X} Número Total de Flujo_X No diligenciados $NT\text{-}_{F\_X}$	Plantilla de caso de uso de Erazo y Martins

6. Desarrollo del caso

Para la aplicación del caso de estudio, los estudiantes recibieron indicaciones del contenido de la guía (anexo II: guía estudios de caso, sección A), además cómo se debía desarrollarla, con el objetivo de contextualizar y unificar los conceptos. El estudio se desarrolló en la clase de videojuegos. También, se consideró la presentación final de cada micro-juego en la empresa, con el fin de comparar las plantillas de casos de uso desarrolladas para describir el comportamiento funcional de cada micro-juego con el juego en funcionamiento y de esta manera verificar la información plasmada en cada plantilla de caso de uso.

7. Organización del grupo

Los estudiantes de la electiva de videojuegos en total eran veinticinco (25) personas. Sin embargo, para el desarrollo de un micro-juego la dinámica era de dos o tres personas, es así que para el desarrollo del caso de estudio esta organización se mantuvo. Con la ventaja de que los 24 estudiantes conocían las funcionalidades o dinámica de los 12 micro-juegos desarrollados.

8. Dinámica del proyecto

El desarrollo de los elementos compartidos en la electiva, intuitivamente siguió el enfoque de SPL aplicado al desarrollo de micro-juegos. La línea de productos que se desarrolló fue una línea pequeña, a la que se denominó "*Matrix*". Inicialmente, fue necesario que cada equipo de desarrollo caracterizará los productos que se construirían en base a los requerimientos funcionales del cliente. Durante el desarrollo del proyecto se identificaron los elementos que serían similares y variables en cada juego. Por tanto, se inicia la especificación funcional en la plantilla de caso de uso (ver anexo III: ejecución de estudios de caso, carpeta caso de uso).

9. Resultados obtenidos

a. Resultados cuantitativos

En la tabla 31 se presentan los resultados obtenidos en la evaluación de la plantilla de caso de uso de Erazo y Martins [52], la cual sería utilizada para la técnica propuesta, debido a la similitud del estudio en el contexto de SPL y los resultados de su aplicación revelados en el mismo estudio. También, la tabla 31 permite observar cuales son los que flujos son elementales o más "usados" y cuáles no, para la especificación de un producto a través de

un caso de uso, la información se organiza de acuerdo a la estructura de la plantilla y el número de cuantos campos fueron diligenciados por los participantes, resaltando que para el ejercicio se formaron en total participaron 8 grupos.

Tabla 31. Resultado de la evaluación realizada a la plantilla de caso de uso.

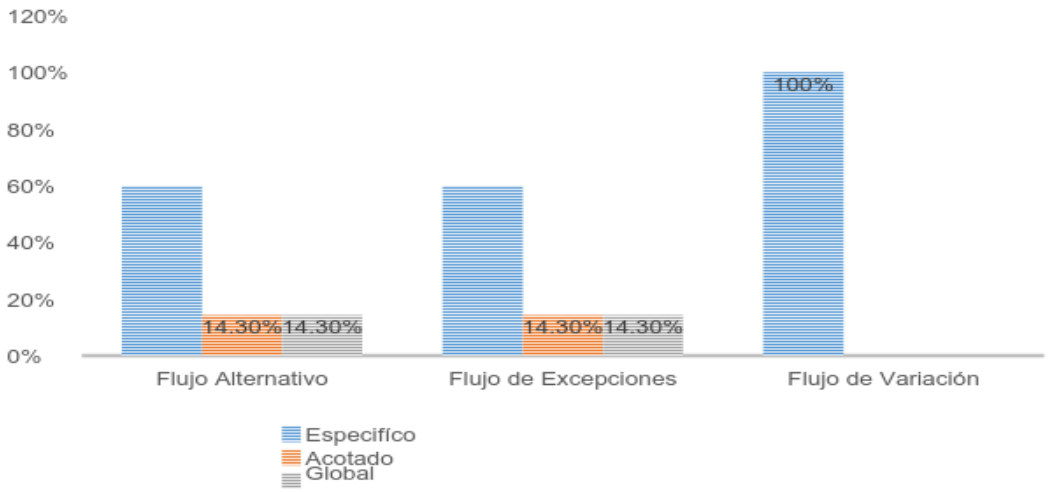
Estereotipos utilizados en la plantilla de casos de uso propuesta por Erazo y Martins [52]	**Número de campos diligenciados**			
ID del caso de uso	8			
Nombre del caso de uso	8			
Tipo del caso de uso	Obligatorio	Opcional	Alternativa	
	7	1	0	
Cardinalidad específica de la característica	[1..1]	[0..1]	[1..n]	
	6	1		
Breve descripción	8			
Característica asociada	6			
Actor primario	8			
Actor secundario	Tiene	No tiene	No digito	
	2	6	0	
Precondición	8			
Dependencia	Tiene	No tiene	No digito	
	2	5	1	
Generalización	Tiene	No tiene	No digito	
	1	7	0	
				Indicados
				Utilidad / **porcentaje**
Flujo de pasos básicos (FPB) Postcondición	8 6			
Flujo alternativo específico (FAE) Postcondición	3 3			UF=0,6 / 60%
Flujo alternativo acotado (FAA) Postcondición	1 0			UF=0,143 / 14,29%
Flujo alternativo global (FAG) Postcondición	**1** **0**			UF=0,143 / 14,29%
Flujo de excepciones específico (FEE) Postcondición	3 1			UF=0,6 / 60%
Flujo de excepciones acotado (FEA) Postcondición	1 0			UF=0,143 / 14,29%
Flujo de excepciones global (FEG) Postcondición	1 0			UF=0,143 / 14,29%
Flujo de variación específico (FVE) Postcondición	4 3			UF=1 / 100%
Flujo de variación acotado (FVA) Postcondición	0 0			UF=0 / 0%
Flujo de variación global (FVG) Postcondición	0 0			UF=0 / 0%

De acuerdo a los resultados que refleja la tabla 31, puede observarse que los flujos diseñados para las variabilidades no fueron llenados por los grupos, puesto que a pesar de la guía proporcionada y de la experiencia en un desarrollo de una SPL, estos no fueron claros. Además, los participantes consideran que no es necesario diligenciar tantas secciones de uno mismo flujo para definir el comportamiento funcional de un producto. Es

así que los flujos con la etiqueta acotado y global no representan un mayor valor en la descripción funcional del producto y añaden confusión en su especificación. De la misma forma la post-condición en cada flujo no tienen un valor agregado más que en el flujo básico de pasos.

En figura 27, se presenta un gráfico de barras para indicar la utilidad de los flujos expuestos en la plantilla de caso de uso evaluada y considerada para la técnica ttSPL. Además, se puede observar que los flujos con la etiqueta “específico” fueron los más utilizados para describir el comportamiento del producto en cada flujo, pues indica un porcentaje de utilidad elevan en comparación con los “acotado” y “global”, pues estos últimos no representan un nivel de utilidad significativo en el comportamiento del producto

Figura 27. Gráfica que representa la utilidad de cada flujo para un caso de uso.



a. Resultados cualitativos

El caso de estudio evidenció que la descripción de escenarios de casos de uso para los productos desarrollados eran escenarios cortos y similares, pues el propósito y la dinámica de cada micro-juego eran análogos, lo que en la descripción implica que varios flujos no aplicaran en la especificación del caso de uso.

Además, se recolectaron algunas sugerencias por parte de los participantes para incluirlas en el nuevo diseño de la plantilla de caso de uso para la técnica propuesta.

- Es excesivo el número de flujos presentados en la plantilla de caso de uso evaluada.
- El contexto de cada grupo (específico, acotado, global) de un determinado flujo tiende a confundir el propósito para cual fue diseñado cada flujo.
- Las siglas que denotan cada flujo tienden a distraer y confundir la descripción del comportamiento del producto.
- No es entendible la definición de los flujos acotado y global.

Figura 28. Participantes del caso de estudio: "Evaluación de la plantilla de caso de uso”

10. Análisis de resultados

A partir de los resultados obtenidos cabe mencionar que es necesario personalizar una plantilla de caso de uso para la técnica ttSPL utilizando el estudio de [44], ya que se evidenció la necesidad de agrupar los flujos (específicos, acotado y global) en uno solo, para evitar confusión en la descripción y lograr un impacto en la especificación de caso de uso para un producto. Algunos estereotipos no se tienen en cuenta en el nuevo diseño de la plantilla de caso de uso para ttSPL, debido a que en los cuestionarios presentaron dificultad en su formalización como es el caso de “cardinalidad específica de la característica” y “generalización”. Se decide eliminar de la nueva plantilla de casos de uso la descripción del flujo de excepciones debido a que representa condición de error y aumentaría significativamente el número de casos de prueba, teniendo en cuenta que el alcance de la técnica no incluye un reporte de pruebas de los productos.

Estudio de caso 2: Caso exploratorio - Línea de Productos Pedagógica “Arcade Game”

Para realizar una descripción de la técnica, acorde a las necesidades propuestas en los objetivos de ttSPL, se hace necesario el estudio de caso de tipo exploratorio que permita obtener un acercamiento tangible al enfoque de la técnica para la derivación de casos de prueba funcionales a partir de casos de uso en el contexto SPL para pequeñas organizaciones.

1. **Pregunta de Investigación**: A partir de la pregunta de investigación del proyecto se evidenció la necesidad de identificar prácticas para la derivación de casos de prueba funcionales adaptables al contexto SPL y prácticas que permiten el manejo de la variabilidad en las SPL, que puedan ser tenidas en cuenta para la especificación de ttSPL. Así que la pregunta que éste estudio de caso busca responder es ¿Qué prácticas de derivación de casos de prueba funcionales y

prácticas del manejo de la variabilidad de SPL se acoplan y pueden ser consideradas para derivar casos de prueba funcionales reutilizables en una pequeña organización?

2. **Objetivo del estudio:** Validar las prácticas seleccionadas que pueden ser tenidas en cuenta en el desarrollo de la técnica ttSPL.
3. **Selección del estudio de caso:** De acuerdo a Runeson y Höst [19], este estudio de caso es exploratorio y holístico con una unidad de análisis, que corresponde a la aplicación de las tareas propuestas en la técnica a una SPL completamente definida.
4. **Contexto del caso:** La línea de productos Software desarrollada por el SEI denominada Arcade Game Maker (AGM) es un ejemplo de línea de productos creada para apoyar el aprendizaje y la experimentación con líneas de productos de software. La línea de productos abarca tres sencillos juegos de arcade. Siendo un ejemplo simple pero comprensivo. Tomando la información que ofrece el SEI se explora la posibilidad de aplicar una serie tareas identificadas en la literatura, para pruebas y SPL, y mezclarlas con el ánimo de verificar su posible uso en ttSPL.
5. **Diseño de estudio de caso:** De acuerdo al objetivo del caso de estudio se estructura la primera versión preliminar (V0.3) de la técnica ttSPL para ejecutar las prácticas que en ese momento conformaban ttSPL (ver Anexo III: Ejecución del estudio de caso par la línea "arcade game"). En la tabla 32, se presentan los indicadores, métricas e instrumentos a emplear en este estudio de caso.

Tabla 32. Indicadores, métricas e instrumentos para el caso exploratorio.

Indicadores	Métricas	Instrumentos
Nivel de Utilidad de los artefactos NU= (AU/(AT-AF))%	AU = Número de artefactos no utilizados como entrada al aplicar el caso. AT = Numero de artefactos totales de la propuesta AF = Numero artefactos finales.	Artefactos de la técnica ttSPL.
Nivel de complejidad de las tareas NC= (CC/CT%)	CT = Cantidad de campos propuestos en las plantillas. CC = Cantidad de campos completos de manera correcta	Plantillas de la técnica ttSPL

6. **Desarrollo del caso**: El caso fue desarrollado por el mismo equipo de trabajo que define la técnica, y fue la aplicación de la primera versión V0.3 de la técnica, donde se desarrollan cada una de las actividades y tareas propuestas en ttSPL. La información de las tareas aplicadas y el desarrollo de estas. (ver anexo III: ejecución de estudios de caso, carpeta arcade game).

7. **Resultados obtenidos**
   a. **resultados cuantitativos**

La tabla 33 relaciona los resultados cuantitativos obtenidos en el estudio de caso exploratorio. Estos permiten ver que las prácticas y artefactos de ttSPL son medianamente complejos de aplicar, pero resultan significativamente útiles en la construcción de caso de pruebas.

Tabla 33. Resultados del nivel de utilidad y complejidad de la técnica ttSPL.

AU	AT	AF	Nivel de Utilidad	CT	CC	Nivel de Complejidad
3	10	8	60%	58	24	41,4%

b. Resultados cualitativos

Se logró apreciar que, si la ingeniería de dominio de una línea de productos contiene los elementos generales que componen la SPL, entonces la ingeniería de aplicación tiene la ventaja de reutilizar todos aquellos elementos y crear nuevos productos de una forma práctica y sencilla con la ventaja de asegurar la calidad de los productos derivados.

El plan de producción de la línea “arcade game” también es representado a través de un modelo de características que contiene todos los elementos identificados para la construcción de un juego pedagógico de la línea y que se diferencia de otro en la definición de reglas específicas pero que no se salen del contexto de leyes de la física, lo anterior permitió entender las características similares y variables de los productos que pertenecen a la SPL.

8. Análisis de resultados

El desarrollo de este caso exploratorio permitió identificar que algunas actividades de la técnica ttSPL debían ser re-diseñadas, otras que no eran necesarias para definir la técnica ttSPL o en algunos casos estas actividades deberían ser de tipo opcional, debido a que no siempre debían ser ejecutadas para construir el caso de pruebas del producto.

También se logró entender que había artefactos que no daban ningún valor a la construcción de casos de pruebas del producto, algunos eran difíciles de diligenciar debido a la terminología o conceptos diferentes entra la técnica ttSPL y la línea “arcade game”.

La documentación que proporciona el SEI para contextualizar al lector sobre la línea “arcade game” permite entender la esencia de la línea y los beneficios que se pueden lograr al definir todos los elementos reutilizables en la ingeniería de dominio y el alcance de la línea y de esta manera lograr conocer los productos que se pueden construir a través de la misma.

Estudio de Caso 3: Aplicabilidad y consistencia de la técnica propuesta en una pequeña organización.

Para que los detalles de la técnica ttSPL, estuvieran dirigidos hacia una técnica aplicable en proyectos reales en el contexto de una pequeña organización de desarrolladora de software, en este estudio se consideraron los factores que pueden refinar y potencializar las actividades que componen la técnica ttSPL, mediante la adopción del enfoque de SPL y la participación con una pequeña organización.

1. **Pregunta de investigación:** a partir de la pregunta de investigación del proyecto era necesario reflexionar sobre la posibilidad real de que una pequeña organización adoptará el enfoque de la técnica ttSPL. La pregunta para este estudio de caso es ¿la técnica ttSPL es una técnica aplicable y consistente en una pequeña organización para

construir casos de pruebas funcionales del producto?

2. **Objetivo del estudio de caso:** el objeto de este estudio fue determinar si la técnica ttSPL es aplicable y consistente dentro de una pequeña organización de software.
3. **Selección del estudio de caso:** de acuerdo a Runeson y Höst [19], este estudio de caso es descriptivo, con una unidad de análisis, la cual corresponde al proceso de desarrollo de la pequeña organización.
4. **Contexto del caso:** el desarrollo de una aplicación móvil multiplataforma es un campo nuevo por explorar dentro de la pequeña organización de software, además de ser una tendencia en el mercado que permite ampliar los horizontes y la visión del software móvil híbrido como herramienta de trabajo. Es así, que esta organización construyó una solución móvil para un producto de software mediante la estrategia de producción SPL para eventos académicos, que tuviera la capacidad de desarrollo en un periodo corto de tiempo, adaptable (acorde a las necesidades), con un núcleo o core como elemento principal de la aplicación y que los requerimientos del cliente se adecuará en el tiempo del desarrollo del proyecto, para ello se desarrolló el proyecto **"**Desarrollo de una Aplicación Móvil Híbrida Multiplataforma Informativa para Congresos Académicos Basados en la Estrategia de Producción, Software Product Line (SPL) **"** como trabajo de grado para optar el título de tecnólogo en desarrollo de software dentro de Institución Universitaria Colegio Mayor del Cauca.
5. **Diseño del caso:** el objeto de este estudio fue determinar la aplicabilidad y consistencia de las actividades de la técnica ttSPL en una pequeña empresa de software que está adoptando el enfoque SPL. La descripción de los indicadores, métricas e instrumentos diseñados para este estudio se relacionan a continuación en la tabla 34.

Tabla 34. Parámetros del caso de estudio aplicabilidad y consistencia de la técnica ttSPL.

Indicadores	Métricas	Instrumentos
Nivel de Aplicabilidad(NA) NA = (NE+NU)/2 Rangos 0<NA≤0.3 es baja Si 0.4≤NA≤0.6 es media Si 0.7 ≤NA≤1 alta	NE: Nivel de Entendimiento de la técnica ttSPL (NF/NTF). NF- Número de Formatos bien diligenciados NTF- Número Total de formatos Nivel de Utilidad (NU) de los artefactos (AU/NAA%) AU - Número de artefactos utilizados en etapas posteriores NAA: Número de Artefactos Aplicados	Encuesta Repositorio del proyecto
Nivel de consistencia (NC) NC = (NACPD/NCPP)% Rangos 0<NC≤39% es baja Si 40%≤CA≤69% es media Si 70% ≤NC≤100% alta	NACPD: Número de Aciertos en la descripción del Caso de Pruebas NCPP: Número de Casos de Pruebas Planteados.	Repositorio del Proyecto

6. Desarrollo del caso

Para el desarrollo de este caso descriptivo se toma la versión (V0.7) de la técnica ttSPL (ver anexo II: guías estudios de caso, sección B) y se aplica a la línea de productos de la

pequeña organización de software, la cual a través de la exploración de aplicaciones informáticas de eventos académicos caracterizaron los productos de la línea. Además, los participantes del proyecto iniciaron con la construcción de un núcleo o core representado en un modelo de características que les facilitó el desarrollo de tres aplicaciones móviles para diferentes eventos académicos utilizando como proceso de desarrollo el *proceso Small SPL*[26]y cada producto fue evaluado con los organizadores de cada evento para comprobar la funcionalidad de cada aplicación como si fueran productos independientes.

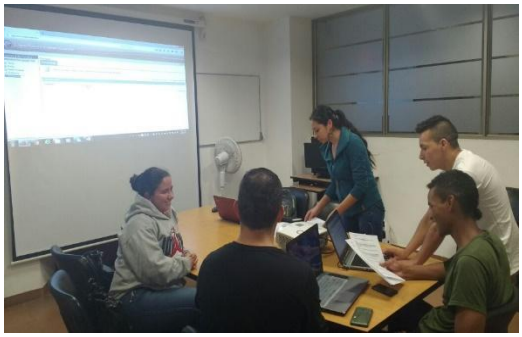
7. Organización del grupo

Para el desarrollo del caso se contó con la participación de la Esp. Ing María Isabel Vidal Caicedo, directora del proyecto: "Desarrollo de una Aplicación Móvil Híbrida Multiplataforma Informativa para Congresos Académicos Basados en la Estrategia de Producción, Software Product Line (SPL)", y los tecnólogos en desarrollo de software Manuel Arturo Melo Legarda y Janier Alejandro Banguera Correa. Todos miembros inscriptos en la Institución Universitaria Colegio Mayor del Cauca. De acuerdo a lo establecido para llevar a cabo este estudio se estructura la segunda versión de técnica ttSPL. Donde a Manuel Arturo Melo Legarda se le asigna el rol de participante 1 y a Janier Alejandro Banguera Correa el de participante 2. El estudio se realizó en dos secciones, una para el desarrollo del procedimiento ttSPL en ingeniería de dominio y la otra para aplicar el procedimiento ttSPL en ingeniería de aplicación.

8. Dinámica del proyecto

A cada participante se le asignó una característica del modelo de características de la línea de productos de eventos académicos para desarrollar los procedimientos ttSPL tanto en la ingeniería de dominio como de aplicación. Durante el desarrollo del proyecto se recolectó la información siguiendo el diseño del estudio en plantillas de Excel donde se representaron los diferentes artefactos propuestos por la técnica (ver anexo III: ejecución de estudios de caso, carpeta aplicabilidad y consistencia de ttSPL).

Figura 29. Participantes del estudio de caso: "aplicabilidad y consistencia de la técnica ttSPL.



[26] Small SPL en las Pymes Desarrolladoras de Software del Cauca: Una Experiencia desde Colmayor [2].

9. Resultados

Para presentar los resultados del estudio de caso se haya útil la siguiente información:

Datos del participante 1:

Nombre Completo:	Manuel Arturo Melo Legarda
Ocupación:	Estudiante de ingeniería informática y desarrollador
Profesión:	Desarrollador de software
Roles desempeñados en La SPL	Analista de dominio. Arquitecto del producto Tester de producto Soporte logístico

Datos del participante 2:

Nombre Completo:	Janier Alejandro Banquera Correa
Ocupación:	Estudiante de ingeniería informática y desarrollador
Profesión:	Desarrollador de software
Roles desempeñados en La SPL	Arquitecto de dominio. Desarrolladores de componentes Ingeniero de producto Soporte logístico

c. Resultados cuantitativos

La tabla 35 relaciona los resultados cuantitativos obtenidos en el estudio de caso descriptivo. Estos permiten ver que los artefactos de los procedimientos ttSPL en ingeniería de dominio y aplicación son medianamente complejos de diligenciar, pero resultan significativamente entendibles y útiles. Aunque los artefactos para la ingeniería de dominio de ttSPL resultaron más complejos de adaptar y finalmente la aplicabilidad de los artefactos en la construcción de los productos resultó alta. Por otro lado, los artefactos de la técnica ttSPL son medianamente consistente, especialmente en la ingeniería de aplicación, puesto que existen artefactos no conservaban la estructura de la ingeniería de dominio.

Tabla 35. Resultados de los parámetros de evaluación del caso.

Procedimiento ttSPL en	NE	NU	Nivel de Aplicabilidad	Nivel de consistencia
Participante 1				
Ingeniería de dominio	6/7	4/7	0.7	
Ingeniería de aplicación	3/5	4/5	0.7	0.6= 60%
Participante 2				
Ingeniería de dominio	4/7	4/7	0.5	
Ingeniería de aplicación	4/5	4/5	0.8	0.7=70%

d. Resultados cualitativos

Durante el desarrollo del estudio de caso se toma el tiempo utilizado para la realización del procedimiento ttSPL en ingeniería de dominio y el procedimiento ttSPL en ingeniería de aplicación, donde se logró evidenciar que hay una disminución considerable en el tiempo

para realizar el caso de pruebas del producto.

Tiempo empleado en el estudio de caso:

Procedimiento	Hora inicio	Hora fin	Total de horas utilizadas
Ingeniería de dominio	6:40 p.m.	9:40 p.m.	3 horas
Ingeniería de aplicación	6:30 p.m.	7:43 p.m.	1 hora y 13 minutos

Seguidamente, se detallan las preguntas que contiene la encuesta planteada con la calificación otorgada por cada participante, la calificación está relacionada con la escala de Likert (ver tabla 36).

Tabla 36. Escala Likert.

Escala	Detalle
1	Totalmente en desacuerdo
2	En desacuerdo
3	Ni en desacuerdo
4	De acuerdo
5	Totalmente de acuerdo

En la tabla 37 se presentan algunas de las preguntas planteadas en la encuesta diseñada para este estudio de caso que tiene como objetivo validar la aplicabilidad y consistencia de la técnica ttSPL, las preguntas fueron calificadas según la escala indica en la tabla 36.

Tabla 37. Preguntas y calificación de la encuesta para validar la aplicabilidad y consistencia de la técnica ttSPL.

	Calificación asignada	
Pregunta	**Participante 1**	**Participante 2**
¿Considera usted que la técnica propuesta es fácil de ser aplicada en una pequeña organización?	4	4
¿De acuerdo a su experiencia, la técnica propuesta es fácil para ser aplicada?	4	4
¿Considera usted que la técnica propuesta es útil para una pequeña organización?	5	4
¿Considera usted que la técnica propuesta tiene una estructura adecuada?	4	4
¿Considera usted que las tareas propuestas en cada actividad de la técnica propuesta son apropiadas para ser utilizadas por una pequeña organización?	5	5
¿De acuerdo a su experiencia, las tareas propuestas en cada actividad de la técnica propuesta son apropiadas?	5	5
¿Considera usted que los artefactos formalizados son adecuados para cada actividad de la técnica propuesta?	5	4
¿Considera usted que los diagramas presentados describen de forma clara la técnica propuesta y las actividades propuestas para la técnica presentada?	4	5
¿Considera usted que la definición para cada actividad y tarea de la técnica propuesta es entendible?	4	4

	Calificación asignada	
Pregunta	**Participante 1**	**Participante 2**
¿Considera usted que la aplicación de la técnica propuesta requiere de un tiempo y esfuerzo considerable?	5	3
¿Considera usted que algún artefacto formalizado para la técnica requiere un tiempo y esfuerzo considerable para ser diligenciar?	4	3
¿Considera usted que la estrategia de reutilización de los elementos que define la técnica propuesta es evidente y aplicable?	5	4

También, en la encuesta se formularon preguntas para responder abiertamente o según la perspectiva sobre los procedimientos ttSPL utilizados en la ingeniería de dominio y aplicación para este estudio.

1. ¿Considera usted que los artefactos formalizados para la técnica propuesta son apropiados para ser utilizados en una pequeña organización?

Respuesta del participante 1	Respuesta del participante 2
Si, son apropiados, entendibles y acorde a la temática	Son aplicables siempre y cuando se automatice un poco más algunos artefactos en la parte de ingeniería de dominio, lo que facilitaría la aplicación de la técnica propuesta.

2. ¿De acuerdo a su experiencia, los artefactos formalizados para la técnica propuesta son apropiados?

Respuesta del participante 1	Respuesta del participante 2
En el ámbito de desarrollo si son apropiados y facilita el trabajo en SPL	Son apropiados para personas que tengan conocimientos en el tema por tanto son fáciles de entender. Pero se podrían mejorar para hacerlos aún más entendibles para personas que aun tenga la iniciativa de adoptar SPL.

3. ¿Considera usted que los roles propuestos en la técnica propuesta pueden ser asumidos en una pequeña organización?

Respuesta del participante 1	Respuesta del participante 2
Sí, no generan mayor complejidad	Totalmente de acuerdo

4. ¿Considera usted que la técnica propuesta le hacen falta algunos elementos?

Respuesta del participante 1	Respuesta del participante 2
A fondo abarca toda la temática a tratar y de forma se mejora usando elementos gráficos y ejemplos de modo de comparación	

5. ¿Considera usted que la técnica propuesta permite obtener casos de pruebas derivados?

Respuesta del participante 1	Respuesta del participante 2
En nuestro ejercicio si aplica para generar nuevos productos y se obtienen características derivadas	Sí, porque lo facilita como iteraciones.

10. Análisis de resultados

De acuerdo a los resultados obtenidos la técnica ttSPL es aplicable y consistente en la línea de productos de la pequeña organización, pues se adaptó en el proyecto a la línea de aplicaciones móviles para congresos académicos. La versión evaluada de la técnica ttSPL permitió la construcción de los artefactos del dominio como: la especificación de casos de uso, especificación de pruebas, la definición escenarios de pruebas y los casos de pruebas. Así mismo, permitió aplicar las actividades o tareas definidas en la ingeniería de aplicación, dando la posibilidad de derivar los casos de pruebas y personalizar una nueva funcionalidad para las características seleccionada para un producto.

El proceso de aplicar las tareas definidas para ttSPL, evidenció la necesidad de reorganizar y estructurar algunas tareas de tal forma que fueran más detalladas y descritas de forma más sencilla para el participante, especialmente la parte de ingeniería de aplicación, puesto que no está dividida en tareas que indique los pasos para de cada una de ellas

Un hallazgo importante que se encontró al momento de desarrollar las actividades para la ingeniería de aplicación, fue que al momento de agregar una nueva característica para un producto no tiene que ir directamente al diseño de pruebas, se debe diseñar todos los elementos del dominio (ingeniería de requisitos y diseños de pruebas), con el fin de garantizar la trazabilidad entre los requisitos y los casos de pruebas, ya que únicamente se tenía en cuenta el diseño de pruebas para la construcción de un producto a partir de la configuración del modelo de características y no se estaba especificando los casos de uso asociados.

Revisión con un experto en pruebas de software: Evaluación de los artefactos que pertenecen a la técnica ttSPL

Para que de la técnica ttSPL tuviera un valor agregado en cuanto a la aplicabilidad y consistencia en los artefactos desarrollados tanto para el procedimiento ttSPL en ingeniería de dominio y el procedimiento ttSPL en ingeniería de aplicación, se consulta un agente experto en pruebas de software de una pequeña organización de software dedicada al diseño, implementación y ejecución de pruebas de software para la industria.

1. **Pregunta de investigación:** a partir de la pregunta de investigación del proyecto era necesario reflexionar sobre la posibilidad real de que una pequeña organización adoptara el enfoque de la técnica ttSPL. La pregunta para este estudio de caso es ¿Los artefactos de la técnica ttSPL son aplicable y consistentes en una pequeña organización para construir casos de pruebas funcionales del producto?
2. **Objetivo del estudio de caso:** El objeto de este estudio fue observar y contrastar la

práctica propuesta con las prácticas que usualmente realiza un ingeniero de pruebas.

3. **Selección del estudio de caso:** de acuerdo a Runeson y Höst [19], este estudio de caso es descriptivo.
4. **Contexto del caso:** los casos de pruebas representan la esencia de la técnica ttSPL, por tanto, se haya necesario los conocimientos de una experta en pruebas para validar el proceso cómo se construyen los casos de pruebas y el propósito para derivar los productos que pertenecen a la SPL, a partir de los casos de pruebas y los casos de uso.
5. **Diseño del caso:** el objeto de este estudio fue determinar la aplicabilidad y consistencia de los artefactos desarrollados para la técnica ttSPL, lo que brinda la oportunidad de contrastar el proceso de pruebas que utiliza la empresa de software donde labora la experta en pruebas o ingeniera de pruebas, con el procedimiento que desarrolla la técnica ttSPL utilizando el enfoque de línea de productos.
6. **Desarrollo del caso:** Para el desarrollo de este estudio se desarrolla una guía (ver anexo II: guías estudios de caso, sección C) donde se haya necesario contextualizar a la experta en pruebas sobre el enfoque SPL y la técnica ttSPL. En la medida que se expone la técnica ttSPL se van presentan los artefactos desarrollados tanto en la ingeniería de dominio como de aplicación haciendo énfasis en el procedimiento diseñado para los caso de pruebas del dominio y los casos de pruebas para el producto, además se decide ir comparando el proceso que siguen la experta para desarrollar las pruebas en la industria de software con lo que desarrolla ttSPL y así contrastar el proceso reales de pruebas que sigue una organización con lo expuesto por ttSPL.

Figura 30. Evaluación de la técnica ttSPL con la experta en pruebas.



7. Resultados

Para validar la percepción acerca del procedimiento que plantea la técnica ttSPL para la construcción de los casos de pruebas frente al conocimiento y experiencia de la ingeniera de pruebas se desarrolla una encuesta que se responde según la escala de Likert (ver tabla 36).

Datos de la experta en pruebas

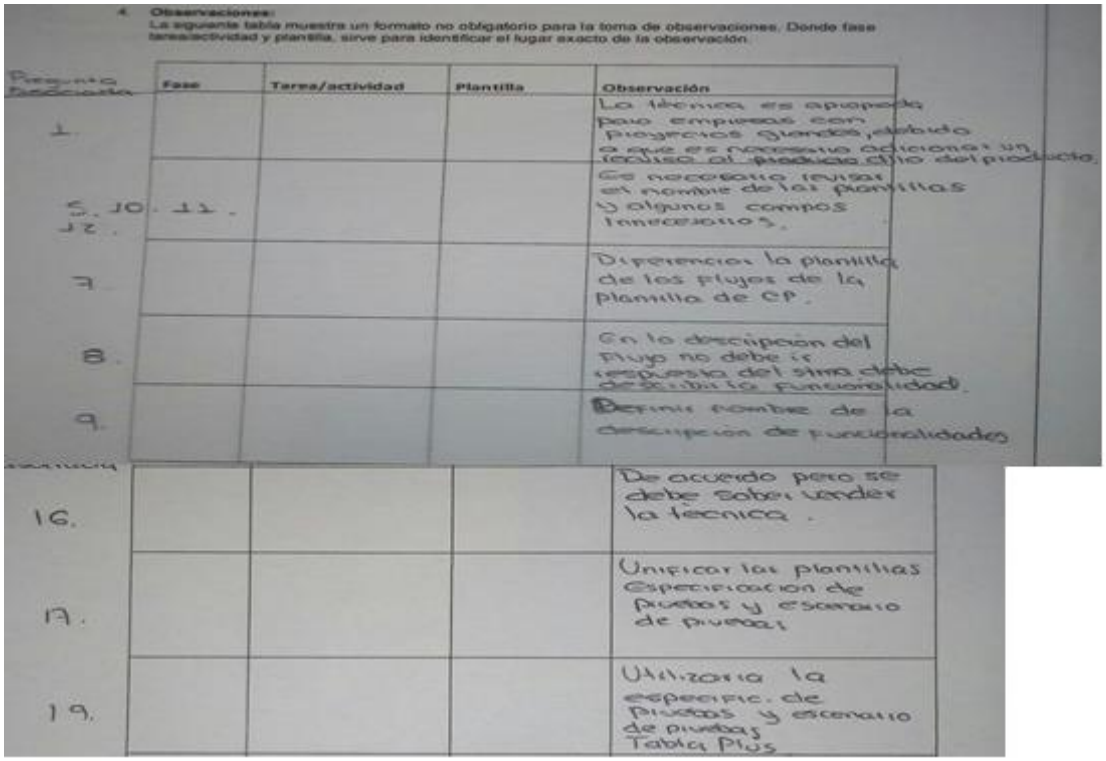
Nombre Completo:	Adriana Vásquez
Ocupación:	Ingeniería de pruebas
Profesión:	Ingeniera informática
Años de experiencia	6 años de experiencia

Resultado de la encuesta.

Nro.	Pregunta	1	2	3	4	5
1	¿La técnica propuesta es fácil de ser aplicada en una pequeña organización?		X			
2	¿La técnica cuenta con un proceso de pruebas bien definido?					X
3	¿Los artefactos que hacen parte de ttSPL son considerados los necesarios para el desarrollo de la técnica considerando las pruebas?				X	
4	¿Las plantillas tienen una estructura definida y consistente?				X	
5	¿La definición de los campos y secciones de las plantillas es perfectamente claro?			X		
6	¿Los campos de las plantillas pueden ser llenados a partir de los artefactos de entrada para la tarea?					X
7	¿Existe ambigüedad en los artefactos de la técnica?			X		
8	¿Existe ambigüedad en los campos delas plantillas de la técnica?			X		
9	¿Los nombres utilizados para los diferentes artefactos o productos de trabajo tienen relación con su contenido?			X		
10	¿Los nombres utilizados para los diferentes campos de las plantillas tienen relación con su contenido?			X		
11	¿Las etiquetas usadas son útiles?			X		
12	¿La trazabilidad de las etiquetas puede observarse en los diferentes artefactos?			X		
13	¿Las definiciones de la tarea tienen estrecha relación con los productos generados?					X
14	¿Se puede crear una idea del contenido del producto de trabajo a partir de la lectura del objetivo de la tarea?					
15	¿Las tareas propuestas son el número adecuado teniendo en cuenta el enfoque de pequeñas organizaciones?	X				
16	¿La fase de ingeniería de dominio en el diseño de las pruebas está completa?					X
17	¿Los artefactos de la ingeniería de dominio en el diseño de pruebas son estrictamente necesarios para el buen desarrollo de la técnica?			X		
18	¿La fase de ingeniería de dominio en el diseño de las pruebas es comprensible?					X
19	¿La fase de ingeniería de dominio en el diseño de las pruebas utiliza todos sus artefactos?			X		
20	¿La fase de ingeniería de aplicación en de las pruebas está completa?					X
21	¿Las tareas de la ingeniería de aplicación son estrictamente necesarios para el buen desarrollo de la técnica?					X
22	¿La fase de ingeniería de aplicación es comprensible?					X

Sugerencias realizadas por la experta en pruebas para ttSPL:

Figura 31. Sugerencias de la experta en pruebas.



4. Observaciones:
La siguiente tabla muestra un formato no obligatorio para la toma de observaciones. Donde fase tarea/actividad y plantilla, sirve para identificar el lugar exacto de la observación

Pregunta	Fase	Tarea/actividad	Plantilla	Observación
1.				La técnica es apropiada para empresas con proyectos grandes, debido a que es necesario adicionar un [illegible]
5. 10. 11. 12.				Es necesario revisar el nombre de las plantillas y algunos campos innecesarios.
7.				Diferencias la plantilla de los flujos de la plantilla de CP.
8.				En la descripción del flujo no debe ir respuesta del sitma debe describir la funcionalidad.
9.				Definir nombre de la descripción de funcionalidades
16.				De acuerdo pero se debe saber vender la técnica.
17.				Unificar las plantillas Especificación de pruebas y escenario de pruebas
19.				Utilizaría la especific. de pruebas y escenario de pruebas Tabla Plus

8. Análisis de los resultados

La evaluación con la experta en pruebas fue positiva debido a que permitió identificar que algunas prácticas para la construcción de pruebas de software que utiliza en su ámbito laboral corresponden a ciertas tareas de la técnica ttSPL o beneficios que brinda el enfoque SPL.

También, con la evaluación se logró renombrar algunos de los roles utilizados en ese momento por la técnica ttSPL, de tal forma que quedaran más acorde a los de la industria en un entorno de pruebas real y utilizando el enfoque SPL. Se renombrar el rol “líder de pruebas” que anteriormente era el encargado de diseñar las pruebas, por el rol “ingeniero de pruebas”.

Además, por sugerencia de la experta se integran la estructura de los documentos “especificación de pruebas” y “escenarios de pruebas” en uno, pues la función de estos tiene el mismo propósito por tanto permite dicha unión y representación

CAPÍTULO V

CONCLUSIONES, LIMITACIONES Y TRABAJOS FUTUROS

CONCLUSIONES

Con la consideración de la uniformidad y variabilidad en el diseño de las pruebas de una línea de productos software se disminuye el esfuerzo requerido y se aumenta el efecto alcanzado lo que mejora la calidad en los productos de software y potencializa las oportunidades para que la empresa productora sea más competitividad en el mercado. Pues, al facilitar una técnica que permita a la empresa continuar aplicando la ideología del enfoque de desarrollo de líneas de productos en la realización de las pruebas, la identificación de los elementos comunes hace que los ingenieros de pruebas se esmeren en el cumplimiento de las funcionalidades comunes, e identificar los puntos variantes permiten que en implementación de las pruebas se concentre en los elementos no verificados hasta el momento, esto mejora el uso de los recursos y la calidad de los productos.

La técnica propuesta en este proyecto facilita la derivación y el refinamiento de los casos de prueba de los productos a partir de los casos de uso de la ingeniería de dominio, en el desarrollo de una línea de productos. Esta técnica busca que pequeñas empresas puedan empalmarla a su proceso de desarrollo de SPL sin elevar significativamente el esfuerzo requerido para incrustarla y disminuyendo el esfuerzo aplicado en la elaboración de pruebas de los productos e incrementando los resultados obtenidos. ttSPL es una primera aproximación en la cual se han considerado actividades que consideran los ajustes entre artefactos diferentes, como también aspectos concretos de las SPL, la propuesta fue evaluada en diferentes contextos, donde se validó su aplicabilidad y consistencia.

La revisión de la literatura como los hallazgos en los diferentes estudios de caso evidenció que las pequeñas organizaciones de software utilizan prácticas similares a algunas de SPL, estas empresas tienden a desarrollar sus productos similarmente o utilizando una plataforma común que le permite el desarrollo de sus productos de software, en un tiempo corto, o buscan producir elementos reutilizables en sus diferentes productos, la técnica por lo tanto puede ser utilizada por empresas que usen elementos de SPL aunque no exista una adopción total del enfoque

Para construir la técnica ttSPL fue necesario tener claros los conceptos que componen las líneas de productos, ya que uno de los inconvenientes encontrados en la exploración de la estrategia SPL fue la falta de normalización en la terminología, ya que un mismo concepto es representado de diferentes formas lo provoca confusión y ambigüedad en la desarrollo y utilización de líneas de productos.

LIMITACIONES

Para poder aplicar la técnica se debe realizar ciertos artefactos que serán insumos para la aplicación de la misma, esto puede indicar más trabajo para el equipo de desarrollo si su proceso no considero estos artefactos y tocaría contrastar entre esfuerzo y los beneficios para determinar si vale o no la pena aplicar la técnica

Identificar los flujos y las variantes no es una tarea fácil.

TRABAJOS FUTUROS

- Incluir una plantilla que facilite la gestión de la reutilización entre la misma y otras líneas de productos, esto facilitaría que los encargados de pruebas verifiquen que casos de pruebas ya tienen diseñados que puedan servir para otros productos de otra línea.
- La automatización de la técnica ttSPL.
- Evaluar la técnica ttSPL en la industria de software con expertos en pruebas de software.

GLOSARIO DE TÉRMINOS

Para nuestra propuesta de investigación se emplearán los siguientes conceptos:

- Proceso de software: “es un conjunto coherente de políticas, estructuras organizacionales, tecnologías, procedimientos y artefactos que son necesarios para concebir, desarrollar, instalar y mantener un producto software” [75][76][77].
- Industria de Software: es la producción que involucra la investigación, el desarrollo, la distribución y la comercialización de software [78].
- Producción sistemática: es la elaboración ordenada de productos, que se ajusta a un sistema [79].
- Activos de software son: una colección de partes de software (requisitos, diseños, componentes, casos de prueba, etc.) que se configuran y se componen de manera prescrita para producir los productos de la línea [80].
- Arquitectura de software: abarca el conjunto de decisiones importantes acerca de la organización de un sistema de software, que incluye la selección de los elementos estructurales y sus interfaces mediante el cual el sistema se compone [81].
- Líneas de Productos Software (Software Product Lines - SPL): “*se definen las líneas del producto de software como un conjunto de sistemas software, que comparten un conjunto común de características (features), las cuales satisfacen las necesidades específicas de un dominio o segmento particular de mercado, y que se desarrollan a partir de un sistema común de activos base (core assets) de una manera preestablecida*” [58].
- Las organizaciones de software, se enfrentan al reto de ser competitivas para lograr su supervivencia en la industria del software. Además, apuntan hacia la mejora de procesos de software y a la innovación en sus técnicas, prácticas y herramientas [2].
- Refinamiento, “operación frecuente en la generación de modelos que permite detallar una entidad a partir de su descomposición en una estructura de nuevas entidades” [82]. El refinamiento en ttSPL permiten detallar casos de pruebas del producto en el modelo de casos de pruebas de dicho producto.
- Derivación, “ proceso completo de construcción de productos de la línea de productos” [83]. La derivación en ttSPL permite construir casos de pruebas del producto de una SPL a partir de los casos de pruebas del dominio y los casos de uso.
- Una técnica consiste en “un sistema de reglas orientadas a conseguir un resultado útil”, teniendo en cuenta la habilidad para aplicar dichas reglas [73]. Una técnica en ttSPL, comprende el conjunto de pasos o acciones utilizadas para realizar una determinada tarea de ttSPL, brindando la información necesaria para establecer los pasos específicos y bajo qué condiciones se realizar una determinada tarea de ttSPL.
- Pruebas funcionales: tienen como objetivo validar cuando el comportamiento observado del software probado cumple o no con sus especificaciones [84], a través de la especificación de casos de prueba [85].

REFERENCIAS

[1] F. García, J. Marqués, and J. Maudes, “Mecano: Una propuesta de componente software reutilizable,” *Las II Jornadas Ing. Del Software*, pp. 232–244, 1997.

[2] M. C. C. Ojeda, “SPL en las PYMES Desarrolladoras de Software del Cauca: Una Experiencia Desde Colmayor,” p. 130, 2013.

[3] T. E. Colanzi, W. K. G. Assunção, D. de Freitas Guilhermino Trindade, C. A. Zorzo, and S. R. Vergilio, “Evaluating Different Strategies for Testing Software Product Lines,” *J. Electron. Test.*, vol. 29, no. 1, pp. 9–24, 2013.

[4] B. P. . Lamancha, M. P. . Usaola, and M. P. . Velthius, “Software product line testing - A systematic review,” *ICSOFT 2009 - 4th Int. Conf. Softw. Data Technol. Proc.*, vol. 1, pp. 23–30, 2009.

[5] C. Y. Laporte, R. V O’Connor, and L. H. G. Paucar, “Software Engineering Standards and Guides for Very Small Entities: Implementation in two start-ups,” *10th Int. Conf. Eval. Nov. Approaches to Softw. Eng. (ENASE 2015)*, no. May, 2015.

[6] Y.-G. Kim, S. K. Lee, and S.-B. Jang, “Variability Management for Software Product-Line Architecture Development,” *Int. J. Softw. Eng. Knowl. Eng.*, vol. 21, no. 7, pp. 931–956, 2011.

[7] A. Bertolino, A. Fantechi, S. Gnesi, and G. Lami, “11 Product Line Use Cases : Scenario-Based Specification and Testing of Requirements,” *Chapter 11 Softw. Prod. Lines Res. Issues Eng. Manag. T. Käkölä J. C. Duenas, Springer*, 2006.

[8] C. Denger and R. Kolb, “Testing and inspecting reusable product line components,” *Proc. 2006 ACM/IEEE Int. Symp. Int. Symp. Empir. Softw. Eng. - ISESE ’06*, vol. 2006, p. 184, 2006.

[9] R. Rabiser, P. Leary, and I. Richardson, “Key activities for product derivation in software product lines,” *J. Syst. Softw.*, vol. 84, no. 2, pp. 285–300, 2011.

[10] F. Pino, F. Garcia, and M. Piattini, “Proceso de Valoración para la Mejora de Procesos Software en Pequeñas Organizaciones,” *XI Work. Ing. Requisitos y Ambient. Software, IDEAS 2008*, pp. 211–224, 2008.

[11] V. Pero and A. R. Reserved, “El modelo,” 2010.

[12] M. Lucía, R. F. José, P. J. Mauricio, M. L. Rojas-montes, F. J. Pino-correa, and J. M. Martínez, “Proceso de pruebas para pequeñas organizaciones desarrolladoras de software Testing process for small software development organizations Processo de provas para desenvolvedoras de software pequenas organizações,” vol. 24, no. 39, pp. 55–70, 2015.

[13] A. Bertolino and S. Gnesi, “Pluto: A test methodology for product families,” *Softw. Prod. Eng.*, vol. 3014/2004, pp. 181–197, 2004.

[14] T. Von Der Maßen and H. Lichter, “Modeling Variability by UML Use Case Diagrams RWTH Aachen 1 Introduction 2 Modeling Functional Variability,” no. September, pp. 19–25, 2002.

[15] C. Y. Laporte, “Software Engineering Standards and Guides for Very Small Entities : Implementation in two start-ups,” no. Enase, 2015.

[16] C. Y. Laporte, R. V. O. Connor, R. Houde, and J. Marvin, “ISO / IEC 29110 : Systems

Engineering Standards for Very Small Enterprises," no. February, 2015.
[17] M. Fazal-baqaie, M. Luckey, and G. Engels, "Assembly-based Method Engineering with Method Patterns," pp. 435–444, 2013.
[18] M. Bunge, "Mario Bunge," Sci., vol 28, no1, p 72, 1961.
[19] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," pp. 131–164, 2009.
[20] B. González-Baixauli, M. A. Laguna, and J. C. S. do P. Leite, "Análisis de Variabilidad con Modelos de Objetivos.," *Wer*, pp. 77–87, 2004.
[21] K. Garcés, C. Parra, and H. Arboleda, "Administración de Variabilidad en una línea de producto basada en modelos," *Proc.*, 2007.
[22] A. Van Lamsweerde, "Requirements Engineering in the Year 00: A Research Perspective," *ICSE '00 Proc. 22nd Int. Conf. Softw. Eng.*, pp. 5–19, 2000.
[23] I. F. Da Silva, P. A. Da Mota Silveira Neto, P. O'Leary, E. S. De Almeida, and S. R. D. L. Meira, "Software product line scoping and requirements engineering in a small and medium-sized enterprise: An industrial case study," *J. Syst. Softw.*, vol. 88, no. 1, pp. 189–206, 2014.
[24] M. F. Johansen, "Testing Product Lines of Industrial Size: Advancements in Combinatorial Interaction Testing," 2013.
[25] I. S. Santos, R. M. Andrade, and P. a Santos Neto, "Templates for textual use cases of software product lines: results from a systematic mapping study and a controlled experiment," *J. Softw. Eng. Res. Dev.*, vol. 3, no. 1, p. 5, 2015.
[26] P. Clements and L. Northrop, *Software Product Lines: Practices and Patterns*. Addison-Wesley, 2002.
[27] G. Heller, "Report of the GI Work Group " Requirements Engineering for Product Lines "," no. 121, 2003.
[28] A. Dur, D. Benavides, and J. Bermejo, "to Requirements Engineering Process Definition," vol. 1, pp. 140–151, 2004.
[29] K. Pohl, G. Böckle, and F. Van Der Linden, *Software Product Line Engineering. Foundations, Principles, and Techniques*, vol. 49, no. 12. 2005.
[30] J. D. Mcgregor, "Testing a Software Product Line," no. December, 2001.
[31] M. a Laguna, "Desarrollo de Líneas de Productos : un Caso de Estudio en Comercio Electrónico," pp. 1–11, 2009.
[32] J. Kim, M. Kim, and S. Park, "Goal and Scenario Based Domain Requirements Analysis Environment," *J. Syst. Softw.*, vol. 79, no. 7, pp. 926–938, 2006.
[33] H. Gomaa, *Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 2004.
[34] M. Eriksson, *An Approach to Software Product Line Use Case Modeling*. 2006.
[35] J. D. McGregor, "Testing a Software Product Line," *Test. Tech. Softw. Eng. Second Pernambuco Summer Sch. Softw. Eng. PSSE 2007, Recife, Brazil, December 3-7, Revis. Lect.*, vol. LNCS 6153, no. December, pp. 104–140, 2007.
[36] F. C. Roos, "Análisis automático de líneas de producto software usando distintos modelos de variabilidad," pp. 1–96, 2008.

[37] J. Van Gurp, J. Bosch, and M. Svahnberg, “Managing Variability in Software Product Lines,” pp. 1–13, 2010.

[38] M. Harsu and P. O. Box, “A Survey of Product-Line Architectures,” 2001.

[39] P. van den Broek, “Extended Feature Models.” [Online]. Available: https://www.utwente.nl/ewi/trese/b_referaat/broek1/.

[40] K. C. Kang and P. Donohoe, “Feature-oriented product line engineering,” *IEEE Softw.*, vol. 19, no. 4, pp. 58–65, 2002.

[41] W. Van Der Vegt, W. Westera, E. Nyamsuren, A. Georgiev, and I. M. Ortiz, “RAGE Architecture for Reusable Serious Gaming Technology Components,” *Int. J. Comput. Games Technol.*, vol. 2016, 2016.

[42] B. M. Eriksson and J. and K. B. Börstler, “SOFTWARE PRODUCT LINE MODELING MADE PRACTICAL An example from the Swedish Defense Industry.,” *Commun. ACM*, vol. 49, no. 12, pp. 49–54, 2006.

[43] J. R. Salazar, “Herramienta para el modelado y configuración de modelos de características,” p. 101, 2009.

[44] L. Erazo and E. Martins, “Modeling Dependable Product-Families : from Use Cases to State Machine Models,” pp. 3–6, 2016.

[45] A. Fantechi, S. Gnesi, I. John, G. Lami, and J. Dörr, “Elicitation of Use Cases for Product Lines,” Softw. Prod. Fam. Enginnering, 5th Int. Work. PFE, Siena, Italy, pp.152–162, 2003.

[46] U. S. A. John D. McGregor, Clemson University and Luminary Software LLC, “Product Production Structure,” *J. OBJECT Technol.*, vol. 3, no. 10, pp. 89–98, 2004.

[47] G. Guedes, C. Silva, and J. Castro, “Goals and scenarios to software product lines: The GS2SPL approach,” *CEUR Workshop Proc.*, vol. 1005, no. i, 2013.

[48] G. J. Myers, T. M. Thomas, and J. Wiley, *The Art of Software Testing , Second Edition*. 2004.

[49] I. C. Society, *Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)*, 2004.

[50] S. Fujiwara, G. v. Bochmann, F. Khendek, M. Amalou, and a. Ghedamsi, “Test selection based on finite state models,” *IEEE Trans. Softw. Eng.*, vol. 17, no. 6, pp. 591–603, 1991.

[51] A. Henry, T. Zenteno, and N. I. E. X-t, “Metodo de pruebas de sistema basado en modelos navegacionales en un contexto MDWE,” 2008.

[52] J. M. Torio, “Escuela Técnica Superior de Ingenieros de Telecomunicación Departamento de Ingeniería de Sistemas Telemáticos Tesis Doctoral,” 2004.

[53] C. Pardo, J. Ariel, H. Alegria, F. J. Pino, and C. No, “Factores de Éxito o Fracaso para la Mejora de Procesos Software : Caso Real en un Grupo de MiPyMEs,” PyMe, Rev. Gerenc. Tecnológica GTI, pp. 21–29, 2006.

[54] C. Y. Laporte, É. De, R. V. O. Connor, and G. Fanmuy, “International Systems and Software Engineering Standards for Very Small Entities,” no. June, pp. 28–33, 2013.

[55] Q. Boucher, G. Perrouin, J. Deprez, and P. Heymans, “Towards Configurable ISO / IEC 29110-compliant Software Development Processes for Very Small Entities,” Proc. 19th Eur. Conf. "System, Softw. Serv. Process Improv. (EuroSPI 2012),

D. Winkler, R. V. O'Connor, R. Messnarz, Eds. Commun. Comput. Inf. Sci. Ser., vol. 301, no. Springer, pp. 169–180, 2012.

[56] F. J. Pino, O. Pedreira, F. García, M. R. Luaces, and M. Piattini, "Using Scrum to guide the execution of software process improvement in small organizations," *J. Syst. Softw.*, vol. 83, no. 10, pp. 1662–1677, 2010.

[57] A. Bertolino and S. Gnesi, "Use Case-based Testing of Product Lines," in *Proceedings of the 9th European Software Engineering Conference Held Jointly with 11th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pp. 355–358, 2003.

[58] Ó. Díaz and S. Trujillo, "Líneas de producto software," *… Softw. Exp. Tecnol. y …*, pp. 1–14, 2007.

[59] M. Lochau, S. Oster, U. Goltz, and A. Schu, "Model-based pairwise testing for feature interaction coverage in software product line engineering," pp. 567–604, 2012.

[60] C. R. L. Neto, I. D. C. Machado, I. Do Carmo MacHado, P. a. M. S. Neto, E. S. De Almeida, and S. R. De Lemos Meira, "Software Product Lines System Test Case Tool: A Proposal.," *SEKE 2011 - Proc. 23rd Int. Conf. Softw. Eng. Knowl. Eng.*, pp. 699–704, 2011.

[61] S. Mishra, "Specification based software product line testing: a case study," *Proc. Concurr. Specif. Program. Work.*, pp. 243–254, 2006.

[62] S. Gruner, "Software testing in small IT companies: a (not only) South African problem," *South African Comput. J.*, no. 47, pp. 7–32, 2011.

[63] E. Kamsties, K. Pohl, S. Reis, and A. Reuys, "Testing Variabilities in Use Case Models," *Lect. Notes Comput. Sci.*, vol. 3014, no. November 2003, pp. 6–18, 2004.

[64] R. Rabiser, P. Gr??nbacher, and D. Dhungana, "Requirements for product derivation support: Results from a systematic literature review and an expert survey," *Inf. Softw. Technol.*, vol. 52, no. 3, pp. 324–346, 2010.

[65] A. Reuys, E. Kamsties, K. Pohl, and S. Reis, "Model-based system testing of software product families," *Lect. Notes Comput. Sci.*, vol. 3520, pp. 519–534, 2005.

[66] D. Wood and J. Reis, "Use case derived test cases," *STAREAST Softw. Qual. Eng. Conf.*, 1999.

[67] W. Are, S. Software, and O. Different, "Why Are Small Software Organizations Different?," pp. 18–22, 2007.

[68] F. van der Linden, J. Bosch, E. Kamsteries, K. Känsälä, and H. Obbink, "Software product family evaluation," *Third Int. Conf. SPLC*, pp. 110–129, 2004.

[69] C. Gacek, P. Knauber, K. Schmid, and P. C. Clements, "Successful Software Product Line Development in a Small Organization A Case Study," no. 13, 2001.

[70] D. Benavides and J. a J. a Galindo, "Variability management in an unaware software product line company. An experience report," *ACM Int. Conf. Proceeding Ser.*, p. 5:1--5:6, 2014.

[71] N. Nazar and T. M. J. Rakotomahefa, "Analysis of a Small Company for Software Product Line Adoption — An Industrial Case Study," *Int. J. Comput. Theory Eng.*, vol. 8, no. 4, pp. 313–322, 2016.

[72] H. Gomaa and E. M. Olimpiew, "Managing Variability in Reusable Requirement

Models for Software Product Lines," in *High Confidence Software Reuse in Large Systems: 10th International Conference on Software Reuse, ICSR 2008, Beijing, China, May 25-29, 2008 Proceedings*, H. Mei, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 182–185, 2008.

[73] M. Eduardo and E. W. Marcelo, "Modelo formal de pruebas funcionales de software para alcanzar el Nivel de Madurez Integrado 2 A formal model for the functional test of software to achieve maturity integrated level 2 Modelo formal de provas funcionais de software para alcançar o Nível d," vol. 24, no. 39, pp. 31–42, 2015.

[74] M. Toro, "Ingeniería del software y bases de datos, tendencias actuales," in *Patrones y Roles:Una disciplina para la construcción de Software*, Universida., I. R. S. M. D. L. Pérez, Ed. 2000.

[75] G. Canfora and F. Ruiz González, "Procesos Software: características, tecnología y entornos," *Novática Rev. la Asoc. Técnicos Informática*, no. 171, pp. 5–8, 2004.

[76] F. Ruiz, "Ingeniería de Procesos Software," *Engineering*, 2009.

[77] T. E. Journal, "SPT," no. 5, 2004.

[78] Harevalop, "Industria de Software." [Online]. Available: http://www.ivoox.com/industria-software-audios-mp3_rf_838659_1.html.

[79] D. de la lengua Española, "Sistemático," 2006. [Online]. Available: http://www.wordreference.com/definicion/sistemático.

[80] a Montilva, "Desarrollo de Software Basado en Líneas de Productos de Software Contenidos La idea básica :," pp. 1–34, 2006.

[81] and R. K. Bass, Len, Paul Clements, "Chapter 1: What is Software Architecture?," *2nd ed. Addison-Wesley Professional*, 2003. [Online]. Available: https://msdn.microsoft.com/en-us/library/ee658098.aspx.

[82] S. Gonnet, R. Holzer, H. Melgratti y H. Leone., Administración de Versiones de Modelos en una Herramienta de Soporte para Análisis y Diseño Orientado a Objetos.", GIPSI - Dep. Sistemas -Fac. Reg Santa Fe- Univ. Técnológica Nacioanal. Argentina, 2012.

[83] T. Ziadi and J. M. Jezequel, "Software product line engineering with the UML: Deriving products," *Softw. Prod. Lines Res. Issues Eng. Manag.*, pp. 557–588, 2006.

[84] B. P. Lamancha, "Gestión de las Pruebas," Centro de Ensayos de Software,Universidad de la República, Montevideo, Uruguay, pp. 2–7, [Online]. Available: http://www.ces.com.uy/documentos/imasd/CESPRIS_Gestion_Testing.pdf

[85] L. Gonz, "Método Para Generar Casos De Prueba Funcional En El Desarrollo De Software," vol. 8, no. 15, pp. 29–36, 2009.

[86] D. Yin, RK and Campbell, "Case Study Research: Design and Methods Thousand Oaks Sage Publications," *Polit. Educ. Assoc. Bull.*, 2003.

[87] C. Robson, "Real world research. 2nd," *Ed. Blackwell Publ. Malden*, 2002.