risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Iberian Journal of Information Systems and Technologies

Recebido/Submission: 14/11/2022
Aceitação/Acceptance: 16/02/2023

Aplicación del modelo de vistas de arquitectura 4+1 en la gestión de una línea de productos de software

Anderson Jojoa-Giraldo[1], Félix Fernández-Peña[2, 3], Pilar Urrutia-Urrutia[2], Fernando Ibarra-Torres[2]

anderson.jojoa464@comunidadunir.net; fo.fernandez@uta.edu.ec; elsapurrutia@uta.edu.ec; of.ibarra@uta.edu.ec

[1] Universidad Internacional de la Rioja, Avda. de la Paz, 137, 26006, Logroño, España.

[2] Universidad Técnica de Ambato, Av. Los Chasquis y Río Payamino, Ambato, Ecuador.

[2] Universidad Espírito Santo,Km 2.5 vía a Samborondón, Guayaquil, Ecuador.

Pages: 583-594

Resumen: El desarrollo de una arquitectura basada en modelos permite personalizar y añadir funcionalidad con mayor facilidad. Su aplicación en la gestión de una línea de productos de software pretende la reutilización de componentes y la gestión de cambio con eficiencia. Este trabajo propone la definición de un metamodelo que abstrae el dominio conceptual y físico de una línea de productos de software a través del modelo de vistas de arquitectura 4+1. Su instanciación permite modelar el desarrollo de software en término de activos básicos y el desarrollo de productos que usan dichos activos. El Centro de Proceso de Datos de la Cámara de Comercio de Cali fue el escenario escogido para la validación de esta propuesta. En este ámbito se trabajó, y documentó, el desarrollo y mantenimiento de productos de software durante dos años. Los resultados finales permitieron validar experimentalmente un incremento del nivel de eficiencia de la producción en un 39% como promedio.

Palabras-clave: arquitectura basada en modelos; líneas de productos de software, modelo de vistas de arquitectura 4+1.

Applying the 4+1 architectural view model in the management of a software product line

Abstract: A model-driven architecture makes it easier to add and to customize functionalities. Its use in the management of a software product line aims at the reuse of components and efficient change management. This work proposes the definition of a metamodel abstracting the conceptual and physical domain of a software product line through a 4+1 architectural view model. Its instantiation allows modeling software development in terms of basic assets and the development of products that use these assets. The Data Processing Center of the Cali Chamber of Commerce was the setting chosen for the validation of this proposal. In this scenario, the development and maintenance of software

products took place for over two years. The final results show an average increase of production efficiency of 39%.

Keywords: *Model driven architecture; software product lines; 4+1 architectural view model.*

1. Introducción

El desarrollo de productos de software en la actualidad lleva innegablemente a la búsqueda de la escalabilidad y eficiencia del desarrollo basado en la reutilización (Hillenbrand, 2021). En este sentido, es tendencia actual el extender la funcionalidad de los activos de la empresa al reutilizar sus componentes de software. Rinker, & Waltersdorfer et al. (2021) han concluido que se plantea hacerlo abarcando diferentes nichos de mercado con familias de productos de software montadas en una línea de productos de software (SPL).

Una SPL tiene como principal objetivo proveer una plataforma común para la derivación de sistemas particulares (Rinker, & Waltersdorfer et al., 2021). El objetivo es disminuir el tiempo y los costos de implementación. Para lograrlo se necesita que esté caracterizada por un nivel de flexibilidad que le permita adaptarse a nuevos y cambiantes requisitos. La configuración de la variabilidad es la clave para garantizar un nivel de flexibilidad adecuado (Cazzola & Favalli, 2022) (Horcas, Cortiñas, Fuentes & Luaces, 2022).

En este trabajo se describe el metamodelo creado para la abstracción de la gestión de la variabilidad de una SPL como un modelo de vistas de arquitectura 4+1. El objetivo planteado fue dotar de un conjunto de artefactos que permiten la definición del modelo específico de una SPL para la gestión de su variabilidad centrada en la eficiencia de la creación y mantenimiento de productos de software.

Los artefactos propuestos fueron validados experimentalmente en el ámbito del desarrollo de diez productos de software en el Centro de Proceso de Datos de la Cámara de Comercio de Cali en el transcurso de dos años. El presente artículo muestra los resultados obtenidos y está estructurado de la siguiente manera. En la sección 2 se discuten los resultados de trabajos relacionados. En la sección 3 se presenta la solución propuesta. En la sección 4 se analizan los resultados obtenidos y en la sección 5 se arriba a conclusiones.

2. Trabajos Relacionados

En la actualidad existe un marcado interés en el estudio de procesos de negocio dinámicos (Amdah et al., 2021) (Rinker et al., 2021) (Habba et al., 2021) (Horcas, Cortiñas, Fuentes & Luaces, 2022). Amdah et al. (2021) resaltan la importancia de una ingeniería basada en modelos para hacer que el desarrollo de aplicaciones sea más flexible. Así mismo, consideran que este tipo de desarrollo provee un marco de trabajo comprensible para definir sistemas interconectados, reduciendo así la complejidad del desarrollo del software.

En este contexto, Gottschalk et al. (2022) resaltan la importancia de la trazabilidad de las decisiones de diseño de software basado en componentes. En un escenario donde

se pretende aumentar el nivel de reutilización, se considera fundamental la gestión de la relación característica-componente, y se destaca la importancia del uso de una arquitectura basada en modelos (Gottschalk et al., 2022). Por su parte, Santos et al. (2021) han propuesto el uso del modelo de vistas de arquitectura 4+1 para describir la arquitectura de una interfaz de usuario gráfica para el procesamiento de imágenes.

En el análisis de los resultados obtenidos de Santos et al. (2021), se destaca la utilidad de este modelo al conceptualizar la arquitectura de un software donde se utiliza un lenguaje de programación visual con capacidad de procesamiento distribuido y paralelo. También resaltan la utilidad de algunos artefactos en la definición de la arquitectura. Entre ellos, el diagrama de paquetes, el diagrama de secuencia, y el diagrama de componentes (Santos et al., 2021).

Rinker et al. (2021) se enfocaron en la modelación de una SPL. En su estudio, destacan la importancia de integrar vistas desde múltiples disciplinas y atender las diferencias semánticas que pueden surgir de la interpretación de diferentes descripciones de un mismo activo. Las diferencias en la semántica también son atendidas por Melo (2021) en su estudio sobre la integración de dominios de aplicación y la infraestructura genérica de ciudades inteligentes. En este sentido, es que en nuestra propuesta consideramos la definición de modelos específicos que permitan unificar conceptualizaciones tanto en el dominio conceptual como en el físico.

Con el estudio de Rinker et al. (2021) compartimos el enfoque en la reutilización de componentes. La principal diferencia es que su propuesta está enfocada en entornos de operación de autómatas mientras que la nuestra está definida para la administración de sistemas de información. Su aporte fundamental está en la transformación de modelos mientras que nuestra propuesta se centra en mecanismos de integración basados en modelos de semántica común, de acuerdo con el dominio en que se definen.

Por su parte, Vahdati & Ramsin (2022) proponen la gestión de una arquitectura basada en modelos encapsulada en servicios de modelación y servicios de transformación automática de modelos. De esta manera, la semántica de los servicios se encarga de encapsular las diferentes interpretaciones que requieran tener las vistas de la arquitectura de software. La limitante fundamental de esta propuesta está asociada a la complejidad que agrega a la gestión de la línea de productos de software con la creación de una capa extra de abstracción basada en servicios.

Horcas, Cortiñas, Fuentes & Luaces (2022) han trabajado en la gestión de diferentes niveles de granularidad de la variabilidad en una línea de productos de software. Sin embargo, su propuesta se limita a procesos de la ingeniería web. Habba et al. (2021) proponen la alineación de los sistemas en un entorno de negocio a través de la alineación del diagrama de clases (como artefacto para describir al sistema) con un modelo de gestión de procesos de negocio (como artefacto para describir los procesos de negocio). La propuesta resulta interesante para sistemas heredados en que existe una concepción del sistema aislada del proceso de negocio. Sin embargo, en nuestra opinión, una visión integral desde la perspectiva de vistas del dominio conceptual y físico garantizan un nivel de cohesión semántica mucho mayor y, al mismo tiempo, la flexibilidad en la adaptación de los activos en la construcción de los productos requeridos por la entidad de negocio.

En general, no encontramos otros trabajos con el enfoque en que hemos trabajado. Los trabajos relacionados consultados sí demuestran la actualidad y pertinencia del tema tratado. Desde el punto de vista de la validación de resultados, Hillenbrand (2021), en busca de una teoría para la modelación del desarrollo de sistemas, considera que la aplicabilidad de determinado modelo para el desarrollo de un software va a depender del proceso productivo específico en que se concibe. En este sentido, consideramos que la evaluación experimental se acepta como la vía para la validación de nuestra propuesta. Hillenbrand (2021) considera además a la eficiencia del proceso de producción de software como la variable fundamental en la interpretación de la utilidad de determinado modelo, aspecto que consideramos en nuestro trabajo para determinar el valor de esta propuesta.

3. Propuesta de Solución

Se propone el uso de un metamodelo que abstrae los aspectos arquitectónicos de una SPL a través del modelo de vistas de arquitectura 4+1 (ver figura 1). Este metamodelo está adaptado a las actividades esenciales para el desarrollo de los activos básicos y el desarrollo de los productos. El mismo se constituye en los dominios conceptual y físico. El dominio conceptual apoya la actividad de gestión de los activos básicos. En el dominio físico se definen los productos de software a través de procesos de composición y orquestación de los activos de la empresa.

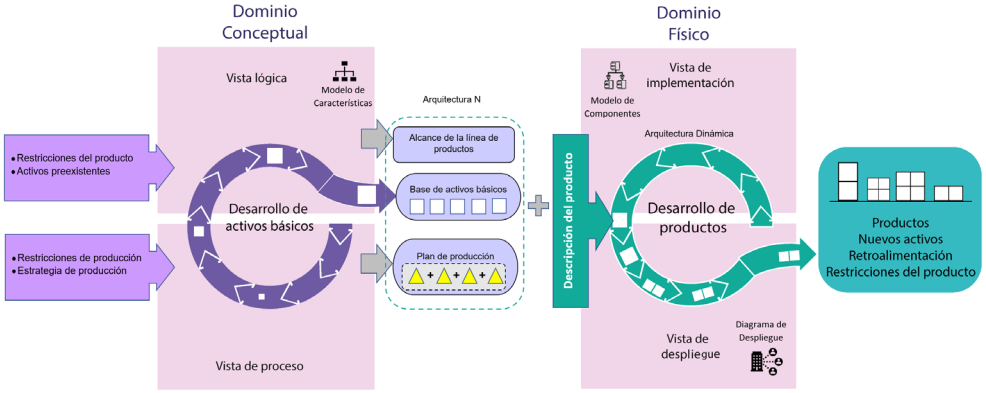

Figura 1 – Metamodelo de vistas de arquitectura 4+1.

El dominio conceptual se caracteriza a través de las vistas lógica y de procesos. La vista lógica tiene como entrada las *restricciones del producto* ($\mathbf{E}_{cl_1}$) y los *activos preexistentes* ($\mathbf{E}_{cl_2}$). Las salidas de la vista lógica son *el alcance de la línea de productos* ($\mathbf{S}_{cl_1}$) y *la base de activos básicos* ($\mathbf{S}_{cl_2}$). La gestión de los activos a nivel lógico global tiene lugar a través del *Modelo de Características*, que aglutina las relaciones lógicas, tal y como se muestra, para el caso de estudio, en la figura 2.

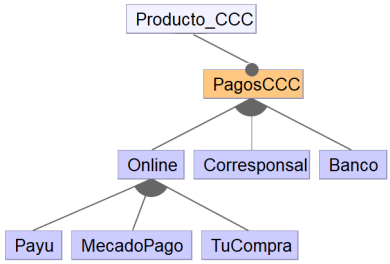

Figura 2 – Ejemplo modelo de características del Caso de Estudio.

El modelo de características capta las características comunes y específicas de la línea de productos. En este modelo, cada producto está representado por una configuración única de un conjunto de características. Por ejemplo, "RENOVACIÓN DE MERCANTIL/ ESAL" esta compuesto por el componente "PagosCCC", que en esta configuración corresponde a pago por derechos de renovación en Corresponsal, o Banco, o pago en línea por la pasarela PayU {*PagosCCC, Corresponsal, Banco, Online, PayU*}; a diferencia del producto "CERTIFICADO ELECTRÓNICO" que solo acepta la configuración de pago en línea { *PagosCCC, Online, TuCompra*} ya que el certificado firmado electrónicamente se descarga inmediatamente después del pago, lo que no se podría hacer en un corresponsal o banco. El conjunto de todas las combinaciones posibles de características representa a los posibles miembros de la línea de productos.

La vista de procesos, en cambio, tiene como entrada las *restricciones de producción* ($\mathbf{E}_{cp1}$) y la *estrategia de producción* ($\mathbf{E}_{cp2}$). La salida de la vista de procesos es el *plan de producción* ($\mathbf{S}_{cp1}$). En este ámbito, la decisión fue utilizar los diagramas de secuencia, actividades, e interacción para la descripción de los procesos del SPL.

El alcance de la línea de productos es una lista enumerada de nombres de productos; describe los ítems de la línea de productos. Se elabora teniendo en cuenta las similitudes y diferencias funcionales de los productos. Todos los productos incluidos quedan validados contra las funcionalidades del modelo de características.

La base de activos básicos incluye a los componentes núcleo, que son el punto de partida para la generación de productos de software. La documentación de cada componente núcleo incluye su *diseño*, *casos de uso de prueba* y *plan de producción*. *El plan de producción* define el proceso que permite utilizar al componente en el desarrollo de un producto.

El dominio físico se caracteriza a través de las vistas de implementación y de despliegue. Estas dos vistas reciben como entrada a las salidas de las vistas del dominio conceptual $\mathbf{S}_{cl1}$, $\mathbf{S}_{cl2}$ y $\mathbf{S}_{cp1}$, y la *descripción del producto a desarrollar* ($\mathbf{E}_{f1}$). La vista de implementación gestiona la variación y configuración de los componentes en relación con el montaje y liberación de productos. Esta gestión tiene lugar a través del *Modelo de Componentes*, como se ilustra en la figura 3 para el caso de estudio en cuestión. Por su parte, la vista de despliegue caracteriza los nodos de la topología del hardware que soporta la línea de productos al ser desplegados con base el *Diagrama de Despliegue* ($\mathbf{S}_{fd1}$). En la figura 4

se muestra un diagrama de este tipo para el caso de estudio en cuestión, que representa el balanceo de carga para el producto “Certificados electrónicos” definido en la vista de despliegue.

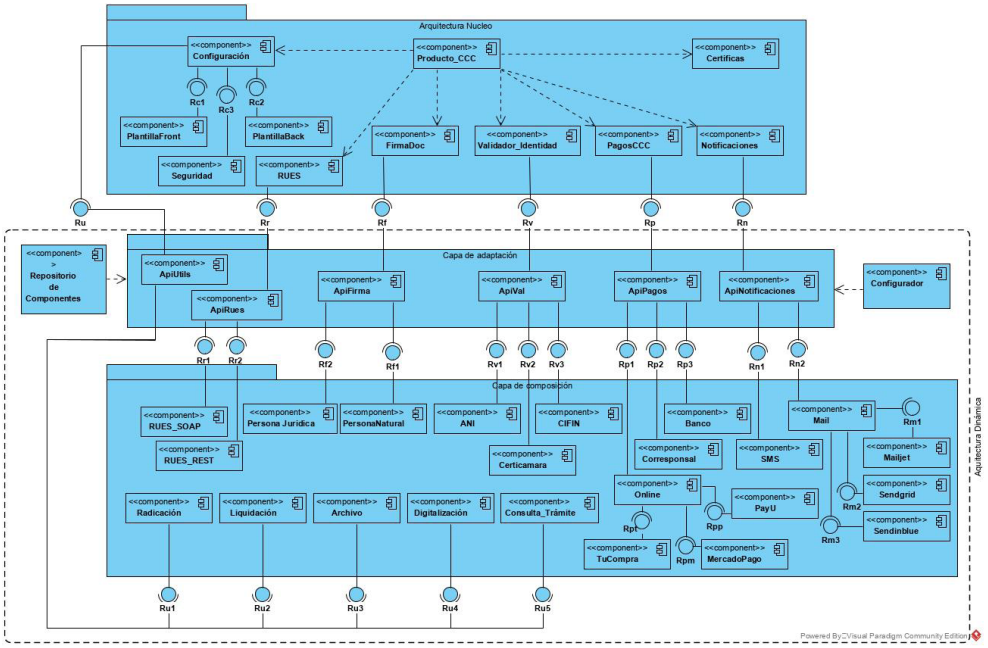

Figura 3 – Modelo de Componentes del Caso de Estudio.

El *Modelo de Componentes* representa la arquitectura de software de la SPL. Existe una correspondencia directa entre un conjunto válido de características del *Modelo de Características* y un elemento del *Modelo de Componentes*. Este modelo está formado por tres capas. Por un lado, tenemos la *Arquitectura Núcleo*, formada por componentes base, que no varían en el tiempo, y que son comunes a todos los productos. En el lado opuesto se encuentra la *Arquitectura Dinámica*, conformada por componentes de composición dinámica, con alta cohesión y bajo acoplamiento, y con capacidad de evolucionar en el tiempo. La tercera capa está formada por componentes de relación, los que se encargan de conectar a las dos arquitecturas anteriormente descritas.

Los productos individuales, que se crean en la instanciación de la vista de implementación, constituyen salidas transitorias de la línea de productos. Estos pueden ser descartados o recreados, según sea necesario. Por su parte, la visión de la vista de despliegue se centra en la disponibilidad, confiabilidad, rendimiento y escalabilidad de los productos de software desplegados. El resultado final incluye *el producto de software* ($\mathbf{S_{fi1}}$), *la retroalimentación* al proceso de desarrollo ($\mathbf{S_{fi2}}$ **y** $\mathbf{S_{fd1}}$), *el registro de nuevos activos* ($\mathbf{S_{fi3}}$) y *la actualización de las restricciones de producción* ($\mathbf{S_{fi4}}$ **y** $\mathbf{S_{fd2}}$).

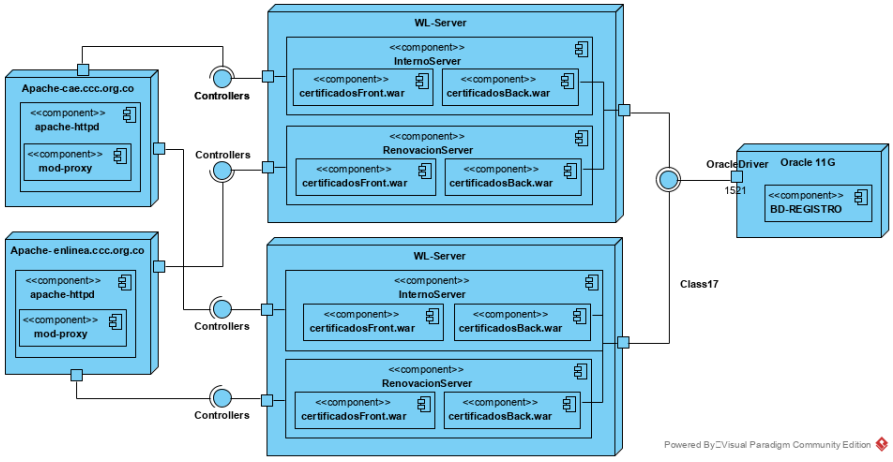

Figura 4 – Diagrama de Despliegue para el Producto “Certificados Electrónicos”.

3.1. Pasos para la aplicación del metamodelo propuesto

La aplicación del metamodelo propuesto requiere de un proceso cíclico que sigue los siguientes pasos:

1. Diagnóstico del estado actual de la organización.
   a. Construcción de la vista lógica.
   b. Construcción de la vista de proceso.
   c. Construcción de la vista de implementación.
   d. Construcción de la vista de despliegue.
2. Construcción del modelo específico de la organización.
   a. Construcción del modelo desde la vista lógica.
   b. Construcción del modelo desde la vista de proceso.
   c. Construcción del modelo desde la vista de implementación.
   d. Construcción del modelo desde la vista de despliegue.
3. Definición del cronograma de una iteración.
   a. Definición del cronograma de cada vista del metamodelo.
   b. Combinación de los cronogramas de las vistas.
   c. Refinación del cronograma general.
   d. Determinación de puntos de corte.
   e. Control y seguimiento del cronograma.
5. Repetir desde el paso 1.

3.2. Resultado de la Experimentación

Como parte de la experimentación en el ámbito del caso de estudio se obtuvo el modelo específico para el Centro de Proceso de Datos de la Cámara de Comercio de Cali centrado

en SPL. Se trabajó desde enero/2020 a enero/2022. Se concientizó a la gerencia de la empresa sobre la necesidad de organizar los procesos de producción. Se obtuvieron 10 productos de la misma familia (sistemas de información), 34 características e igual cantidad de componentes. 11 de estos componentes son de la arquitectura núcleo y 23 de la arquitectura dinámica. Otros 6 componentes pertenecen la capa de adaptación.

Se decidió utilizar una implementación DAO para el acceso a la capa de datos desde los servicios del back-end. Estos, a su vez, exponen la información en servicios REST, los mismos que son consumidos por la capa del front-end. Los productos se clasificaron en dos categorías: servicios públicos (consumidos por cualquier interesado) y servicios privados (utilizados por funcionarios de la entidad).

4. Discusión de Resultados

La implementación del modelo específico del caso de estudio generó un incremento del nivel de productividad en los proyectos de desarrollo de software, con respecto al diagnóstico inicial. A continuación, se muestra cómo el valor de la eficiencia de la producción fue aumentando paulatinamente.

El valor de productividad del desarrollo (PD) se calculó teniendo en cuenta el tamaño del producto (TD), dado por la cantidad de requisitos a implementar, y la cantidad de recursos (RD), expresado como el tiempo, en meses, requerido para el desarrollo. El cálculo de PD se hizo según la formulación 1.

$$PD = \frac{TP}{R} \tag{1}$$

Para el caso de las acciones de soporte, la productividad (PS) se calculó a partir del tamaño del producto (TI), expresado como la cantidad de incidencias detectadas, y de la cantidad de recursos (RS), expresada como el tiempo de respuesta, en meses, necesario para resolver las incidencias. El cálculo de PS se hizo según la formulación 2.

$$PS = \frac{TI}{R} \tag{2}$$

Por su parte, la eficiencia total (E) se calculó, teniendo en cuenta un coeficiente de desarrollo (CD) 0.7 y coeficiente de soporte (CS) 0.3, según la formulación 3.

$$E = (CD * PD) + (CS \times PS) \tag{3}$$

Los resultados del cálculo de la eficiencia se muestran en la tabla 1.

Estos datos de productividad se reflejan, de manera gráfica en la figura 5. A partir de los resultados obtenidos, se demuestra un aumento de la productividad en los proyectos después de aplicado el modelo específico del Centro de Proceso de Datos de la Cámara de Comercio de Cali, con una tasa de crecimiento promedio semestral en la eficiencia

del 39%. Para ello, se consideró el valor más reciente de eficiencia (presente) y el primer valor de eficiencia (pasado), con un valor de cantidad de iteraciones 4, de acuerdo a la formulación 4 para determinar un valor de tasa de crecimiento semestral (TC).

Proyecto	TP	RD	PD	TI	RS	PS	E
Renovación de Mercantil/ESAL. 2020	60	16	3,75	18	8	2,25	3,30
Proponentes. 2020	65	14	4,64	24	6	4	4,45
Matricula de Mercantil/ESAL. 2020	50	7	7,14	15	5	3	5,90
Certificado electrónico. 2021	55	6	9,17	10	2	5	7,92
Constitución de SAS. 2021	40	4	10,00	6	1	6	8,80
Consulta del Estado Trámite. 2021	35	3	11,67	0	1	0	8,17
Inscripción de actos y documentos. 2022	45	4	11,25	8	2	4	9,08
Autoliquidación.2022	35	3	11,67	7	1	7	10,27
Libros Electrónicos. 2022	40	3	13,33	0	2	0	9,33
Consulta expedientes. 2022	42	3	14,00	16	2	8	12,20

Tabla 1 – Productividad de los proyectos durante su desarrollo

$$TC = \left[\frac{presente^{1/n}}{pasado} - 1\right] 100\% \quad (4)$$

Este incremento en la eficiencia se logró a través de la reutilización de los componentes que constituyen *la base de activos básicos* almacenados en el período evaluado. Paulatinamente, los recursos de desarrollo y soporte disminuyeron con respecto a los requerimientos especificados en los proyectos. En el proyecto "Renovación de Mercantil / ESAL. 2020" se desarrollaron los componentes de firma electrónica, envío de SMS y portal de pagos. Estos componentes fueron utilizados en otros proyectos.

Esto implica un ahorro significativo en las etapas de análisis, diseño y desarrollo para los proyectos subsiguientes en lo referente a estos componentes. En el proyecto "Certificado electrónico. 2021", por ejemplo, se utilizaron 6 recursos para 55 requerimientos en desarrollo y 2 recursos para 10 casos de soporte. La situación es muy diferente a la del primer proyecto, en cuyo caso no se contaba con activos básicos previamente desarrollados.

La repercusión del metamodelo propuesto implica que, hoy en día, en el Centro de Proceso de Datos de la Cámara de Comercio de Cali los proyectos de software se piensan como productos que se ensamblan en una línea de productos; los componentes se sacan de la base de activos básicos y cada componente nuevo se concibe como un activo potencialmente reutilizable para futuras implementaciones.

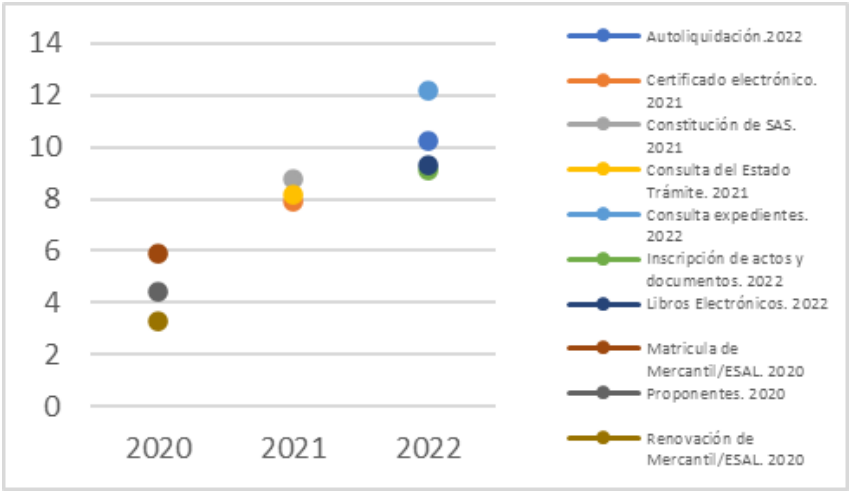

Figura 5 – Evolución del nivel de productividad durante el período analizado.

5. Conclusiones

El metamodelo propuesto está centrado en líneas de producción de software. Su instanciación implica definir el dominio conceptual y físico de la SPL. Para ello, se utilizan artefactos que permiten caracterizar los activos y productos en cada una de las etapas.

La *base de activos básicos* constituye el punto de partida para la generación de productos de software. Más allá, la reutilización se sustentó en la gestión de la variabilidad que, conceptualmente, se recoge en el *Modelo de Características*, y funcionalmente se gestiona a través del *Modelo de Componentes*.

La documentación generada se convierte así en un activo más de la entidad. Este activo adicional resultó determinante en la reutilización de componentes de software y el incremento gradual de la eficiencia en la creación y soporte de productos de software.

La guía de aplicación del metamodelo determinó un proceso cíclico de desarrollo del software que utiliza como materia prima a los activos de la entidad y modela cada producto a través de un proceso de orquestación de los componentes existentes y la creación de otros nuevos. Su aplicación implicó un cambio cultural, ya que los proyectos de software se conciben como proyectos de desarrollo de un producto.

En este sentido, cada producto es el resultado del ensamblaje de componentes en una línea de productos. Del mismo modo, cada nuevo componente se concibe como un activo que se agrega a la *base de activos básicos* de la entidad. Por su parte, los artefactos utilizados en la documentación, desde la perspectiva de cada una de las vistas, permiten la interacción de los diferentes actores, desde los desarrolladores y usuarios hasta el personal de configuración de software y gerentes de proyecto.

Como resultado del proceso de validación de la propuesta, se obtuvieron 10 productos de la misma familia, en cuyo desarrollo se evidenció un incremento de la eficiencia de los equipos de desarrollo de hasta el 39%.

Agradecimiento

El desarrollo de la presente investigación tuvo lugar como parte de un trabajo de fin de máster de la UNIR. La contribución de investigadores de la Universidad Técnica de Ambato en la redacción de este artículo científico tuvo lugar como parte de la ejecución del proyecto de investigación SFFISEI-02, financiado por la Dirección de Investigación y Desarrollo de dicha universidad. La Universidad Espíritu Santo patrocina el desarrollo de este trabajo poniendo a disposición su plataforma tecnológica para la comunicación entre los participantes en la investigación.

Referencias

Amdah, L., Essadi, N., & Anwar, A. (2021). A model-driven architecture for collaborative business processes. *International Journal of Advanced Computer Science and Applications*, 12(8), 719-725.

Cazzola, W., & Favalli, L. (2022). Towards a recipe for language decomposition: Quality assessment of language product lines. *Empirical Software Engineering*, 27(4) doi:10.1007/s10664-021-10074-6

Gottschalk, S., Yigitbas, E., & Engels, G. (2022). Model-driven continuous experimentation on component-based software architectures. *In Proceedings of 19th International Conference on Software Architecture Companion*, (pp. 20-24). IEEE.

Habba, M., Chaouni, S., & Fredj, M. (2021). *Aligning software system level with business process level through model-driven architecture*. International Journal of Advanced Computer Science and Applications, 12(10), 174-183.

Hillenbrand, A. (2021). Towards a theory of models in systems development modeling. *In Proceedings of the 9th International Conference on Model-Driven Engineering and Software Development*., (pp. 322-329). Scitepress.

Horcas, J., Cortiñas, A., Fuentes, L., & Luaces, M. (2022) Combining multiple granularity variability in a software product line approach for web engineering. *Information and Software Technology*, 148, 106910.

Melo, P. (2021). A model-driven middleware approach to reduce the semantic gap between application domains and the generic infrastructure of smart cities. *In Proceedings of the 24th International ACM/IEEE Conference on Model-Driven Engineering Languages and Systems*, (673-678). IEEE.

Rinker, F., Waltersdorfer, L., Meixner, K., Winkler, D., Luder, A., & Biffl, S. (2021). Continuous integration in multi-view modeling: a model transformation pipeline architecture for production systems engineering. *In Proceedings of the 9th International Conference on model-driven engineering and software development*, (286-293). Scitepress.

Santos, R., Soares, M., & Oliveira, D. (2021). A System Architecture in Multiple Views for an Image Processing Graphical User Interface. *In Proceedings of the 23rd International Conference on Enterprise Information Systems*, vol.2, (pp. 213-223). Scitepress.

Vahdati, A., & Ramsi, R. (2022). Modeling and Model Transformation as a Service: Towards an Agile Approach to Model-Driven Development. *In Proceedings of the 6th International Conference on Lean and Agile Software Development*, (116-135). Springer.