# Universidad de las Ciencias Informáticas

## Facultad 4


# Título: Propuesta para la implantación de una biblioteca de componentes reutilizables en la UCI.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

**Autores:**

Tayché Capote García

José Manuel Rodríguez Penado

Rubén Rodríguez Torres

**Tutores:**

MSc. Fernando Antonio Peón Sánchez

Lic. Miguel Sancho Fernández

**Ciudad de La Habana, junio de 2007**

*"No hacen falta alas*
*para hacer un sueño*
*basta con las manos*
*basta con el pecho*
*basta con las piernas*
*y con el empeño."*

*Silvio Rodríguez.*

**Declaración de Autoría**

Tayché Capote García, José Manuel Rodríguez Penado y Rubén Rodríguez Torres, se declaran como únicos autores de este trabajo y autorizan a la Facultad 4 y la Dirección de la Infraestructura Productiva de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los 25 días del mes de junio del año 2007.

| Tayché Capote García | José Manuel Rodríguez Penado | Rubén Rodríguez Torres |
|---|---|---|
| (autora) | (autor) | (autor) |

| MSc. Fernando Antonio Peón Sánchez | Lic. Miguel Sancho Fernández |
|---|---|
| (tutor) | (tutor) |

**Opinión del usuario del trabajo de diploma**

El Trabajo de Diploma, titulado “Propuesta para la implantación de una biblioteca de componentes reutilizables en la UCI”, fue realizado en la Dirección General de la Infraestructura Productiva. Esta entidad considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface

☒ Totalmente

☐ Parcialmente en un ____ %

La Universidad de las Ciencias Informáticas cuenta con los recursos necesarios para lograr automatizar todos sus procesos y convertirse en líder de la Industria Cubana de Software. Entre los procesos más importantes, como en toda empresa de software, están los procesos reusabilidad de componentes, pues esto hace que la producción de nuevos productos sea más rápida, eficiente y con menos costos. Desde hace varios años en la UCI se esta analizando la necesidad de contar con un buen sistema que gestione este importante renglón en la cadena productiva.

Actualmente en la UCI está instalado el sistema Gforge que permite una primera aproximación a la solución del problema, pero no se usa de manera eficiente, ni recoge todos los intereses de la entidad, no cubre las necesidades para el buen desempeño de sus actividades, limita el dinamismo que se necesita y no permite la introducción de actividades propias de esta institución. El trabajo realizado por los estudiantes, aunque en el mismo no están cumplidas todas las etapas del desarrollo del software, marca el comienzo de un sistema que completará el sistema de gestión de componentes de la UCI y que responderá a los requerimientos que para este tipo de actividad necesita la universidad.

Y para que así conste, se firma la presente a los 12 días del mes de junio del año 2007.

| | |
|---|---|
| Ing. Renier Pérez García | Director Técnico de la IP |
| Representante de la entidad | Cargo |
| _____ | _____ |
| Firma | Cuño |

**Opinión del tutor del trabajo de diploma**

**Título: "**Propuesta para la implantación de una biblioteca de componentes reutilizables en la UCI"

**Autores:** Tayché Capote García, José Manuel Rodríguez Penado, Rubén Rodríguez Torres

La naciente Industria Cubana de Software, necesita que todos sus procesos se hagan de manera eficiente, entre ellos la reutilización de componentes o activos. Este tendrá que convertirse en uno de los mecanismos más eficientes para lograr la disminución eficaz y rápida de los tiempos de entrega, así como de los costos de desarrollo de los productos y además contribuir a mejorar la calidad de los mismos, teniendo como objetivo asegurar la disponibilidad de activos, para apoyar el desarrollo de un producto, manteniendo información relevante de cada componente usado, su especificación técnica, historia o registro de uso, clasificación, documentación, etc.

En la Universidad de las Ciencias Informáticas (UCI), no se cuenta con una biblioteca de componentes de software reutilizables adecuada a las características del proceso productivo de la misma, por esta razón entre otras muchas, tampoco existe una cultura de reutilizar componentes ya existentes, sino que se comienzan a desarrollar desde cero cada vez que se inicia un nuevo proyecto. Por todo ello no se trata, no solo de sugerir una biblioteca sino además de que está se acompañe de una metodología para su uso y correspondiente asimilación al proceso de producción de software.

Los autores, estudiantes de la Facultad #4, de excepcionales cualidades humanas, que se han destacado en todas las tareas que se les han asignado, en particular las encomendadas por la FEU, la UJC, donde se han desempeñando como cuadros, además cumplieron satisfactoriamente misiones en el exterior, cuentan con resultados académicos sobresalientes, mostraron en todo momento, independencia, originalidad, creatividad, laboriosidad, que nos obligaron a capacitarnos cada día más en los conocimientos en una rama tan nueva para nosotros y tan necesaria, que en ocasiones implicaba tener acudir a expertos para poder dar respuesta a sus exigencias, con mucha responsabilidad, pero sobre todo mucha tenacidad, amor por lo que hacen y deseos de lograr el mejor resultado, aún sabiendo que en la etapa de su trabajo era casi imposible lograr ese grado de perfeccionamiento.

De cada uno por separado podemos hablar mucho, de Pepe por ser siempre el cadete que conocí y un incansable batallador antes las adversidades, con una capacidad de resolver cualquier dificultad; de Ruben su seriedad ante todas las cosas y por su capacidad de soportar las más duras críticas sin molestarse. Por último Tayché, mi siempre presidente de la FEU,

capaz de cambiar en un momento para un tema totalmente desconocido y convertirse en pocos días en una experta del tema. Todos serán recordados como los muchachos del Comité Organizador de Informática 2007, donde también se destacaron por su capacidad de organizadores.

Esperamos que pronto estemos en un acto similar, solo que ese momento sea en sus defensas de futuros grados científicos en la Especialidad de Informática.

Por todo lo anteriormente expresado consideramos que los estudiantes están aptos para ejercer como Ingenieros en Ciencias Informáticas y proponemos que se le otorgue al Trabajo de Diploma la calificación de 5 puntos.

_____ _____

Firma Fecha

_____ _____

Firma Fecha

**Datos de Contacto**

Tutor: Fernando Antonio Peón Sánchez, ciudadano cubano, fue miembro del 1er Contingente del Destacamento Pedagógico Manuel Azcunce Domenech (1972), Graduado como: Profesor en Matemática, por el Instituto Superior Pedagógico “Enrique José Varona” (ISPEJV) en 1977, Licenciado en Educación en la especialidad de Matemática, por el ISPEJV en 1979, Licenciado en Matemática, por la Universidad de La Habana en 1988, Máster en Informática Aplicada a la Ingeniería y a la Arquitectura, por el CREPIAI, CUJAE en 1995. Profesor Auxiliar, tiene 35 años de trabajo como profesor y de ellos 30 en la educación superior. Actualmente es Director de Formación Posgraduada de la Universidad de las Ciencias Informáticas desde el 2003, es fundador de la UCI y en su primer curso fue Decano de la Facultad #4. Email: fpeon@uci.cu

Tutor: Miguel Sancho Fernández, Ciudad de la Habana (1980). Licenciado en Ciencia de la Computación, en la Facultad de Matemática y Computación de la Universidad de la Habana en 2003. Actualmente cursa la Maestría en Ciencia de la Computación. Profesor Instructor Adjunto de la UCI, 3 años de trabajo como profesor en la educación superior. Se desempeña laboralmente como Cuadro Profesional de la UJC en la UCI en la responsabilidad de funcionario que atiende la producción, fundador de la UCI. Email: sancho@uci.cu

**Agradecimientos Compartidos**

*A Sancho, por su ayuda, su ciencia y sus despistes durante la realización de la tesis.*

*A Peón, por haber sido pesado, dedicado, incondicional y único con nosotros.*

*A todos los que han sido nuestros profesores, los buenos y los no tan buenos, porque siempre nos enseñaron algo importante.*

*A nuestras segundas escuelas la FEU y la UJC, por hacer de la UCI mucho más que una universidad y prepararnos para la vida.*

*A la UCI, por ser nuestra casa estos 5 años, por mantenernos conectados al Futuro y la Revolución.*

*A la Revolución, porque gracias a ella hoy somos profesionales, por confiar siempre en nosotros.*

*A todos los que aportaron tiempo, conocimientos y amor en este trabajo.*

***Pepe***

*A mis padres por estar conmigo incondicionalmente, sin sus enseñanzas no estaría aquí ni sería quien soy ahora.*

*A mis abuelos, tíos, primos y hermanos por estar siempre a mi lado y conformar eso lindo que se llama familia.*

*A mis dos compañeros de tesis, después de todo, no siempre estar de acuerdo, también es bueno, gracias a eso terminamos la tesis.*

*A Rubén, Yoan, Ariagna, Alionuska, Lidice, Nely, Carlitos, Yoannis y Erik por saber ser amigos a pesar de todo y sobre todo, gracias por los trabajos voluntarios.*

*A Peón por ser más que un tutor, o mejor por ser mi tutor los 5 años.*

*A mi novia por sobre todo ser mi amiga y entenderme.*

*A la gente con quien compartí en el ITM, fueron ustedes quienes me enseñaron a pensar distinto.*

*A los amigos que descubrí en Venezuela, sin su ayuda no lo hubiera logrado.*

*A los que están o estuvieron y han hecho que la UCI sea más que una simple universidad.*

***Tayché***

*A papi, que aunque no esté físicamente lo he sentido más cerca que nunca en estos últimos 8 años, por hacer de mí lo que soy hoy, sin tus enseñanzas y amor no sería nada.*

*A mami, por ser mi vida entera, por vivir por y para mí y por junto a papi, haber hecho que llegara hasta aquí…todos mis resultados son por ustedes.*

*A nana, piti y yoyo, por ser parte indispensable en mi vida y hacerme sentir muy querida.*

*A mis primos, más bien hermanos: Thais, Jeily y Jorgito, por su cariño, sus travesuras y entregarme su corazón en cada gesto o palabra.*

*A mis compañeros de tesis, si no fuera por ustedes hubiera sido muy difícil poder hacerla sola desde cero tan tarde, gracias por las discusiones y las risas.*

*A mi novio, por su amor, comprensión y amistad, sin ti no hubiera sido posible.*

*A Peón, "mi decano" estos 5 años, por sus regaños, enseñanzas, paciencia y dedicación, eres un tutor único.*

*A personas excepcionales como Nely, las profes Alicia, Alina y Rosa, Zulema y mamá Hilda, gracias por sus enseñanzas y apoyo, su ejemplo fue mi guía todos estos años.*

*A Taily, Lídice, Alionuska, Yoan, Carlitos, Silvano, Yoannis y Robertico (Dunn), por todos los instantes maravillosos que pasamos juntos, por su amistad incondicional.*

*A todos mis amigos y compañeros de la UCI, esos que han sido parte de algún momento de mi vida, a los que están y los que no, siempre me enseñaron y ayudaron mucho.*

***Rubén***

*A mi mamá y a mi papá por ser mi fuente de inspiración, ejemplo a seguir y paradigma como profesionales durante estos años de universidad.*

*A mi hermanita toti por estar siempre pendiente de mí y apoyarme en todas mis decisiones.*

*A Santiago por ser también mi papá y por haberme aconsejado siempre en el momento oportuno.*

*A mi familia por haber confiado en mí durante todos estos años, en especial a mi abuelita Nélida y mi tía Odalis.*

*A mis grandes amigos de la Universidad Pepe y Yoan por aguantarme estos años.*

*A Peón por ser más que el tutor de nuestra tesis, por las horas extras que nos ha dedicado y por la dedicación que ha tenido con nosotros.*

*A mis amigos de siempre, en especial a Osley que ha estado conmigo desde primer año apoyándome en todo lo que he necesitado.*

*A mis compañeros de tesis que a pesar de nuestros caracteres, este trabajo no hubiera sido posible sin su apoyo.*

*A Jessica por su confianza en mí y por el amor y la ternura que me ha dado.*

*A mis amigos del Comité Primario, en especial Haydee, Cealys y Dinia.*

*A mis compañeros y amigos de la UJC y la FEU por enseñarme algo más que informática.*

*A Carlitos, Yoannis, Ariagna y La Negra por estar siempre cuando los necesité.*

**Dedicatoria**

*Rubén:*

*A la memoria de mi mamá que siempre la llevo en mi corazón.*

*Tayché:*

*A papi y mami por ser mi vida, mi inspiración.*

*Pepe:*

*A mi mamá y mi papá por todo su amor.*

*En nombre de los tres, le dedicamos este trabajo a una persona en especial, por su entrega, constancia, rebeldía, confianza e ideas, porque él va al futuro, regresa y echa a andar sueños, imposibles para muchos, como la UCI, al Comandante de todas las batallas, a Fidel. Siempre seremos parte de su Tropa de Futuro.*

## Resumen

La reutilización de componentes o activos en la creciente industria del software, se ha convertido en uno de los mecanismos más realistas para lograr la disminución eficaz y rápida de los tiempos de entrega, así como de los costos de desarrollo de los productos y además contribuye a mejorar la calidad de los mismos. Las bibliotecas de componentes de software reutilizables, son repositorios donde se guardan estos componentes y tienen como objetivo asegurar la disponibilidad de activos, para apoyar el desarrollo de un producto de software. Mantienen información relevante de cada componente usado, como son la especificación técnica, historia o registro de uso, clasificación, documentación, etc.

En la Universidad de las Ciencias Informáticas (UCI), a pesar de que la reutilización de componentes se ha manifestado de forma aislada en algunas áreas, no se ha estimulado de forma ordenada, desde la Dirección de la Producción, ninguna política al respecto y no se cuenta con una biblioteca de componentes de software reutilizables adecuada a las características del proceso productivo de la UCI, que permita la utilización de activos elaborados en un proyecto determinado y que sea necesario para el desarrollo de otro proyecto de la Universidad. Por esta razón, tampoco existe una cultura de reutilizar componentes de software ya existentes, sino que se comienzan a desarrollar desde cero cada vez.

En el presente trabajo de diploma se realizan dos propuestas para la implantación de una biblioteca de componentes de software reutilizables en la UCI. La primera de forma inmediata, utilizando el Gforge como herramienta de colaboración ya instalada en un servidor de la Universidad, de forma que al implantarse en uno de los Polos Productivos, constituya una experiencia que sirva para comprobar la factibilidad de la misma. La segunda propuesta es el desarrollo, a la par que se pone en marcha la primera, de una biblioteca de componentes de software reutilizables que se adapte a las características del proceso productivo de la UCI.



## Palabras Claves

Reutilización, repositorio, biblioteca de componentes de software reutilizables.

# Índice

## Introducción

Desde el surgimiento de la informática, la producción de software se inició con el desarrollo de programas simples que implementaban algoritmos. El conjunto de problemas que debían resolverse utilizando soluciones de tipo software aumentaban continuamente, entonces se hicieron más complejos los sistemas a construir, los clientes más exigentes en cuanto a calidad y tiempo, y el equipo de desarrollo necesario para un proyecto iba creciendo continuamente.

Muchos de estos proyectos utilizaban componentes de software similares, los cuales eran desarrollados desde cero una y otra vez. Esto traía consigo, emplear tiempo en lo mismo que ya se había desarrollado antes, mayor esfuerzo, e incluso, que en ocasiones pudieran cometerse los mismos errores incesablemente. Algo que ha aprendido el hombre desde tiempos muy antiguos, es a reutilizar el conocimiento existente para sus cada vez más ambiciosas empresas. En efecto, al reutilizar segmentos de experiencias, ideas y artefactos, no solo se asegura no cometer las faltas del pasado, sino que se logra construir cosas más grandes y maravillosas, con bases firmes y calidad incomparable. Era cuestión solo de aplicar estas técnicas en el mundo del software.

En el diseño de cualquier sistema de software con cierta complejidad, uno de los aspectos fundamentales es su estructura o arquitectura, la cual está representada por un conjunto de elementos computacionales o componentes, y una serie de conexiones o interacciones entre ellos. Aplicar técnicas y métodos en esta fase, es un campo al que se le dedica cada vez mayor interés. No sólo se ha venido estableciendo una colección de métodos, técnicas, esquemas y expresiones para describir la arquitectura de un sistema de software, sino que se tiende al desarrollo de entornos que permitan la reutilización efectiva, tanto de los componentes de un sistema, como de la propia arquitectura o configuración del mismo.

Cuba apuesta porque el sustento de su economía esté basado en las producciones intelectuales desde hace algunos años, y el campo de la informática, que está dando grandes pasos de avance en la actualidad, puede aportar mucho en los ingresos del país por conceptos de producción de software. La Universidad de las Ciencias Informáticas (UCI) juega un papel importante en el desarrollo de la Industria Cubana del Software, e intenta convertirse en líder nacional en este campo, como universidad de nuevo tipo, primera de la Batalla de Ideas que lleva a cabo el pueblo cubano y con un novedoso modelo de formación que combina el estudio, la producción y la investigación. Ha demostrado en su poco tiempo de creada y sin graduar aún ningún ingeniero, que esta rama puede aportar muchos ingresos al país.

La UCI desde sus inicios comenzó a insertarse en la industria del software con mucha fuerza y en solo cuatro años ha ganado en seriedad, organización y ha alcanzado prestigio tanto a nivel nacional como en el ámbito internacional, muestra de ello son los disímiles proyectos de exportación en los que está trabajando. Pero sucede, que cuando se firma un nuevo contrato y se emprende el desarrollo del proyecto, se comienza desde cero en la construcción del producto, reprogramando componentes que pueden haberse desarrollado anteriormente por otros equipos de trabajo, invirtiéndose tiempo y recursos adicionales que aumentan los costos del proyecto. Una manera de revertir esta situación, puede ser la implantación en la Universidad de una biblioteca de componentes reutilizables, de forma tal que la comunidad universitaria, acceda a repositorios que contengan diferentes elementos desarrollados en un proyecto determinado, que pueden ser utilizados en otros.

Teniendo en cuenta todo lo descrito anteriormente el **Problema Científico** de la investigación es el siguiente:

¿Cómo gestionar los componentes de software para que puedan ser reutilizables, en beneficio del proceso productivo-investigativo, en la Universidad de las Ciencias Informáticas?

Del problema científico se puede definir que el **Objeto de Estudio** de esta tesis, es la reutilización de componentes de software.

Se plantea como **Objetivo General** de la investigación:

Proponer soluciones para la implantación de una biblioteca de componentes reutilizables, que responda a las necesidades de reusabilidad de la naciente industria de software en la UCI.

Los **Objetivos Específicos** son los siguientes:

- Analizar diferentes herramientas que se usan para gestionar componentes reutilizables.
- Hacer un análisis de las características de la producción de software en la UCI, para definir qué características debe tener una herramienta de componentes reutilizables.
- Definir las funciones que debe tener una biblioteca de componentes reutilizables que se ajuste a la industria de software en la UCI.
- Definir qué herramienta de las analizadas se puede adaptar al modelo de producción de software en la UCI.
- Hacer la propuesta de soluciones para la implantación de una biblioteca de componentes reutilizables en la UCI.

Del objeto de estudio analizado, se puede definir que el **Campo de acción** es**:** la reutilización de componentes de software en la Universidad de las Ciencias Informáticas.

Se plantean como guía de la investigación a desarrollar las siguientes **Preguntas Científicas**:

- ¿Es posible la reutilización de componentes de software en la UCI?
- ¿Cuáles son las características de una biblioteca de componentes reutilizables?
- ¿Cuáles son algunas herramientas que se usan en la reutilización?
- ¿Cómo sería una biblioteca de componentes reutilizables adaptada al modelo UCI?
- ¿Cuáles serían las posibles soluciones para la implantación de una biblioteca de componentes reutilizables en la UCI?

**Métodos teóricos:**

- **Analítico – sintético:** este método sirvió para distinguir los elementos relacionados con las bibliotecas de componentes de software reutilizables y proceder a revisar ordenadamente cada uno de ellos por separado, para posteriormente procesar toda la información, poder sintetizar y diferenciar cada una de ellas, además de poder extraer los elementos más importantes teniendo en cuenta el objeto de estudio. Estas operaciones no son independientes, pues el análisis de un objeto se realiza a partir de la relación que existe entre los elementos que lo conforman y a su vez, la síntesis se produce sobre la base de los resultados previos del análisis.
- **Histórico – lógico:** como está vinculado al conocimiento de las distintas etapas de los objetos en su sucesión cronológica, posibilitó conocer los antecedentes y tendencias de la reutilización de componentes de software en el mundo, en Cuba y más específicamente en la UCI. Mediante este método, se analiza la trayectoria concreta de la teoría, su condicionamiento a los diferentes períodos de la historia y al basarse en el desarrollo histórico, pone de manifiesto la lógica interna de su desarrollo, su teoría y encuentra el conocimiento más profundo de esta, su esencia.
- **Sistémico:** mediante este método se modeló el objeto a través de la determinación de sus componentes, así como las relaciones que existan entre ellos, las que determinan no solo la estructura del objeto sino también su dinámica.

**Método empírico:**

- **Entrevista:** se realizaron entrevistas a varios especialistas dentro y fuera de la UCI, con el fin de conocer en qué estado se encuentra el tema de las bibliotecas de componentes de software reutilizables en el mundo, Cuba y la UCI, así como sus criterios sobre la implantación de una de ellas en la Universidad.

**Aporte teórico:** las variantes que se proponen constituyen una alternativa para dirigir la atención sobre la implantación de una biblioteca de componentes reutilizables, que responda a las necesidades de reusabilidad de la naciente industria de software en la UCI.

**Significación práctica:** está dada por la disponibilidad de herramientas metodológicas que posibilitan usar el Gforge como una solución inmediata pero no permanente en un Polo Productivo. El desarrollo de una biblioteca propia para la universidad propiciará una herramienta robusta que permitirá acortar los plazos de desarrollo de software, así como sus costos.

La tesis se encuentra estructurada en una **Introducción**, tres capítulos, **Conclusiones** y **Recomendaciones**.

En el **Capítulo 1** se hace un análisis de las principales tendencias mundiales de la reutilización de componentes de software, su importancia y una breve reseña histórica. También se aborda sobre el modelo de producción de la UCI y algunas características de herramientas de gestión de proyectos.

En el **Capítulo 2** se describen las características que tiene una biblioteca de reutilización de componentes de software en la UCI, sus objetivos, componentes y qué elementos se deben reutilizar.

En el **Capítulo 3** se describen las dos soluciones que se proponen, una existente en el mercado y otra desarrollada en la UCI que se ajuste al modelo de producción de la Universidad.

# Capítulo 1: Análisis de las principales tendencias mundiales en la reutilización de componentes de software.

## 1.1 Introducción.

En el capítulo se presenta una reseña sobre la reutilización, desde que se comenzó a hablar del tema por primera vez en la industria del software hasta la actualidad, así como una breve panorámica de la reutilización de componentes de software y las bibliotecas que contienen estos elementos en el mundo, en Cuba y en la UCI. También se mencionan las principales características y ventajas de algunas de las herramientas de gestión de proyectos más utilizadas. Se aborda además sobre el modelo de producción de la UCI, profundizando en el Subproceso de Gestión del Conocimiento de la Organización.

## 1.2 Breve reseña sobre la reutilización de componentes de software.

La idea de que el software fuera construido de componentes prefabricados, fue publicada inicialmente por Douglas McIlroy, matemático, ingeniero y programador, profesor adjunto del Computer Science en el Dartmouth College, New Hampshire, Estados Unidos, en la primera conferencia sobre desarrollo de software patrocinada por el Comité de Ciencia de la OTAN y celebrada en Garmisch, Alemania, en octubre de 1968, siendo esta una de las alternativas para superar la “crisis del software”[1] (1) (2). En ese momento se propuso la creación de fábricas (industria de componentes reutilizables) de elementos de software, análogas a las ya existentes, de componentes hardware y la adopción de una librería de componentes de código fuente y técnicas automatizadas, para su adaptación en diferentes grados de precisión y robustez. En esos años existía una gran demanda de estos productos informáticos sin que las empresas que se dedicaban a la producirlos pudieran hacerle frente. Como se evidencia, era necesaria alguna salida para esta problemática. (3)

A partir de esta propuesta, comienzan a surgir soluciones de reutilización para tareas específicas en diferentes áreas de la computación como son: entrada/salida, computación numérica y el procesamiento y almacenamiento de texto en línea. La idea de Douglas McIlroy fue clave para el concepto de factoría de software. A pesar de que el concepto de reutilización como tal no estaba aún definido dentro del mencionado proceso, esta experiencia fue el punto de partida para las futuras factorías de software, especialmente en Japón, donde el proceso de reutilización fue definido.

Para hablar con más profundidad de los aspectos relacionados con términos tales como: reutilizar, componentes, reutilización de software, bibliotecas, bibliotecas de componentes reutilizables y repositorios, se deben definir estos conceptos.

¿Qué es reutilizar? *Es utilizar algo, bien con la función que desempeñaba anteriormente o con otros fines.*

¿Qué es un componente? *Algo que compone o entra en la composición de un todo.* (4)

¿Qué es una biblioteca? *Es una institución cuya finalidad consiste en la adquisición, conservación, estudio y exposición de libros y documentos. Local donde se tiene considerable número de libros ordenados para la lectura. Colección de libros o tratados análogos o semejantes entre sí, ya por las materias de que tratan, ya por la época y nación o autores a que pertenecen* (4)*. Es el centro local de información, brindando toda clase de conocimiento e información disponible a sus usuarios.* (5)

¿Qué es una biblioteca virtual? *Es una biblioteca en que una proporción significante de los recursos de información se encuentran disponibles en el formato digital (pdf, doc, etc.), accesible por medio de las computadoras. El volumen digital puede sostenerse localmente o puede accederse remotamente vía las redes de la computadora. Una biblioteca digital es un sistema de tratamiento técnico, acceso y transferencia de información digital, estructurado alrededor del ciclo de vida de una colección de documentos digitales, sobre los cuales se ofrecen servicios interactivos de valor añadido para el usuario final.* (6)

Algunos de los conceptos que se utilizan cuando se define reutilización de software son:

*"Reutilización de software es el proceso de crear sistemas de software a partir de software existente, en lugar de desarrollarlo desde el comienzo."* (7)

*“Reutilización es la capacidad de los elementos de software de servir para la construcción de muchas aplicaciones diferentes.”* (8)

*“La reutilización de software es el proceso de implementar o actualizar sistemas de software usando activos de software existentes.”* (9)

*“La reutilización es un enfoque de desarrollo de software que trata de maximizar el uso recurrente de componentes de software existentes."* (10)

¿Qué es entonces un repositorio? *Es un lugar donde se guarda algo.* (4)

Teniendo en cuenta el concepto de lo que es una biblioteca tradicional, una virtual y los anteriormente expuestos, se puede decir que una biblioteca de componentes de software reutilizables, son repositorios donde se guardan o conservan documentos, ideas, pedazos de código, librerías de software y todos aquellos componentes que participan en el desarrollo de un software, que puedan ser reutilizados, transformados o consultados.

El proceso de reutilización se fundamenta en la recuperación de activos en un repositorio y su posterior adaptación para el proyecto donde se desean aplicar. Por lo que se debe partir de un repositorio que permita identificar sus componentes, de cara a poder evaluarlos y seleccionar el candidato ideal a formar parte del nuevo proyecto.

La reutilización juega un papel clave en varios temas como son: productividad, capacidad de mantenimiento, portabilidad y calidad. Por ello, la reutilización debe aplicarse a cada etapa del ciclo de vida (levantamiento de requisitos, análisis, diseño, construcción, pruebas y mantenimiento). De lo contrario, no se recogerán todos sus beneficios potenciales.

A nivel individual o de pequeños grupos de desarrollo, este tema no es en realidad nada nuevo y desde siempre los programadores han utilizado fragmentos de código ya existentes para ahorrarse trabajo a la hora de programar. En otros casos se han utilizado bibliotecas de rutinas o de clases, que han dado buenos resultados, pero sin lograr los objetivos de la reutilización, o sea la disminución sustancial de los costos de desarrollo y mantenimiento, así como la mejora de los tiempos de desarrollo.

En los años 90 resultaba imposible poder representar artefactos de software en un repositorio por su propio contenido, lo cual impedía que cualquiera pudiera buscarlos sin conocimiento previo de su existencia, un poco por falta de cultura en la reutilización y otro poco, por la falta de organización en el tema que existía.

Por este motivo, para reutilizar hubo que realizar el proceso inverso: primero parar la organización para ver qué se podía reutilizar, después hacerlo reutilizable, lo siguiente era obligar a que todos conocieran lo que existe para reutilizar en la organización, y luego intentar “obligar” a que se reutilizara lo existente, por lo que tendrían que saber operar con ellos de antemano.

### 1.2.1 ¿Basta con reutilizar código fuente?

Hasta hace poco tiempo, se había enseñado que escribir código directamente a partir de un documento que describía vagamente las necesidades de usuario, no era una buena idea. Se sabía que el comienzo tenía que ser una correcta captura de requisitos, identificar todos los

interesados en el proyecto (stakeholders), modelar el negocio donde va a residir la futura aplicación, realizar diagramas de análisis, luego de diseño, etc. Sin embargo, ¿se llevaban a cabo todas estas tareas? La respuesta a esta pregunta resultaba ser un rotundo no.

Con esto en mente, ¿cómo era la reutilización de software que se aplicaba hace unos años? Claramente pasaba por reutilizar código fuente. Sin embargo, su reutilización es realmente problemática debido a que:

- Alguien que quiera reutilizar código en antiguas plataformas de desarrollo (COBOL, FORTRAN, LISP, C, etc.) estaría perdido en los tiempos actuales.
- Los que apostaron por reutilización basada en componentes COM o CORBA, se encuentran hoy día con que la tecnología actual se orienta hacia los Web Services.

La reutilización de componentes de software años atrás; no solamente a nivel de código fuente, podía traer grandes beneficios a la empresa que la usara. Estas buenas prácticas asociadas a la reutilización permitían:

- Una reducción de los costos de desarrollo.
- Un aumento de la calidad de los productos.
- Un aumento de la productividad, mediante la mejora de los tiempos en los que se desarrollaban los nuevos proyectos informáticos.
- Mejoras en las actividades de mantenimiento y soporte de aplicación.
- Mejoras en las actividades de control y planificación por la reducción de desviaciones en los desarrollos.

Pero no todo era bueno, también esta práctica traía consigo algunos inconvenientes:

- Las técnicas tradicionales implicaban una alta inversión inicial.
- Las nuevas herramientas y cambios en el proceso productivo avivaban el temor al cambio.
- Se hacía necesario formar personal.
- Existía dificultad para institucionalizar los procesos.

La existencia de estos inconvenientes, unido con el hecho de que la tecnología de recuperación de los componentes no estaba preparada para solucionar la reutilización, hicieron que la adopción de la reusabilidad en las organizaciones productivas, haya sido moderada durante este tiempo.

## 1.3 La reutilización de componentes de software en el mundo.

Si bien es cierto que hace varios años se hablaba de manera incipiente de la reutilización de componentes de software, es una realidad que en la actualidad existe una fuerte tendencia hacia el tema. Se reutiliza tanto código fuente, como el resto de los artefactos que se obtienen del proceso de desarrollo como son el diseño, especificación de requisitos, manuales de usuario, planes de prueba, procedimientos de instalación, etc. A todos estos artefactos se les llama comúnmente activos esenciales o core assets.

Según el estudio realizado por Cecilia Bastarrica del Departamento de Ciencias de la Computación de la Universidad de Chile, sobre la *"Arquitectura Base en una Línea de Productos de Software"*, la generalidad es que las empresas desarrolladoras de software se especifican en tipos particulares de aplicaciones y resulta esencial cómo de un desarrollo al siguiente, existen muchas partes que son o podrán ser en un momento determinado potencialmente reutilizadas, a las cuales se les llama activos. No obstante, la real magnitud de esta reutilización, depende en gran medida de la generalidad y anticipación del cambio que se haya utilizado en el desarrollo de los activos. (11) En este sentido desarrollar activos congruentes, generales, portables, robustos, documentados y probados, es claramente más caro, que desarrollarlos específicamente para una aplicación determinada. Sin embargo, si estos activos son reutilizados un número de veces suficiente, los costos se ven compensados. Se gana además en otros aspectos dentro de los que se puede mencionar: la confiabilidad en la calidad del nuevo producto elaborado, debido a que se integran activos ya probados; la planificación y la administración del desarrollo se tornan más fácil, ya que existen menos partes a desarrollar y se gana también en oportunidad de comercialización, pues desarrollos grandes y complejos pueden hacerse en un plazo de tiempo mucho menor. Por otra parte los clientes ganan en familiaridad con el nuevo software y la curva de aprendizaje se hace más corta, sobre todo si se reutilizan manuales de usuario, diseño de interfaces y mecanismos de interacción.

Un nuevo paradigma en el desarrollo de software son las Líneas de Productos de Software (LPS), que captura esta idea de la reutilización de activos de software a gran escala. La premisa esencial resulta ser, que no hay reutilización efectiva si la misma no se ha planificado. Para desarrollar líneas de productos de software se requiere una estructura organizacional particular, en la que personas con diferentes roles se encarguen de desarrollar los activos (ingeniería de dominio), integrar los productos (ingeniería del producto), o tomar decisiones estratégicas (administración) dentro de la empresa, todo esto de una forma coordinada.

Una visión muy similar a la expuesta anteriormente, es la de Jonás A. Montilva, miembro de la IEEE(Institute of Electrical and Electronics Engineers) y profesor de la Universidad de los Andes, quien en su investigación *"Desarrollo de Software Basado en Líneas de Producto de Software"*, explica que la idea básica de una LPS es el ensamblaje de partes de software previamente elaboradas, inspirada en los procesos de producción de sistemas físicos, la cual está fundamentada en la reutilización de software y que asume la existencia de una industria por partes.

Jonás se basa en las siguientes definiciones de LPS:

- *"... es un conjunto de sistemas de software que comparten un conjunto común y gestionado de aspectos que satisfacen las necesidades específicas de un segmento de mercado o misión y que son desarrollados a partir de un conjunto común de activos fundamentales [de software] de una manera preescrita."* (12)
- *"...consiste de una familia de sistemas de software que tienen una funcionalidad común y alguna funcionalidad variable."* (13)
- *"...se refieren a técnicas de ingeniería para crear un portafolio de sistemas de software similares, a partir de un conjunto compartido de activos de software, usando un medio común de producción."*
    - *La funcionalidad común descansa en el uso recurrente de un conjunto común de activos reutilizables (requisitos, diseños, componentes, servicios web, etc.).*
    - *Los activos son reutilizados por todos los miembros de la familia.* (14)

Teniendo en cuenta estos conceptos, define el modelo básico de una LPS en la siguiente figura.

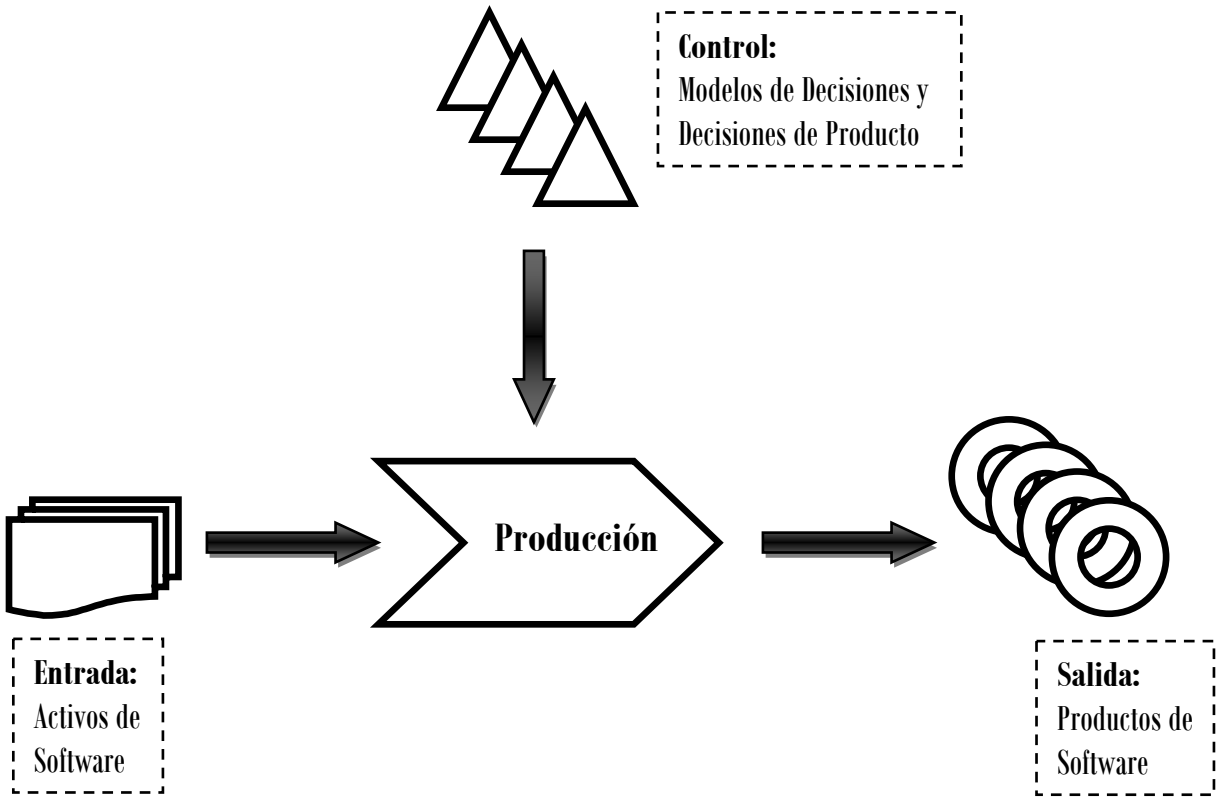



**Figura 1. Modelo Básico de una Línea de Productos de Software.**

En el caso de la UCI, este concepto que se describe como Líneas de Productos de Software, se asocia a los Polos Productivos, que contienen los procesos de Administración de los Proyectos Específicos y Desarrollo, Mantenimiento y Soporte, realiza sus actividades según los elementos de funcionamiento transmitidos por la Dirección General de Producción, entrega información a esta y los productos que son generados, así como toda la documentación concebida en los proyectos, para mantener la base de conocimiento de la organización. Su propósito es establecer y llevar a cabo sistemáticamente las actividades que posibilitan cumplir con los objetivos de un proyecto en tiempo y costos esperados, así como la realización sistemática de las actividades de análisis, diseño, construcción, integración y pruebas de productos de software nuevos o modificados, cumpliendo con los requerimientos especificados. En la definición de los polos se agrupan los productos, se trazan las líneas y prioridades claras y se alinean los proyectos. Para completar, desde la visión de la UCI, el concepto de las líneas de productos, que son los polos productivos como se ha explicado, se hace necesario llevar a la práctica la propuesta del presente trabajo de diploma, de implantar una biblioteca de

componentes de software reutilizables en la Universidad, de forma tal, que los productos sean desarrollados a partir de un conjunto de activos, que se utilizaron ya en un sistema previo, y que están disponibles en la biblioteca para el que lo necesite.

Para comprender mejor todo lo que se integra en un Polo Productivo, según el concepto de la UCI, se muestra la Figura 2.

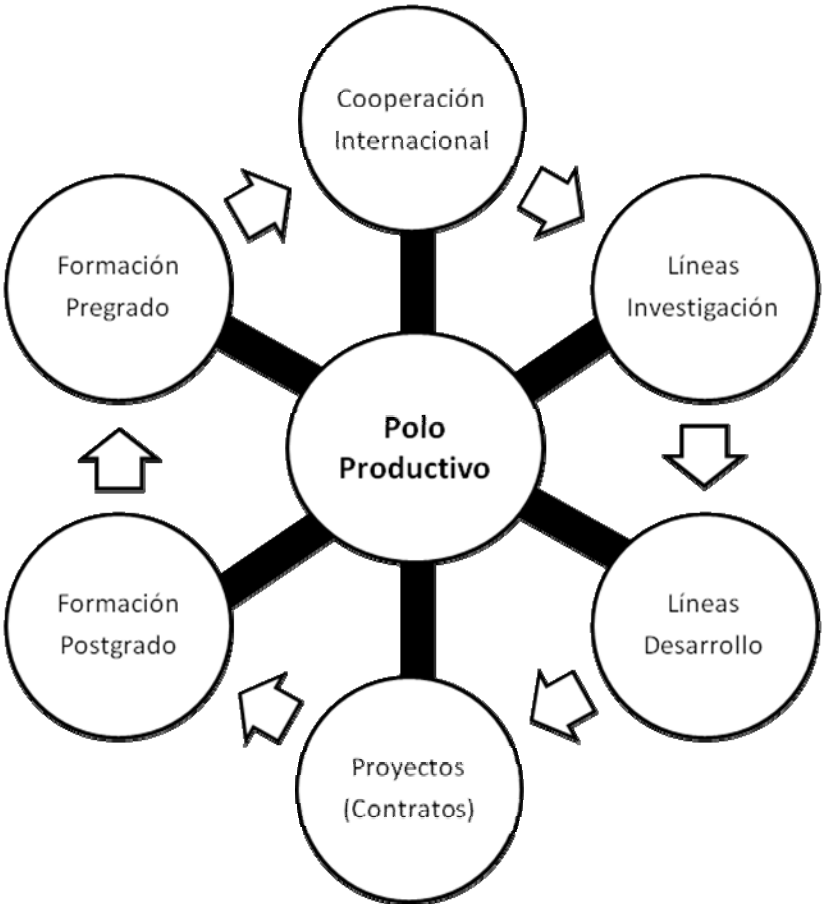



**Figura 2. Polos Productivos en la UCI.**

En su estudio, Jonás A. Montilva resume en el siguiente esquema la evolución de la reutilización de software, poniendo en la cima del mismo el Desarrollo de Software Basado en Líneas de Producto, cumbre a la que debe llegar la UCI con un concepto más práctico de Polos Productivos.

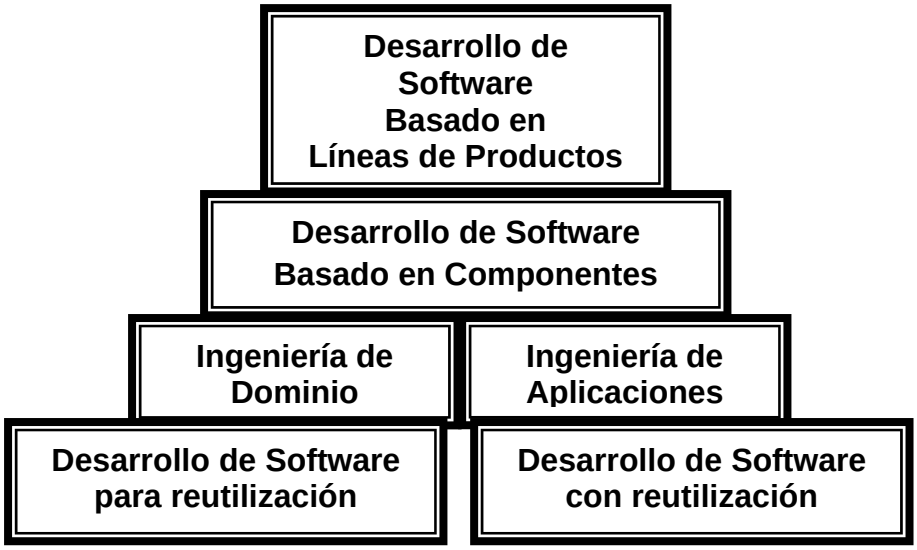



**Figura 3. Evolución de la reutilización de software.**

La reutilización de activos de software posee varias características, dentro de las que se pueden señalar:

- **Es estratégica:** consolida lo común entre los productos, maneja estratégicamente la variación entre ellos y elimina la duplicación de esfuerzos de ingeniería.
- **Es predictiva:** la reutilización de activos se da en uno o más productos sobre una línea bien definida y en lugar de reutilizar componentes de manera oportunista, se reutilizan arquitecturas de software.
- **Es gestionada:** es sistemática, planificada, institucionalizada y mejorada.

Los activos de software reutilizables son productos de software diseñados explícitamente para ser utilizados múltiples veces en el desarrollo de diferentes sistemas o aplicaciones. En este sentido, un componente de software puede ser:

- Una especificación de requisitos.
- Un modelo de negocio.
- Una especificación de diseño.
- Un algoritmo.
- Un patrón de diseño.

- Una arquitectura de dominio.
- Un esquema de base de datos.
- Una especificación de prueba.
- La documentación de un sistema.
- Un plan.

En una LPS, un componente reutilizable es:

*"Una pieza [de software] funcional que es liberada independientemente [de otras] y que proporciona acceso a sus servicios a través de sus interfaces."* (15)

Para el desarrollo de software en la UCI a través de los Polos Productivos, un componente o activo reutilizable, es cualquiera de los artefactos que se generen durante alguna de las etapas del ciclo de vida de un software, un módulo generado a partir de estos artefactos, o un producto ya elaborado que está formado por dichos módulos.

Se requiere almacenar los activos en repositorios, los cuales son una base de datos especializada que almacena componentes de software y facilita la recuperación y el mantenimiento de los mismos. Tienen como objetivo asegurar la disponibilidad de componentes para apoyar el desarrollo de un producto de software. Estos repositorios mantienen información relevante de cada activo usado, como son la especificación técnica, historia o registro de uso, clasificación y documentación.

La creciente industria del software ha buscado formas de disminuir eficaz y rápidamente, los tiempos de entrega y los costos de desarrollo de sus productos, al mismo tiempo que pretenden mejorar la calidad de estos. La reutilización es considerada como uno de los mecanismos más realistas de los planteados hasta el momento para lograr los objetivos mencionados anteriormente.

Es importante mencionar que hoy existen en el mercado varias herramientas que apoyan el concepto de reutilización. Dentro de ellas podemos mencionar las de tipo CASE[2], las cuales buscan automatizar los aspectos claves de todo el proceso de desarrollo de un sistema, desde su comienzo hasta el final. Estas herramientas tienen algunos inconvenientes asociados como son: los elevados costos tanto para adquirirlos como para el entrenamiento del personal en su uso, la falta de adaptación de la herramienta a la arquitectura de la información y a las metodologías de desarrollo utilizadas por la organización.

Grandes compañías de software como ORACLE, IBM, MICROSOFT, entre otras, cuentan con motores propietarios que buscan en cierto modo la reutilización de componentes, suministrando un conjunto de herramientas para el desarrollo de sistemas tales como el acceso a datos, la seguridad, la administración de procesos y documentación. El problema que existe con estas herramientas es el alto costo de licenciamiento, lo que le deja sólo a grandes empresas la posibilidad de adquirirlos.

Muchos sistemas empresariales necesitan ser conectados de alguna forma con otros, tanto dentro de una misma compañía como a lo largo de Internet y en muchos casos las nuevas demandas hacen que la integración sea esencial. Es entonces, cuando el desarrollo de componentes se orienta a construir aplicaciones largas de software mediante la integración de activos ya existentes. Este enfoque se fundamenta porque ciertas piezas de software deben ser escritas cada vez, mientras que otras que son comunes pueden reutilizarse escribiéndose solo una vez.

Este tema del desarrollo de software basado en componentes ha sido investigado también por Julio Casal Terreros, Ingeniero de Sistemas Computacionales de la Universidad Católica de Santiago de Guayaquil. En uno de sus trabajos explica, que los sistemas de hoy en día son cada vez más complejos, deben ser construidos en tiempo record y cumplir con los estándares más altos de calidad. Entonces, para hacer frente a esto, se concibió y perfeccionó lo que se conoce como Ingeniería de Software Basada en Componentes (ISBC), la cual se centra en el diseño y construcción de sistemas computacionales que utilizan componentes de software reutilizables. Se trabaja bajo la filosofía de “comprar, no construir”, idea que se ha hecho común en casi todas las industrias desde hace bastante tiempo, pero que es relativamente nueva cuando de construcción de software se trata.

Con respecto a la afirmación anterior los autores consideran que la ISBC es acertada, pues se trabaja básicamente con la reutilización de componentes de software, pero no cree que la UCI deba trabajar en la filosofía “comprar, no construir” y sí en la de reutilizar los componentes que la Universidad, como industria de software, construya. Aunque es importante mencionar que si en un momento determinado, por requerimientos de un cliente específico, es necesario comprar algún componente ya elaborado, porque se necesite particularmente ese y no otro, se podrá hacer, lo que no debe ser la práctica habitual.

El paradigma de ensamblar componentes y escribir código para hacer que estos componentes funcionen, se conoce como Desarrollo de Software Basado en Componentes (DSBC). Su uso posee algunas ventajas:

- Se alcanza un mayor nivel de reutilización de software.
- Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de examinar el conjunto completo de componentes ensamblados.
- Cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema.
- Dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo.

Así mismo, el hecho de optar por comprar componentes de terceros en lugar de desarrollarlos, también tiene algunas ventajas:

- La adición de una pieza dada de funcionalidad tomará días en lugar de meses o años, por lo que los ciclos de desarrollo son más cortos.
- Usando correctamente esta estrategia, el retorno sobre la inversión (ROI)[3] puede ser más favorable que desarrollando los componentes la propia empresa u organización.
- Para usar un componente que contenga una pieza de funcionalidad, solo se necesita entender su naturaleza, no sus detalles internos. Así, una funcionalidad que sería impráctica de implementar en la empresa, se vuelve ahora completamente asequible.

El DSBC se convirtió en el pilar de la Revolución Industrial del Software y se proyecta en la actualidad en diversas y nuevas formas de construir software de calidad, con los costos más bajos del mercado y en tiempos que antes eran una utopía. Empresas como Microsoft incorporaron el potencial de esta metodología hace años y hoy ofrece iniciativas novedosas y herramientas que buscan llevar al proceso de fabricación de software hacia el lugar privilegiado en el que debió colocarse desde un principio. Clemens Szyperski, arquitecto experimentado de Microsoft, ha expresado lo siguiente:

*"El construir software ensamblando componentes conlleva grandes promesas para la ingeniería de software de la nueva generación. Yo iría más allá y aseguraría que no se puede hablar de ingeniería antes de haber dominado este paso … Si no, se está hablando de hacer las cosas a mano, algo similar a la manufactura temprana previa a la Revolución Industrial. Así que, es claro que deberíamos empezar a construir software basado en componentes."*

Julio Casal presenta más adelante en su trabajo, las Fábricas de Software como el nuevo paradigma. Estas fábricas son una iniciativa propuesta por Microsoft, que plasma la necesidad y proporciona los medios para hacer la transición desde “el hacer a mano”, hacia la fabricación o manufactura. Se define una Fábrica de Software como un ambiente de desarrollo configurado para soportar el desarrollo acelerado de un tipo específico de aplicación. No son más que el siguiente paso lógico en la evolución continua de los métodos y prácticas de desarrollo de software; sin embargo, prometen cambiar el carácter de la industria de software introduciendo patrones de industrialización. (16)

Un tema recurrente que se puede observar en la creación de un tipo de herramientas que permitan la automatización del proceso de desarrollo del software, es el hecho de pensar en modelos, más que solamente en objetos.

Sin embargo resulta, que aún cuando los modelos juegan un rol importante, no lo son todo. Para llegar a niveles más altos de productividad, se necesita la habilidad de configurar, adaptar y ensamblar rápidamente componentes desarrollados independientemente, autodescriptivos e autónomos de ubicación, para producir así familias de sistemas similares, pero diferentes. Entonces para lograrlo, será necesaria la creación de patrones con dominio específico, frameworks y herramientas que se alineen tanto con la arquitectura del producto, como con el ciclo de vida del producto. Más aún, se deben considerar los procesos usados para analizar requerimientos, desarrollar software y ponerlo en producción. Se precisa disponer de las mejores prácticas, contenido reutilizable y distintos tipos de patrones. La aplicación de todo esto en conjunto, es lo que se conoce como Fábrica de Software.

Los autores consideran que los criterios que se exponen sobre las fábricas de software son acertados, pues para producir con mucha mayor rapidez y calidad, es necesario tener todo lo que explican estos conceptos. Es un nivel de madurez por el que la UCI debe apostar.

Estas fábricas son además una línea de producto que configura herramientas de desarrollo extensibles con contenido empaquetado y guías, las cuales son cuidadosamente diseñadas para construir tipos específicos de aplicaciones. Las tres ideas en las que se basa fundamentalmente son:

- **Un esquema de fabricación:** la analogía de esto es una receta, en la que se listan ingredientes como proyectos, código fuente, directorios, archivos SQL y archivos de configuración, y explica cómo deberían ser combinados para crear el producto. Detalla qué Lenguajes de Dominio Específico (DSL) pueden ser usados y describe cómo los modelos

basados en ellos pueden transformarse en código y otros artefactos, o en otros modelos. Describe también la arquitectura de la línea de producto, y las relaciones clave entre componentes y frameworks que la componen.

- **Una plantilla de fábrica de software:** provee los patrones, guías, plantillas, frameworks, ejemplos, herramientas personalizadas como la utilizadas para la edición visual de DSLs, scripts, XSDs[4], hojas de estilos, y otros ingredientes para construir el producto.
- **Un ambiente de desarrollo extensible:** cuando se configura con una plantilla de fábrica de software, se convierte en una fábrica de software para la familia de productos.

Se asegura que estas fábricas son posibles hoy en día y que representan el intento de aprender de otras industrias, que enfrentan problemas similares y que aplican además patrones específicos de automatización a tareas de desarrollo que existen manualmente. Estas fábricas vuelven más rápida, barata y fácil la construcción de aplicaciones, concretando de esta forma la visión de la industrialización del software moderno.

El DSBC desde el principio validó la idea de construir software en corto tiempo y con la misma calidad que la mayoría de las industrias actuales. Se evidencian los sorprendentes avances logrados en la comprensión de la forma correcta de reutilizar el software, el conocimiento existente y el hecho de que es solo el inicio.

Un artículo publicado en Scientific Electronic Library Online, SciELO Perú, llamado *"Reutilización de software y su impacto en el costo del sistema"*, plantea que el proceso de industrialización del software, exige que los ingenieros y técnicos diseñen nuevas alternativas, para incrementar la productividad de los desarrolladores y analistas, en el desarrollo de sistemas de software.

En este contexto, la publicación valora la productividad de un desarrollador, realizando una comparación entre un sistema desarrollado reutilizando software y otro en el que no se reutilizan los componentes. Por una parte, los usuarios demandan mayor complejidad y funcionalidad de los sistemas de software, mientras que por otra, los desarrolladores tienen que manejar la complejidad propia de la lógica de negocios y de la herramienta de programación, teniendo en cuenta lo limitados que puedan ser los recursos económicos y de tiempo. En este escenario, una de las mejores opciones para disminuir los plazos de entrega, es sin dudas, la reutilización de software.

Las conclusiones a las que se arriba en este material, después de los estudios y cálculos realizados, son la reducción del 43,89% en el plazo de entrega, si se utiliza software a través

de componentes, y un ahorro del 78,15% en los costos de mano de obra. La reutilización de componentes se constituye como una forma de mejorar la productividad en el desarrollo de sistemas de software.

Las compañías Fujitsu Limited y Software AG, después de aliarse estratégicamente, anunciaron la liberación al mercado de una oferta conjunta para ayudar a las organizaciones en la gestión de componentes de integración bajo el paradigma de las Arquitecturas Orientadas a Servicios (SOA)[5]. CentraSite, es un repositorio abierto de generación SOA (Service-Oriented Architecture), que promueve una mayor colaboración entre los usuarios de negocios y los de las tecnologías de información, mediante la unión de meta datos de productos de integración orientados a servicios.

El Dr. Jose Luis Fernández Sánchez, Profesor Titular de la Cátedra de Proyectos de la Escuela Técnica Superior de Ingenieros Industriales de Madrid, en su estudio *"Reusabilidad y Desarrollo Orientado a Objetos"*, publicado en L´atigoo, página web de la Asociación Técnicos de Informáticos de España, expone que las experiencias de la industria indican que existen 3 factores esenciales que llevan al éxito en la reutilización, y que no son tecnológicos sino organizativos (17). Los factores mencionados son:

1. **Soporte de la dirección.** Las decisiones en la utilización de procesos de desarrollo de software comunes, herramientas, y lenguajes, así como las inversiones en formación, construcción de activos reutilizables y adaptación de la organización son inviables sin un apoyo explícito de la alta dirección. Esto se traduce en la UCI, a que exista una política desde la Dirección General de la Infraestructura Productiva de la Universidad, que es la categoría que enmarca según la propuesta de modelo para la producción de software en la UCI, el proceso de Gestión de Recursos, dentro del cual está el Subproceso de Gestión del Conocimiento de la Organización, en el que la reutilización componentes de software se hace evidente. Ya la Universidad comenzó a trabajar en este sentido y para todos los proyectos debe cumplirse, que las tecnologías o herramientas a utilizar para el desarrollo de software, sean presentadas y ratificadas en el Consejo Técnico de Producción[6] y para ello deben llevar un dictamen de aprobación que avale su uso.
2. **Cambios en la organización.** La reutilización sistemática conlleva la división del personal de desarrollo en dos grupos principales con funciones distintas. El grupo de Ingeniería de Dominio, que construye los activos reutilizables, sean estos modelos del dominio, bibliotecas, arquitecturas o generadores de aplicaciones, y el grupo de Ingeniería de Aplicación, que construye aplicaciones con los activos reutilizables existentes en la

organización. En el caso particular de la Universidad de las Ciencias Informáticas se cree que esta división no es el modelo que debe proponerse en los momentos actuales, pues conllevaría a crear un grupo que se dedique solamente a desarrollar elementos, que después puedan ser reutilizados, y otros que serian los proyectos, que los utilicen para construir sistemas. En la UCI se organizan grupos de proyectos según los compromisos de trabajo, los cuales pertenecen a las facultades, se agrupan en Polos Productivos, y todos realizan aportes a la biblioteca de componentes de software reutilizables, según los resultados que van obteniendo durante todo el ciclo de vida de los mismos, y a la vez, pueden reutilizar los que están publicados en dicha biblioteca y que se adapten al sistema que están desarrollando.

3. **Cambios en el personal.** La formación y los incentivos son útiles para cambiar prácticas habituales en el personal y disminuir la desconfianza en la utilización de código desarrollado por otros como parte del suyo propio. Cuando una organización se decide a adoptar la reutilización, debe ir de las soluciones tecnológicas más simples, como puede ser las bibliotecas de clases, hasta llegar a una mayor complejidad, como son las arquitecturas de referencia. Así la empresa sigue una evolución in crescendo. Se concuerda con la idea de que por la falta de cultura que existe cuando de reutilización se trata, es necesario desde la formación de los estudiantes de la UCI, introducir la importancia, las ventajas, la esencia de la reutilización de componentes de software y en este sentido se hacen propuestas en el siguiente capítulo.

En el Capítulo 2 se presentan las tres acciones organizativas que los autores consideran que deben llevarse a la práctica en la UCI, para lograr el éxito de la implantación de una biblioteca de componentes de software reutilizables.

El autor referencia en su trabajo que Griss propone un modelo de evolución en la adopción de la reutilización por una organización, que conlleva cinco etapas en la evolución de la misma:

- **Corta-Pega:** en esta primera fase de la evolución, la organización trata de disminuir los tiempos de desarrollo mediante la clonación de trozos de código. Aunque es una solución a primera vista, los problemas aparecen en el mantenimiento, pues a la hora de realizar cambios hay que ser consciente de todos los sitios en el código donde estos se aplican. Es un problema típico de la gestión de configuración.
- **Caja Negra:** en la reutilización de caja negra se identifican componentes software de uso común en la organización. Se garantiza que existe un original único de ellas. La utilización

de bibliotecas y taxonomías de clasificación puede ayudar en esta fase de la evolución de la organización.

- **Activos Reutilizables:** en esta etapa la organización considera otros activos reutilizables distintos del código. Entre estos destacan principalmente los procedimientos de prueba, las especificaciones, ficheros de ayuda, herramientas y otros. Con esta aproximación se incrementa la reutilización en las diversas fases del ciclo de vida del software, no solo en la fase de implementación como ocurría especialmente en la Caja Negra.
- **Arquitecturas:** en esta fase se generan componentes reutilizables, que podrían ser los anteriores, y una arquitectura software que los aglutina, permitiendo desarrollar aplicaciones en un dominio específico de negocio, sea este financiero, control de tráfico aéreo, gestión de red u otros.
- **Reutilización sistemática:** se basa en la estandarización de los activos reutilizables y los procesos para producirlos, la creación de una infraestructura para su producción y los mecanismos organizativos adecuados para facilitar la reutilización de los mismos. La separación del personal en grupo de ingeniería de dominio, o responsable de crear y mantener los activos reutilizables y el grupo de ingeniería de aplicación, responsable de utilizarlos, es un aspecto fundamental en esta fase.

En el caso de la UCI, los autores consideran que la evolución de la reutilización de componentes de software debe empezar por una etapa similar al corta y pega que se describe por Griss, pero consideran en una segunda etapa como análoga, la de arquitecturas que se describe, donde se integran las dos anteriores (Caja Negra y Activos Reutilizables), lo cual genera un profundo estudio que debe realizarse para definir con claridad, cuáles son específicamente los activos a reutilizar en dependencia del tipo de software. En una fase superior, debe estar la reutilización sistemática, o sea, la creación de un grupo de ingeniería de dominio, cuya función es desarrollar componentes para que sean reutilizados por los proyectos de la Universidad. La Dirección de la Producción en la UCI apuesta por esta etapa como un nivel superior en la reutilización.

El tema de la reutilización de componentes de software hasta el momento se ha tratado poco en España, pero está surgiendo de manera progresiva como una nueva disciplina dentro de la Ingeniería del Software, con el fin de facilitar el desarrollo y construcción de aplicaciones de software mediante componentes de software reutilizables, por las ventajas que posee, las que han sido mencionadas en el desarrollo del presente capítulo. REUTI-ES es un foro destinado a

discutir y fomentar todos los aspectos de la reutilización de software a nivel general. Tiene como objetivo esencial brindar mayor visibilidad del tema de la reutilización tanto en España como en el resto del mundo, dando a conocer en qué consiste la reutilización de software en sus diferentes formas y cómo es posible aplicarla en múltiples ámbitos, tanto empresariales como científicos.

Los objetivos principales en sentido general son:

- Dar a conocer qué es la reutilización de software y sus diferentes formas.
- Establecer dónde y cuándo se puede aplicar.
- Conocer las técnicas y métodos principales que emplea.
- Discutir y fomentar procesos de reutilización de tipo sistemático, estableciendo las bases para el desarrollo de una metodología consistente.
- Relacionar a personas y organizaciones relacionadas con el tema, tanto a nivel particular como a través de proyectos.
- Integración de los procesos de reutilización dentro del desarrollo de aplicaciones.
- Proporcionar información y bibliografía.

Dentro de los temas que proponen para el intercambio y análisis podemos encontrar los siguientes:

- Análisis del Dominio e Ingeniería del Dominio.
- Reutilización de componentes.
- Reutilización de Conocimiento (análisis, diseños, especificaciones, arquitecturas).
- “Product Lines.”
- Validación y Verificación de componentes y procesos de reutilización.
- Reutilización sistemática.
- Métricas de reutilización.
- Repositorios.
- Megaprogramación.
- Reutilización y técnicas orientadas a objetos.

A criterio de los autores, la reutilización como nueva disciplina dentro de la ingeniería de software, es un paso para crear el hábito y la cultura en la reutilización, así como profundizar en su importancia en el proceso de producción de software. Con respecto a este tema se hace una propuesta en el Capítulo 2.

### 1.4 Herramientas para la gestión de proyectos colaborativos.

Desde que se comenzó a hablar del término “reutilización” en la industria del software, se han dado varios pasos de avance y se han desarrollado herramientas, con sus fortalezas y debilidades, pero que contribuyen con esta idea. Algunas de ellas contienen elementos ya integrados, que proporcionan utilidades adicionales, permitiendo así, un mejor desarrollo del software en una comunidad. Dos de las herramientas más conocidas y llamadas paradigmas de la reutilización, son el SharePoint y el Gforge.

SharePoint es un conjunto de varias tecnologías que facilitan de algún modo compartir información en la red y algunas de las características que posee son las siguientes:

- Obtener tecnología de búsqueda adaptada a la información empresarial.
- Satisfacer las necesidades básicas con una única tecnología de búsqueda para la intranet y los sitios de Internet.
- Ampliar fácil y directamente la búsqueda de los repositorios, así como los tipos de archivos normales.
- Compatibilidad del cumplimiento y protección de la propiedad intelectual, con los resultados de la búsqueda garantizados y seguros.
- Mejorar los resultados de la búsqueda mediante la indización y entrega más rápidas.
- Obtener flexibilidad con una estructura muy escalable.
- Responder rápidamente mediante una administración sólida.
- Mejorar las búsquedas mediante interfaces y aplicaciones conocidas, con resultados que se pueden procesar.
- Buscar información fácilmente con la sencilla, eficaz interfaz de usuario, así como la integración de Windows Desktop Search.
- Obtener una solución que crezca al ritmo de las necesidades empresariales.

En su conjunto esta herramienta es muy completa, pues brinda casi todos los servicios que pueden interesar para el desarrollo de software en equipo, como parte de una comunidad, pero tiene en contra el hecho de ser una herramienta de Microsoft. Significa entonces que no es de código abierto o libre, así como que no admite algunas tecnologías como PHP o JSP y teniendo en cuenta además, que Microsoft es muy dinámica sacando al mercado sus nuevos "inventos", es muy posible que deje de funcionar en un momento determinado.

Gforge es una herramienta basada en plataforma libre para el desarrollo de software en forma comunitaria, que permite organizar y administrar grandes cantidades de proyectos. Proporciona un conjunto integrado de herramientas que facilitan el trabajo en colaboración.

Algunas de las funciones que ofrece esta herramienta son:

- Un servidor Web por proyecto (Web Site).
- Control de versiones.
- Servidor Web para la coordinación del equipo de desarrollo.
- Foros de discusión.
- Tratamiento de incidencias.
- Peticiones de mejora y distribución de parches[7].
- Comunicación mediante listas de distribución de correo.
- Compartición de la documentación.
- Descargas de archivos.
- Gestión de la planificación de tareas.
- Distribución de noticias significativas para el proyecto.

Dentro de sus características más relevantes están:

- **Técnicas:** web, abierto, libre, escalable, robusto.
- **Funcionales:** gestión, comunicación, coordinación, centralización (agrupa fuentes de información y hace homogéneo su modo de acceso), control.
- **Utilidad de gestión de proyectos:** tareas, incidencias, listas de correo, discusiones, documentos y repositorio de código y software.

El uso del Gforge ofrece ventajas, pues por ejemplo, permite centralizar y homogeneizar la gestión de proyectos, es una página única, se consigue un aumento de la productividad, se tienen herramientas comunes para todo el entorno donde se utilice, es capaz de centralizar los recursos técnicos de un servidor (en lugar de tener que soportar múltiples máquinas por proyecto), y la ventaja esencial es que está basada en plataforma libre.

Según el Ing. Junior A. Altamiranda, miembro de la Red Nacional de Integración y Desarrollo de Software Libre de la República Bolivariana de Venezuela, en un estudio realizado en junio de 2006 sobre la plataforma Gforge, alrededor de esa fecha, se gestionaban aproximadamente 80 000 proyectos a través del Gforge y al menos 93 sitios web utilizaban el sistema. Se pueden mencionar dentro de los grupos o empresas más importantes que lo utilizan a:

- Desarrollo Colaborativo RINDE (Red Nacional de Integración y Desarrollo de Software Libre de la República Bolivariana de Venezuela).
- Fundacite – Mérida, Venezuela.
- Philips, Holanda.
- NASA Goddard Space Flight Center, Estados Unidos.
- NOAA (National Oceanic & Atmospheric Administation), Estados Unidos.
- National Science Digital Library, Argentina.
- DARPA (Defense Advanced Research Projects Agency), Estados Unidos.

### 1.5 Tendencias y tecnologías actuales relacionadas con las aplicaciones web.

Una aplicación web es un sistema informático que los usuarios utilizan accediendo a un servidor web a través de Internet o de una Intranet. Las aplicaciones web son populares debido a lo práctico que resulta el navegador web como cliente ligero y a la habilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software, en miles de potenciales clientes. Las aplicaciones web generan dinámicamente una serie de páginas en un formato estándar, soportado por navegadores web comunes como HTML o XHTML.

Las aplicaciones Web se clasifican en dos grupos atendiendo a la complejidad y el funcionamiento: las aplicaciones Web estáticas, que son mucho más sencillas y las aplicaciones Web dinámicas, las cuales son una página que depende de una base de datos que puede ser modificada online.

Por lo general, una aplicación Web está estructurada como una aplicación de tres-capas. En su forma más usual, el navegador es la primera capa, la intermedia es un motor usando alguna tecnología web dinámica (PHP, ASP, etc.) y una base de datos como la última capa. El navegador web manda peticiones a la capa media, que la entrega valiéndose de consultas y actualizaciones a la base de datos, generando de esta manera una interfaz de usuario.

### 1.5.1 Lenguajes de programación.

Entre los lenguajes de programación que más se destacan, dentro de los utilizados para las aplicaciones web, se encuentran:

- **Hipertext Preprocesor (PHP):** lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran librería de funciones y variada documentación. Posibilita la conexión a diferentes tipos de servidores de bases de datos; lo cual permite la creación de aplicaciones web muy robustas. Además posee la capacidad de ser ejecutado en la mayoría de los sistemas operativos.
- **Active Server Pages (ASP):** tecnología desarrollada por Microsoft para crear páginas web de contenido dinámico apoyándose en scripts ejecutados en el servidor. Básicamente una página ASP es una mezcla entre una página HTML y un programa que da como resultado una página HTML, la cual es enviada al cliente (navegador). Las principales ventajas que tiene ASP son su eficacia, flexibilidad, escalabilidad, disponibilidad, facilidad de uso y seguridad.

Para el desarrollo de la propuesta que se explica en el capítulo 3, se plantea como lenguaje programación PHP, debido a que es un lenguaje que se puede utilizar tanto en sistemas simples como complejos, es fácil y rápido, pero a la vez potente y muy amplio. Otras de las características que presenta son: que está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado, es multiplataforma, libre, interpretado, fácil de cambiar, es independiente de plataforma pues existe un módulo de PHP para casi cualquier servidor web, lo que permite que el sistema pueda ser compatible con el lenguaje y significa una ventaja importante, ya que admite portar el sitio desarrollado en PHP de un sistema a otro, sin prácticamente ningún trabajo y además cuenta con una gran comunidad. Posee también conexión con la mayoría de los gestores de base de datos que se utilizan en la actualidad como son: PostgreSQL, MySQL, Oracle y Microsoft SQL Server. Es un software de código abierto, por lo que se presenta como una

alternativa de fácil acceso para todos. Estas razones sustentan el hecho de que la UCI apuesta cada vez más por su uso en la producción.

### 1.5.2 Gestores de Base de Datos.

En el caso de los gestores de base de datos, los que más se utilizan en la actualidad son:

- **El Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos, PostgreSQL:** es uno de los gestores de bases de datos de código abierto más avanzado hoy en día. Ofrece control de concurrencia multi-versión, soportando casi toda la sintaxis SQL (incluyendo subconsultas, transacciones, y tipos y funciones definidas por el usuario).
- **El Sistema de Gestión de Base de Datos Relacional Oracle:** es considerado como uno de los sistemas de datos más completos por su soporte de transacciones, estabilidad, escalabilidad, además de ser multiplataforma. Sus últimas versiones han sido certificadas para poder trabajar bajo Linux, pero la principal deficiencia que posee es su alto costo.
- **El Sistema de Gestión de Base de Datos Relacional Microsoft SQL Server**: está basado en el lenguaje SQL, capaz de poner a disposición de muchos usuarios, grandes cantidades de datos de manera simultánea. Entre las ventajas que posee están la escalabilidad, estabilidad y seguridad, además soporta los procedimientos almacenados, incluye un potente entorno gráfico de administración y permite trabajar en modo cliente-servidor. Presenta como desventaja esencial que no es multiplataforma, ya que sólo está disponible en Sistemas Operativos de Microsoft.
- **El Sistema de Gestión de Base de Datos Relacional MySQL:** es un sistema de gestión de bases de datos, licenciado bajo la GNU GPL[8], pero empresas que quieran incorporarlo en productos privativos pueden comprar a la empresa una licencia que les permita ese uso. Su diseño multihilo[9] le permite soportar una gran carga la información de manera eficiente. El copyright del código está en poder del autor individual, MySQL es propiedad y está patrocinado por una empresa privada, la empresa sueca MySQL AB, que posee el copyright de la mayor parte del código. Este gestor de bases de datos es de los gestores más usados en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Posee como desventajas, que carece de soporte para las transacciones[10], rollback´s[11] y consultas, y que no es viable para su uso con grandes bases de datos, a las que se esté accediendo constantemente, debido a que no implementa una buena escalabilidad.

El sistema gestor de base de datos que se propone para el desarrollo de la propuesta presentada en el capítulo 3 es el PostgreSQL, debido a que posee gran escalabilidad, siendo

capaz de ajustarse al número de CPUs, a la cantidad de memoria que posee el sistema de forma óptima y soportando una mayor cantidad de peticiones simultáneas de manera correcta. También implementa el uso de transacciones, rollback´s y subconsultas, haciendo su funcionamiento mucho más eficaz y tiene la capacidad de comprobar la integridad referencial, así como también de almacenar procedimientos de la propia base de datos. En la UCI es el gestor de base de datos que se utiliza en sistemas mayores, más complejos y en los que requieran transacciones.

### 1.5.3 Servidores Web.

Un servidor web es un programa que implementa el protocolo HTTP (Hypertext Transfer Protocol), el cual está diseñado para los hipertextos, páginas web o páginas HTML (Hypertext Markup Language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música.

Dentro de los servidores web más utilizados en la actualidad se encuentran:

- **Servidor HTTP Apache:** es un servidor de código abierto, multiplataforma. Presenta además entre sus principales características mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido. La mayoría de las vulnerabilidades de la seguridad descubiertas y resueltas puede, en la mayoría de los casos, ser violada solamente por usuarios locales y no puede ser accionada remotamente. La arquitectura del servidor es modular y permite que el servidor de base puede ser extendido con la inclusión de módulos externos. Tiene capacidad para servir páginas tanto de contenido estático, como de contenido dinámico, a través de otras herramientas soportadas que facilitan la actualización de los contenidos mediante bases de datos, ficheros u otras fuentes de información.
- **Servidor HTTP Cherokee:** es un servidor web libre, multiplataforma, abierto bajo la licencia GPL. Resulta ser un servidor web bastante rápido, que también soporta las funcionalidades más comunes de servidor. Está escrito completamente en C y es escalable. Soporta registro y autenticación de usuario, puede también realizar redirecciones y soportar la configuración de servidores virtuales. No es un servidor tan grande y pesado como el Servidor HTTP Apache.
- **Internet Information Services (o Server)- IIS:** es una serie de servicios para los ordenadores que funcionan con Windows. En las computadoras que poseen este servicio instalado, se pueden publicar páginas web tanto local como remotamente y se basa en

varios módulos que le brindan la capacidad de procesar distintos tipos de páginas como ASP, ASP.NET, PHP o PERL. Existen otros servidores como los dos mencionados anteriormente, que son desarrollados en software libre, que pueden usarse como alternativa al IIS que es propiedad de Microsoft Corporation.

Para el caso de la propuesta de los autores, el servidor web seleccionado es el Apache, teniendo en cuenta las características antes mencionadas y que es el servidor web más utilizado en el mundo según estudios realizados por Netcraft, Compañía de Servicios de Internet de Inglaterra, que han determinado que cerca del 60% de los sitios activos están soportados por Apache.

### 1.6 Reutilización de software en Cuba.

Debido a que la industria cubana del software se está ordenando y va madurando gradualmente, algunos temas como los relacionados con la reutilización de componentes de software, aunque se conocen, han sido pensados y se ha comprendido su importancia en muchas de las empresas cubanas, no son de los que más se han llevado a la concreción.

DESOFT, de las empresas que más experiencia tiene en el desarrollo de software en Cuba, y la cual fue recomendada para analizar la situación actual de la reutilización en Cuba por el Dr. Jorge Luis Perdomo DiLella, Viceministro del Ministerio de Informática y Comunicaciones (MIC), no tiene mucha experiencia en reutilizar componentes de software, a pesar de que es algo que desean llevar a la práctica y para lo cual comenzaron a dar los primeros pasos. En estos momentos están inmersos en mantener la metodología del proceso de desarrollo, que todos hagan las cosas de la misma manera a la hora de la producir software. Algo que les interesa es que cuando se termine un proyecto, puedan quedar cosas para reutilizar en los próximos proyectos que deban realizar. De esta forma el primer paso para comenzar un nuevo compromiso de trabajo, es revisar lo que ya existe de experiencias anteriores para reutilizar la mayor cantidad de componentes. Esto en la actualidad está en proceso de aprobación, para iniciar lo antes posible su aplicación, la cual algunas divisiones de la empresa ven más necesarias que otras, razón por la cual es preciso que sea una política firmada por el Presidente de la misma.

Hace un tiempo, se aprobó por todas las direcciones del país una metodología de desarrollo propia, la cual han venido aplicando y en la que se esbozan algunas ideas relacionadas con la reutilización componentes de software. No obstante, hay determinas normativas como estandarizar un nombre, las clases, los objetos y demás, que no se han concretado realmente.

Por otro lado puede decirse, que se ha logrado homogeneizar la documentación, pero todavía no se ha podido alcanzar este objetivo con el código, no se ha logrado una disciplina tecnológica en ese sentido.

La fase a la que se le da más importancia en la empresa, sobretodo por parte de la Dirección de Desarrollo de Ciudad de la Habana, que es la que más ha avanzado en este tema, es al levantamiento de requisitos, el cual está bien organizado, con una documentación homogénea, etc. En el análisis y diseño han dado algunos pasos de avance y el tema de la documentación está bastante organizado, a pesar de que se debe continuar trabajando en la misma. Pero como se mencionaba anteriormente, no han logrado la misma disciplina a la hora de documentar el código, de forma tal que pueda reutilizarse después, aunque se está insistiendo en esta cuestión.

Han trabajado mucho con la herramienta Genexus[12], la cual no es para la reutilización de componentes de software directamente, pero con la que se gestiona el conocimiento, algo que es muy importante y que forma parte de la reutilización también. En sentido general, DESOFT no hecho grandes cosas en este tema, pero sí tienen grandes intenciones y planes.

En entrevista realizada con el Lic. Wilfredo Carlos Díaz, Vicepresidente de Desarrollo y Evolución de DESOFT y la Ing. Lisbet Arguello, Directora de Desarrollo de Ciudad de la Habana de dicha empresa, además de confirmar lo explicado anteriormente, expusieron que si fueran a desarrollar una biblioteca de componentes reutilizables, varios elementos eran esenciales. Según su criterio, no pueden faltar: una base de datos de conocimiento como Genexus; generadores de capas de acceso a datos; patrones de diseño de algunas interfaces, de forma que los desarrolladores los utilicen y siempre sea el mismo, que sea la cara de la empresa; también pedazos de código, pero muy bien documentados, que permita poder usar, quitar y cambiar. Esto último mencionado para la organización es muy importante, pues si se tiene una biblioteca con mucho código almacenado, se corre el riesgo de que las personas no lo utilicen porque no entiendan lo que está programado y además debe estar muy bien organizado, de forma tal que no se pierdan buscando lo que necesiten. Otro elemento que sería positivo, es que todo lo que se vaya a reutilizar, o sea, que se incorpore a la biblioteca, sea revisado previamente por un grupo de calidad, el cual certifique que ese componente está bien documentado y en condiciones de que sea reutilizado.

SOFTEL es una empresa cubana, especialista en la creación de software para la salud, que posee más de 21 años de experiencia, y opinan que la reutilización es imprescindible para alcanzar los niveles de eficiencia y productividad requeridos para garantizar la sostenibilidad de

los productos. Por esta razón, desde hace alrededor de cinco años, han incorporado a sus métodos de trabajo el tema de la reutilización, mediante su aplicación en todos los nuevos desarrollos y en la integración de nuevas soluciones.

La reutilización en esta empresa es amplia, ya que reutilizan todos lo posible, como puede ser código, componentes, datos, programas, módulos, entre otros. Pero es importante señalar, que aunque existen dificultades en cuanto a la cultura tecnológica y la organización del tema, consideran que es un factor necesario para la creación de nuevos productos, a partir de componentes generados en los anteriores.

En el caso de TECNOMÁTICA, que cuenta con más de 25 años de experiencia y se dedican generalmente a la Gestión Empresarial, principalmente al área de la Contabilidad y Finanzas, desde hace más de dos décadas emplean la reutilización en la producción de software. Señalan que ahorra mucho tiempo, permite dar soluciones estandarizadas y más seguras, pues se utiliza código que ha sido probado en la práctica. En sus inicios se hacía de forma individual y no organizada, pero desde hace dos años se aplica una política más efectiva y generalizada al respecto, lo que permite aumentar considerablemente la productividad del proceso de desarrollo. Como se había mencionado ya en SOFTEL, los elementos que más ha reutilizado TECNOMÁTICA son: el código, los componentes, datos, entre otros en menor grado.

La falta de estándares prefijados de documentación técnica de los componentes, estructuras de datos y código que se reutilizan, son las principales dificultades que han enfrentado. Apuestan por la reutilización, a pesar de los problemas que aún puedan presentarse, porque consideran que es más factible crear los nuevos productos aplicándola, pues permite a los técnicos concentrarse en los problemas de la nueva funcionalidad solicitada, al aprovechar programas y datos ya elaborados y probados con anterioridad. Además reduce el tiempo de desarrollo y aumenta la seguridad y confiabilidad del producto final.

No solo se deben mencionar las empresas cubanas productoras de software. Debemos tener en cuenta el aporte que pueden hacer al tema las Universidades, en las que se estudia por supuesto la carrera Ingeniería Informática y que también construyen software.

La Universidad Central de Las Villas “Martha Abreu” (UCLV) ha trabajado esencialmente en las esferas turismo y educación y hace alrededor de 8 meses que se dieron los pasos en el campo de la reutilización de software, como un factor importante para el ahorro de tiempos y esfuerzos. Lo que reutilizan en mayor medida son componentes del tipo de código, arquitectura, diseño, entre otros. Como principal dificultad enfrentada está el hecho de que todo

lo que desarrollan no lo conciben como un elemento que puede ser reutilizado posteriormente, por lo que tienen que dedicar tiempo y esfuerzo extras.

En la UCLV existe un grupo investigativo, Laboratorio de Investigación de Informática Educativa del Centro de Estudios de Informática, que posee más de 17 años de experiencia en el tema de la producción de software, esencialmente para la educación y quienes consideran a la reutilización como un factor importante dentro del proceso de desarrollo de un producto, debido a que contribuye al perfeccionamiento de los resultados y permite ahorrar tiempo. El problema principal que han afrontado es la documentación y la comprensión del código.

### 1.7 La reutilización de software en la UCI.

La Universidad de las Ciencias Informáticas se incorporó a la Industria Cubana del Software hace apenas 4 años y a pesar de que en tan poco tiempo ha hecho grandes contribuciones a la misma, tanto organizativa como económicamente, aún está poniendo orden a su proceso de desarrollo de software, ganando en madurez y estableciendo un grupo de normas para homogeneizar el trabajo de la producción.

Los principales problemas que afronta la producción en la Universidad, y que están dados precisamente por la inmadurez de un proceso productivo que se ha ido desarrollando sobre su propia marcha, se mencionan a continuación:

- Planificación y estimación de costos imprecisos.
- Poca productividad.
- Baja calidad de los productos.
- Incumplimiento con el tiempo de desarrollo estimado.
- Carencia de una metodología orientada a la reutilización.

Se habla de una carencia de una metodología orientada a la reutilización porque el tema no ha sido de los que más se ha trabajado en la Universidad, se ha visto de manera aislada en algunas áreas, pero no se ha impulsado desde la Dirección de Producción ninguna política al respecto de forma ordenada. No obstante, en el transcurso de la investigación, los autores entrevistaron a algunos directivos, responsables del proceso productivo en la UCI, para conocer su criterio respecto a la propuesta de implantar una biblioteca de componentes de software reutilizables.

La Dra. Alina Ruiz Jhones, Vicerrectora Primera de la UCI, opina que las bibliotecas son fundamentales como una de las vías necesarias para optimizar el proceso de producción en la Universidad y lograr convertirla en una industria. Además, su implantación garantizaría no repetir esfuerzos y contar siempre con componentes de calidad disponibles en cualquier momento, porque se estarían mejorando constantemente. En el Capítulo 2 se expone su criterio en cuanto a los objetivos de la implantación de una biblioteca y su objetividad.

En el caso de uno de los proyectos de la Facultad #4, según afirmó su líder, el Ing. Arturo Arias Orizondo, han desarrollado algunos componentes, los cuales son: un framework, un módulo de seguridad, patrones de diseño y una propuesta arquitectónica, que ya están listos para ser reutilizados por otros proyectos de la UCI; incluso tienen referencia de proyectos similares, o sea de gestión, donde determinados componentes de los mencionados, pueden ser necesarios con el fin de no repetir esfuerzos y acortar el tiempo de desarrollo del mismo. A su vez, considera que es importante potenciar el desarrollo de una biblioteca de componentes reutilizables, con el fin de poder contar en todo momento y mediante un fácil acceso, con componentes que son necesarios para un proyecto determinado, y que ya han sido desarrollados por otros proyectos, en los que se han obtenido experiencias positivas. En estos momentos sucede, que si existe algún activo desarrollado en un proyecto es necesario en otro, debe existir una coordinación entre los líderes de dichos proyectos, para llegar a un acuerdo y que se le entregue el componente que se necesita. Esta acción resulta en ocasiones más difícil que comenzar el desarrollo del componente desde cero, problema que sería erradicado con la implantación de una biblioteca de componentes de software reutilizables, como una de las políticas de la Dirección de la Producción en la UCI.

Al preguntar al Ing. Yunier Saborí Ramírez, Director de Informatización de la UCI, sobre la reutilización de componentes de software, expone que esta puede ser componentizada o no. La evolución de las arquitecturas ha traído como resultado que los software se desmiembran en partes reutilizables, las que tienen varios estadíos, pues puede ser utilizado cuando ya está desarrollado en una tecnología determinada, o el análisis y diseño de esa idea.

Puede ser reutilizable el diseño de la arquitectura hacia el sistema, y en dependencia del estilo arquitectónico, es la manera en que se implementa una arquitectura base sobre un lenguaje y una plataforma determinada. En función de cada una de estas cosas que se pueden reutilizar, las cuales pueden ser muchas, se debe pensar la forma más óptima de gestionarlas, que todos sepan dónde hay algo que puedan utilizar en su proyecto o sistema. Ahí es donde está la mayor complicación. Está claro que para llegar a ese punto se deben definir taxonomías de qué

es lo que se reutiliza, cómo se define, cómo se publica, cómo se hace que las personas lo entiendan, pues alguien puede poner códigos supuestamente reutilizables, pero si el resto no los comprende, evidentemente no lo es, o sea que hay que saber cómo se documenta el código.

En la UCI se están concibiendo varias acciones iniciales, que tienen que ver con facilitar después el trabajo de la reutilización de cualquier activo de software. Hablar de reutilización de componentes puede ser algo restrictivo, porque se pueden reutilizar muchas más cosas que un componente, hay que analizar bien entonces cuál es el concepto que se tiene de componente. ¿Se considera por ejemplo la documentación o una idea, como un componente?, ¿o se parte de que toda idea es necesario componentizarla y se va a utilizar el componente solamente? Es mejor dejarlo abierto, así solamente, la reutilización, la cual puede ser de códigos, componentes, ideas, arquitecturas, el diseño de una arquitectura, etc. Si se revisan proyectos de la UCI, ya sean de informatización, exportación o de cualquier otro tipo, se evidencia por ejemplo, que cada cual escribe y describe la arquitectura a su manera, esas son de las cosas que hay que normar, para que si un arquitecto determinado necesita revisar algo de la arquitectura de otro sistema, pueda servirle, lo entienda, y para lograr eso, todos deben redactar de la misma manera. Se pueden incluso reutilizar modelos de datos, pero para que todos los comprendan, los diseños deben hacerse de la misma forma, con la misma estructura y especificación. Es necesario entonces definir las taxonomías de lo que se reutiliza y definir todo lo que es posible reutilizar.

En una primera variante no se va a tener todo lo que es posible reutilizar, pero sí acercarse bastante o al menos poner lo que más comúnmente se reutiliza y sobretodo, precisar cómo se documenta lo que se reutiliza, para si se va a hacer un repositorio de elementos reutilizables, cada vez que se incluya algo, haya una explicación sintética, clara, de qué es la idea y cómo se puede reutilizar.

Hace un poco más de 1 año se comenzaron algunas acciones con el fin de impulsar la reutilización en la UCI, aunque ideas entorno a este tema ya se habían dado por parte de la máxima dirección de la producción en la Universidad con anterioridad. Todo surgió a partir del movimiento de software libre y las comunidades de desarrollo. Estas tienen que ver con cómo se socializa el conocimiento, que es lo más complicado generalmente, porque se pueden tener ideas muy geniales pero si no hay espacio donde reutilizarlas, entonces no tiene sentido. Para darle un inicio a la reutilización en la Universidad, se montó el Gforge, que es una herramienta de desarrollo colaborativo y como tal, socializa varias cosas como la reutilización de códigos, el

cual se publica, pero no tiene un trabajo quizás más pensado, donde se puedan tener un poco más de elementos o donde se puedan reutilizar ideas. Esta herramienta sí da el espacio para poder revisar todo lo que hace cualquier persona, siempre que el proyecto sea público, se puede revisar todo lo que tiene publicado, su documentación, los códigos, los cronogramas, porque da la facilidad incluso de generar el diagrama de Gantt dentro de un cronograma de cualquier cosa. Además da la facilidad de en la medida de que se va utilizando en un proyecto, ir reportando errores, sugerencias, colaborar en el desarrollo, o sea, da un conjunto de elementos de una comunidad de desarrollo al estilo Internet.

Ahora bien, traer esa experiencia a la UCI y crear la cultura en las personas es más difícil, porque también tiene, y es el componente más importante, un control de versiones, y de alguna manera se quería llegar con esa herramienta, hasta tener un control de todos los proyectos de la Universidad, pues da administrativamente un conjunto de estadísticas de cuántos proyectos hay, el porciento de solución que tiene, el estado del mismo, todo esto en función de la configuración que se realice.

Por otro lado está lo que se conoce como la Wikipedia de Producción, una herramienta que permite el desarrollo de artículos, de ideas, pero de manera colaborativa, donde se pueda escribir una definición, un método, un algoritmo, un concepto y todos los que tengan ideas relacionadas con el tema pueden mejorar lo que está escrito, añadirle cosas y así se va consolidando una base de conocimientos que sirve como reutilización de cualquier idea. Como apoyo están las Comunidades de Desarrollo cuyos objetivos principales son: aglutinar las fuerzas por tecnologías, propiciando la socialización de conocimientos y el desarrollo colaborativo, gestionar el conocimiento que se crea en la universidad y promover el intercambio entre los desarrolladores de una misma tecnología.

Cada una de las herramientas tiene un objetivo esencial. Lo que faltaría es el desarrollo de una herramienta que centre el tema como tal, donde se puedan tener integrados la manera de publicar un componente, un algoritmo, un código, porque en la concreta cada vez que se tiene un algoritmo lo más lógico es que se tenga en seudocódigo y quizás se ponga un ejemplo en un lenguaje determinado, pues puede suceder que ese algoritmo lo necesiten varios proyectos y que cada uno programe en un lenguaje diferente. Por eso es importante el tema de las ideas, porque la idea es lo esencial, después cada cual mejor o peor lo aterriza en un lenguaje y habrá soluciones más óptimas que otras, pero siempre que se tenga el espacio, el conocimiento se socializa. Eso es muy importante y es lo que falta en la UCI, además de que

después se debe crear la cultura de cómo utilizar esos espacios. Cada una de las herramientas que existen hoy tiene un pedazo del problema, pero ninguna los aglutina todos en una sola.

### 1.8 Modelo de Procesos para la Producción de Software en la UCI.

En la Universidad de las Ciencias Informáticas se ha ido organizando a lo largo de estos 4 años el proceso productivo de forma gradual, a la par que han aumentado los compromisos de trabajo, tanto nacionales como de exportación. Se ha ganado en seriedad, madurez, experiencia, estructura y a pesar del poco tiempo que tiene esta pequeña industria, ha dado grandes pasos en el impulso de la Industria Cubana del Software.

En el 2006 varios especialistas de la Infraestructura Productiva, se dieron a la tarea de realizar un estudio para proponer un Modelo de Procesos para la Producción del Software en la UCI. Apoyados en esa propuesta, los autores han desarrollado el presente epígrafe.

Los especialistas aseguran que trabajar sin ningún método es una opción, pero no una metodología: Las organizaciones que desarrollan o mantienen software pueden optar por trabajar "a la buena de Dios", o por seguir una metodología. Hacerlo a la buena de Dios no es tan raro. Es la primera forma que se empleó para desarrollar programas: "Aquí tenemos unos ordenadores, aquí unos señores a los que les encanta leer los manuales de programación, y se trata sencillamente de conseguir que estas máquinas terminen imprimiendo facturas (o haciendo lo que sea)". (18)

No sólo son procesos: Los procesos marcan pautas para realizar el trabajo, pero sin las personas y las herramientas (tecnología) necesarias, lo que se puede producir con ellos es más bien poco.

Las únicas combinaciones válidas para formar sistemas capaces de producir resultados son:

Personas + tecnología = Producción heroica.

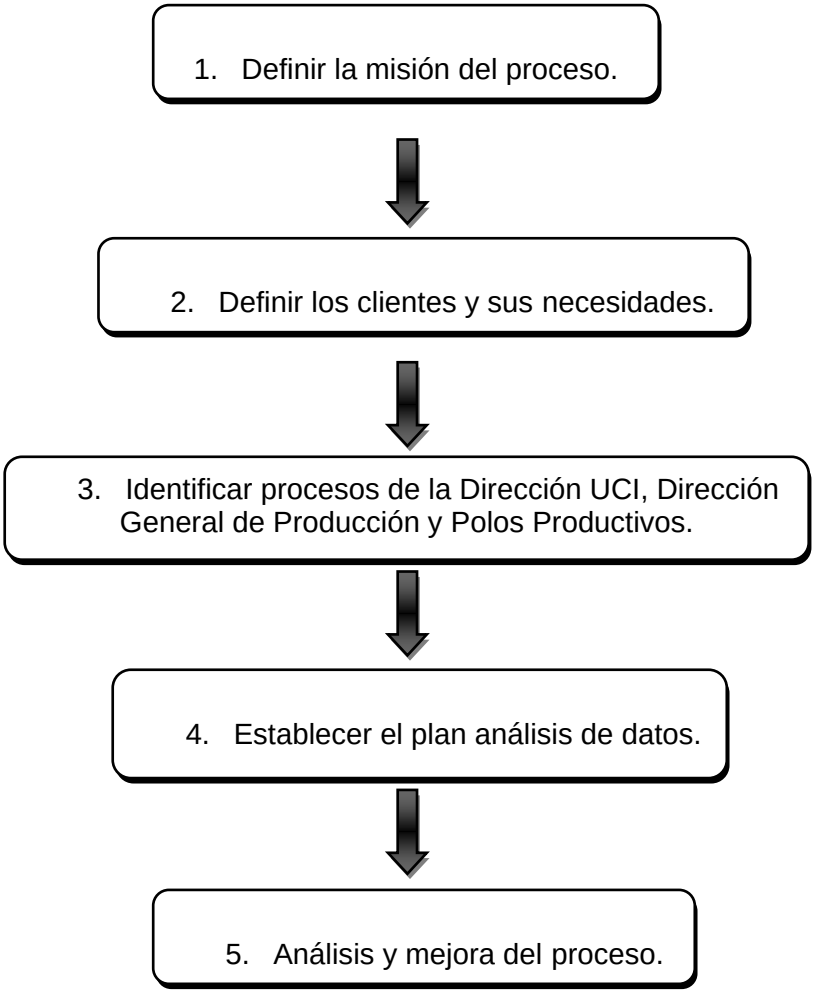
Personas + procesos + tecnología = Producción basada en procesos. (18)

Es trascendental para la UCI una proyección estratégica en la definición de sus procesos productivos, debido a que no está exenta de la problemática mencionada anteriormente. Es por esto que se realiza una propuesta de un modelo de procesos para la producción de software en la UCI, dándole un enfoque general, que sea un punto inicial para la definición de procesos específicos de la producción, de las responsabilidades y funciones dentro de la producción en la Universidad.

Para llevar adelante esta propuesta se emplearon los siguientes criterios:

- Generar un modelo de procesos genérico sin tener en cuenta la estructura organizativa actual de la UCI.
- Proporcionar un modelo genérico fácil de entender.
- Crear un Marco Referencial de Modelos de procesos que permita a la organización tener una referencia para dar un enfoque basado en procesos a su gestión de actividades y recursos en general.
- Cruzar modelos de procesos para adaptar los procesos específicos de la organización y generar modelos híbridos.
- Utilizar como marco general para el proceso productivo de la UCI tres categorías: Dirección UCI, Dirección General de la Producción y Polos Productivos.
- Destacar el Nivel de la Dirección UCI, en la planeación estratégica de la producción, su revisión y mejora continua.
- Considerar a la Dirección General de la Producción como responsable de controlar y vigilar por el cumplimiento de los objetivos estratégicos establecidos y del control de proceso productivo.
- Considerar a los Polos Productivos como ejecutores de los proyectos y el mantenimiento del software según sus especializaciones y competencias.
- Destacar la importancia de la gestión de recursos, dándole mayor importancia a la gestión del conocimiento en los polos productivos, en temas tales como: productos generados, documentación, experiencias adquiridas, etc.

Los pasos propuestos para definir el proceso productivo de la Universidad son:




Es muy útil establecer como primer paso la misión del proyecto y en caso de que esté definida, revisarla. La misión identifica el objetivo fundamental del servicio, su razón de ser. Esta debe tener en cuenta tres aspectos: qué hacer (los productos o servicios que se ofrecen), cómo hacerlo (qué procesos son seguidos) y para quién se hace (a qué clientes va dirigida).

Un proceso es un programa de actividades que van a estar organizadas de forma lógica y ordenada, encaminadas a ofrecer un producto o brindar un servicio, contando siempre con entradas, transformaciones y salidas.

El modelo de procesos propuesto para la producción de software en la UCI, lo conforman tres categorías:

1. **Dirección UCI:** Contiene el proceso de la Planeación Estratégica de la producción de software en la UCI, proporciona los lineamientos y políticas a seguir por las restantes categorías y se retroalimenta con la información generada por ellos, este proceso lo debe llevar a cabo un grupo directivo seleccionado para este fin en la UCI.
2. **Dirección General de la Producción:** Contiene los procesos de Gestión de Proyectos, Gestión de Recursos y Gestión de Procesos, los ejecuta y establece según los lineamientos y políticas establecidas por el proceso de Plantación Estratégica. Además proporciona los elementos para el funcionamiento de los polos productivos, recibe y evalúa la información generada por estos y comunica los resultados a la Dirección UCI, los procesos de esta categoría deben ser ejecutados por la estructura de dirección de la producción en la UCI.
3. **Polos Productivos:** Contiene los procesos de Administración de los Proyectos Específicos y Desarrollo, Mantenimiento y Soporte de Software, realiza sus actividades según los elementos de funcionamiento entregados por la Dirección General de la Producción, entrega información a esta y los productos generados.

El marco general de esta propuesta está basado en el modelo Mexicano MoProSoft[13], por su similitud de cómo los procesos productivos se han estado ejecutando en la Universidad actualmente.

Es importante destacar que en la categoría de la Dirección General de la Producción, dentro del proceso de Gestión de Recursos, existen tres subprocesos: Gestión de la Infraestructura Tecnológica y Servicios de Apoyo, Gestión de los Recursos Humanos y Ambiente de Trabajo y Gestión del Conocimiento de la Organización, que en su conjunto tienen la finalidad de conseguir y dotar a la organización de los recursos necesarios para lograr una correcta producción de software. (19) Este proceso de Gestión de Recursos, tiene como propósito además, crear y mantener la Base de Conocimiento de la Organización. La propuesta que se realiza en el presente trabajo de diploma de una biblioteca de componentes reutilizables de software, se enmarca en el subproceso de la Gestión del Conocimiento de la Organización.

### 1.8.1 Subproceso de Gestión del Conocimiento de la Organización.

El propósito de este subproceso es mantener disponible, administrar y actualizar la Base de Conocimiento que contiene la información y demás temas relacionados con el trabajo, la experiencia y conocimiento adquirido de todos los equipos de proyectos y los productos generados por la organización.

En función del Plan Operativo de Conocimiento de la Organización y Acciones Correctivas de Gestión de Recursos se realizan las siguientes actividades:

- **Planificación:** Establecimiento del Plan de Administración de la Base de Conocimiento que contenga la descripción de actividades para la definición o modificación del modelo conceptual de la Base de Conocimiento (BC), usuarios y sus requerimientos, así como los mecanismos de operación, mantenimiento, verificación, validación en función de los requerimientos de los usuarios.
- **Realización:** Establecimiento del Diseño de la Base de Conocimiento de la organización, está constituido por el modelo conceptual, incluyendo su metamodelo, y por los mecanismos de operación. En función de los requerimientos de los procesos, la BC está compuesta por los siguientes repositorios:
  - **Planeación Estratégica:** documentación utilizada y generada en el proceso de Planeación Estratégica.
  - **Procesos:** documentación utilizada y generada en el proceso de Gestión de Procesos.
  - **Proyectos:** documentación utilizada y generada en el proceso de Gestión de Proyectos y Administración de Proyectos Específicos.
  - **Desarrollo y Mantenimiento:** productos de software generados en el proceso de Desarrollo y Mantenimiento de Software.
  - **Recursos:** documentación utilizada y generada en el proceso de Gestión de Recursos.
  - **Recursos Humanos:** documentación utilizada y generada en el subproceso de Recursos Humanos y Ambiente de Trabajo.
  - **Infraestructura Tecnológica Adquirida y Servicios prestados:** documentación utilizada y generada en el subproceso de Infraestructura Tecnológica y Servicios de Apoyo.
  - **Documentación BC:** documentación utilizada y generada acerca de su estructura, contenido y operación.

Esta BC debe tener además otro tipo de repositorios, como por ejemplo:

- Conocimiento técnico (terminología, conceptos, metodologías).
- Bibliotecas componentes reutilizables.

- Bibliotecas de Código.
- Comunidades de Desarrollo.

Otra actividad del diseño es definir y documentar los mecanismos de operación: alimentación, consulta, mantenimiento y respaldo para cada tipo de repositorio. Finalmente se pone en operación y se da mantenimiento a la Base. (19)

## 1.9 Conclusiones.

En el capítulo se presentó un resumen del desarrollo de la reutilización desde que se dieron los primeros pasos hasta hoy en día. Se realizó un estudio sobre la situación actual de la reutilización de componentes de software en el mundo, en Cuba y más específicamente en la UCI, abordando además en la importancia que tiene la implantación de una biblioteca.

También se mencionaron algunas características de las herramientas más empleadas para reutilizar componentes de software y las principales empresas u organizaciones que las utilizan. Se definieron los conceptos fundamentales que están relacionados con el tema, dando una visión general de cómo debe ser, en una primera versión, el modelo de procesos para la producción de software en la UCI.

# Capítulo 2: Descripción de las características de una biblioteca de componentes de software reutilizables en la UCI.

## 2.1 Introducción.

En el capítulo se define el concepto de biblioteca de componentes de software reutilizables que se propone para la UCI. Se realiza una descripción de la misma teniendo en cuenta el estudio realizado sobre el tema en el Capítulo 1, las características propias de la Universidad y el desarrollo de su proceso productivo. También se describen sus objetivos, los factores que deben tenerse en cuenta para su implementación, los elementos que la forman, los componentes a reutilizar y cómo debe gestionarse la información en ella.

## 2.2 Biblioteca de componentes de software reutilizables en la UCI.

Se entiende por biblioteca de componentes de software reutilizables, para su implementación en la UCI, los repositorios donde se guardan o conservan todos aquellos componentes o activos, que participan en el ciclo de vida de un software, que permitan el trabajo colaborativo de equipos de desarrollo localizados en diferentes lugares geográficos, utilizando las facilidades que ofrecen la Intranet e Internet, de manera tal que los componentes o activos puedan ser aportados, utilizados o consultados.

Los componentes almacenados en el repositorio deben tener una representación estándar y estar bien documentados, siendo entonces el sistema gestor de la biblioteca el encargado de organizar, proteger y gestionar los mencionados componentes.

Son propiedades de esta biblioteca:

- Un sistema de gestión de bases de datos extenso, tanto en capacidad de almacenamiento como en la variedad de componentes a guardar, que permita almacenar, recuperar y consultar, componentes de software, y que sea seguro.
- Un esquema organizativo que actúe como una ayuda navegacional a través de la biblioteca, de tal manera que los usuarios puedan encontrar lo que necesiten sin grandes complicaciones.
- Una estandarización de los componentes, de forma tal que todos puedan entenderlos correctamente.
- Facilidad para crecer.
- Un espacio donde los usuarios puedan intercambiar opiniones, realizar comentarios, preguntar dudas, exponer ideas, etc.

Para llevar a la práctica la implantación de la biblioteca, no solo son importantes los aspectos tecnológicos, o sea, qué tecnologías son fundamentales para desarrollar y mantener activos de software, pues se puede tener una base tecnológica fuerte, lista para echar a andar la reutilización de componentes en la Universidad, y sin embargo, que esta no sea efectiva debido a que los factores organizativos son también esenciales para lograr el éxito.

En el capítulo anterior José Luis Fernández Sánchez, realiza un estudio sobre *"Reusabilidad y Desarrollo Orientado a Objetos"* y hace referencia a Griss en varias ocasiones. Una de ellas es para confirmar que son tres los factores organizativos: soporte de la dirección, cambios en la organización y cambios en el personal, los que llevan al éxito en la reutilización. Los autores expusieron su opinión con relación a cada uno de estos factores, teniendo en cuenta las características propias de la UCI. A partir de esas valoraciones, las acciones organizativas que se hacen necesarias para implantar la biblioteca en la Universidad son:

1. **Alcanzar la representación estándar de los componentes mediante el establecimiento de principios o premisas para el uso de las arquitecturas, plataformas, lenguajes de programación y herramientas para el desarrollo de software en la UCI.** Es esencial que todos los componentes estén escritos de la misma manera y que tengan una documentación de apoyo que los describa, para que todos puedan entender los activos que están en el repositorio.

   Pero esto no es posible si no se establecen ciertas premisas que homogenicen las arquitecturas, plataformas, lenguajes de programación y herramientas que se utilizan en la producción, debido a que hoy sucede en la UCI que esto está desordenado, pues se puede afirmar que se utilizan 18 lenguajes de programación diferentes y por ejemplo de los proyectos que programan en PHP, algunos lo hacen en PHP 4, otros en PHP 5 e incluso en PHP 6, una muestra de lo explicado se grafica en la figura 4. Se usan 11 entornos integrados de desarrollo, 15 frameworks o plugins, 5 gestores de bases de datos, 6 herramientas de modelado, 6 herramientas de gestión de proyectos y 6 sistemas operativos distintos (20); por lo que se hace evidente que con esta situación, sea imposible llevar adelante la reutilización de componentes de software. En este sentido ya se han dado algunos pasos, dentro de los que está, el establecer que para todos los proyectos debe cumplirse, que las tecnologías o herramientas a utilizar para el desarrollo de software, sean presentadas y ratificadas en el Consejo Técnico de Producción, para lo cual deben llevar un dictamen de aprobación que avale su uso.

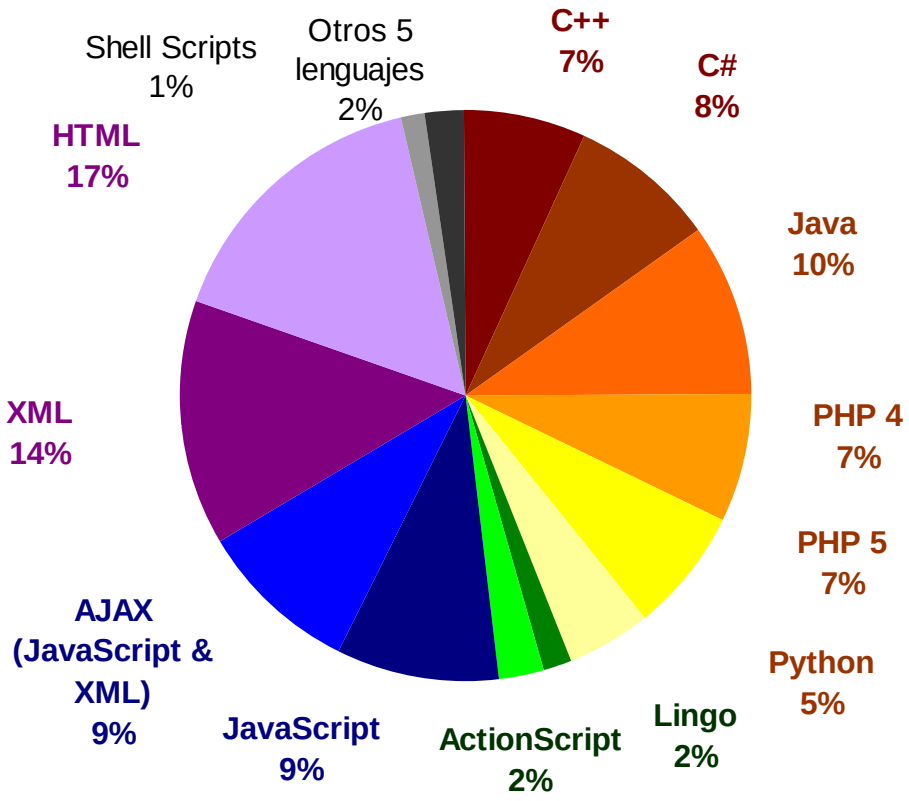



**Figura 4. Lenguajes de programación utilizados en la UCI.**

2. **Incluir la reutilización de componentes de software como una nueva disciplina dentro de la Ingeniería del Software.** Si se quiere lograr una cultura de reutilización, que se comprendan en realidad las ventajas que su aplicación aporta al proceso productivo de software, debe comenzarse a introducir el tema desde la formación de los estudiantes, como parte esencial de la pirámide que forman la producción, la investigación y la propia formación, en la búsqueda del éxito de la naciente industria en la que se aspira a convertir la UCI. Uno de los temas que es tratado en el Capítulo 26 de la Cuarta Edición del libro "Ingeniería del Software. Un enfoque práctico" de Roger S. Pressman, es el de la reutilización de componentes de software, sin embargo no se aborda en las clases, ni se hace conciencia de la importancia que tiene. Incentivar y formar a las personas es muy importante, cuando se requiere cambiar prácticas que son habituales y disminuir la desconfianza que causa el hecho de reutilizar componentes elaborados por otras personas.

3. **Dentro del Grupo de Investigación "Arquitectura de Software" organizar el tema de la reutilización.** Es un grupo que se está organizando en estos momentos y dentro del que la

> Dirección General de Producción de la Infraestructura Productiva y la Dirección de Investigación de la UCI, consideran que debe incluirse la reutilización de componentes de software. Ya está aprobado por estas dos direcciones que se desarrolle la línea investigativa.

Además de los aspectos tecnológicos y los organizativos, es significativo considerar aspectos metodológicos, que tienen que ver con cómo desarrollar y mantener los componentes, así como con los conceptos fundamentales, los cuales deben estar bien definidos, claros y ser dominados por todas las personas involucradas, para que no existan dificultades en la implementación del tema de la reutilización de componentes.

Para implantar una biblioteca de componentes reutilizables en la UCI, se debe ir de las soluciones tecnológicas más simples, hasta llegar a las de mayor complejidad. Así se sigue una evolución in crescendo, que ayuda mucho a que se vaya ganando cultura en cuanto a la reutilización y va mostrando los resultados positivos que tiene su implantación paulatinamente.

En la UCI debe comenzarse por una primera fase, donde se utilice una herramienta ya elaborada, de forma tal que se dispongan de los recursos tecnológicos existentes en la Universidad y no perder mucho tiempo en empezar todo de cero, lo cual no tiene sentido en estos momentos, pues para lograr el éxito en la reutilización de componentes, hay que comenzar a crear la cultura cuanto antes. Se deben definir un grupo de componentes para comenzar a reutilizarlos, sobretodo segmentos de código (nivel más básico de la reutilización), teniendo en cuenta el estado actual de la producción en cuanto a la homogenización de arquitecturas, plataformas, lenguajes de programación y herramientas que se utilizan en los proyectos, para que se pueda entender lo que se guarda en el repositorio y sea consecuente su utilización. Este sería el punto de partida para la posterior evolución de la biblioteca de componentes de software reutilizables, desarrollada con recursos tanto humanos como tecnológicos de la propia Universidad.

En una etapa superior, la UCI debe contar con su propia biblioteca de componentes de software reutilizables, elaborada a partir de las características y particularidades del proceso productivo. Se debe hacer un análisis profundo de los activos posibles a reutilizar, teniendo en cuenta los tipos de software que se realizan, pues no son iguales los componentes que puedan reutilizarse de una multimedia, que de un software para la salud, por ejemplo. Se incrementaría entonces la reutilización en las diversas fases del ciclo de vida del software y no solo en la implementación, como suelen ser los inicios cuando de reutilización de componentes de software se trata. Estos elementos serían aglutinados por una arquitectura, permitiendo

desarrollar aplicaciones en un dominio específico de negocio, cualquiera que este sea. Para lograr este ideal, es necesaria una organización de la producción entorno al tema, donde deben abrirse líneas de investigación y grupos para llevarlas adelante, así como la formación de un proyecto productivo que comience cuanto antes, a trabajar en la biblioteca de componentes de software reutilizables de la UCI.

Como se hacía mención en el Capítulo 1, la fase que está en la cima de la reutilización de componentes de software es la reutilización sistemática. La UCI aspira a convertirse en una gran industria de software y esta es una práctica habitual de las mismas, por lo que se ha pensado en la creación de un grupo de desarrolladores, que tendrían la función que se describe del llamado grupo de ingeniería de dominio, o sea, que sería responsable de crear y mantener activos reutilizables, para que estos sean utilizados por los proyectos que los necesiten.

### 2.2.1 Objetivos de la biblioteca.

En el capítulo anterior se menciona que los principales problemas que afronta la producción en la Universidad son:

- Planificación y estimación de costos imprecisos.
- Poca productividad.
- Baja calidad de los productos.
- Incumplimiento con el tiempo de desarrollo estimado.
- Carencia de una metodología orientada a la reutilización.

Una vía para hacer estas dificultades menores, es la implantación de una biblioteca de componentes de software reutilizables, debido a que precisamente los objetivos esenciales de su establecimiento son:

- No repetir esfuerzos en la producción.
- Reducir considerablemente el tiempo de entrega de los proyectos.
- Contar en todo momento, con componentes de calidad para utilizarlos en cualquier etapa del ciclo de vida del software, pues se estarían mejorando constantemente.
- Obtener productos más acabados y mejores, teniendo en cuenta que todos los componentes que se incluyan en la biblioteca deben estar certificados por calidad.

- Permitir una socialización del conocimiento, lo cual se inscribe en el estilo de trabajo colaborativo que le interesa potenciar a la máxima dirección de la producción de software en la UCI.

El criterio del Ing. Renier Pérez García, Director General de la Infraestructura Productiva, es que la implantación de la biblioteca es el primer paso para la gestión del conocimiento en la UCI y concuerda con los objetivos planteados anteriormente.

La Dra. Alina Ruiz Jhones, Vicerrectora Primera de la UCI, quien atiende directamente el tema del proceso productivo, considera la biblioteca como una de las vías necesarias para optimizar el proceso de producción en la Universidad y lograr convertirnos en una verdadera industria de software. Concuerda con los objetivos expuestos y considera la implantación de una biblioteca es objetiva, pues es perfectamente posible si se tienen en cuenta las condiciones de la UCI para organizarlo y potenciarlo, como quizás no haya parangón en el mundo.

### 2.2.2 Elementos que forman parte de la biblioteca. Sus servicios.

La biblioteca es una herramienta para el desarrollo de software en forma comunitaria, que permite organizar y administrar grandes cantidades de proyectos. La forman un conjunto integrado de herramientas que facilitan el trabajo en colaboración y proporcionan un grupo de servicios:

- **Repositorios de componentes:** bases de datos donde se guardan todos los componentes que se generan durante el ciclo de vida del software, que son reutilizables, y que pueden visualizarse en dependencia de la categoría que el usuario determine de las que sean definidas.
- **Wiki:** es una base de datos de conocimiento que permite la gestión del mismo, mediante la reutilización de ideas, documentos y que posee un motor de búsqueda, que hace más fácil encontrar la información relacionada con el tema que se desea. En la Universidad existe la WikiProd, la que tiene entre sus objetivos: crear una base de conocimiento que sirva de ayuda al proceso de Formación desde la Producción, brindar una herramienta que sirva para gestionar el conocimiento tácito generado en la actividad productiva-investigativa y ofrecer un espacio donde los usuarios de la red puedan tener a la vuelta de un click, información generada en el entorno universitario. No es necesario repetir esfuerzos, sino utilizar los espacios que ya tenemos, solo que integrados en una herramienta, con una cultura de uso adecuada y consecuente.

- **Control de versiones:** permite llevar el control de las versiones que se realizan de los componentes y productos, así como el historial de cada uno de ellos.
- **Foros de discusión:** es un espacio donde los usuarios puedan intercambiar opiniones, realizar comentarios, preguntar dudas, exponer ideas, etc.
- **Comunicación mediante listas de distribución de correo:** posibilita enviar mensajes a un grupo de personas que se necesite, de las que poseen los permisos para gestionar los componentes de la biblioteca, con informaciones de interés referentes a la biblioteca, o noticias importantes que se definan por la administración de la misma.
- **Gestión y monitoreo de la planificación de tareas:** cada proyecto define sus hitos en el desarrollo del software y a partir de ellos realiza su planificación, teniendo en cuenta el tiempo de entrega del producto y las tareas a desarrollar. Dentro de esa planificación deben incluirse tareas relacionadas con la biblioteca y la misma debe ser capaz de alertar el estado de cumplimiento.

Los elementos mencionados son los que no pueden faltar en la biblioteca que se propone, o sea, los básicos. El grupo de desarrollo responsable de impulsar este proyecto, al realizar el análisis para su desarrollo, debe considerar qué otros elementos pueden incluirse, teniendo en cuenta el proceso de desarrollo de software y sus tendencias técnicas.

### 2.3 Componentes que pueden ser reutilizados en la biblioteca.

Cuando se habla de reutilizar en el marco de la producción de software, en lo primero que se piensa generalmente es en reutilizar código de programación, líneas de código compiladas en librerías o frameworks, pero el concepto de reutilización como se ha fundamentado en el Capítulo 1, ha evolucionado. Hoy en día se reutiliza todo lo que entra en el ciclo completo de vida del software y esa es la propuesta para la concreción de la reutilización en la UCI.

Para continuar abordando cuáles son los componentes o activos que deben reutilizarse, primero se debe definir ¿qué es un componente o activo de software para la biblioteca en la UCI? un componente o activo de software, en el marco de la UCI, es cualquiera de los artefactos que se generen durante alguna de las etapas del ciclo de vida de un software, un módulo generado a partir de estos artefactos o un producto ya elaborado que está formado por dichos módulos. Pero para que estos componentes o activos sean reutilizables, es indispensable que estén bien documentados, para que todos puedan entenderlos. Sin comprensión plena de los componentes, no hay reutilización posible.

Es importante señalar que en dependencia del tipo software que sea, son diferentes los activos que se generan, aunque se puedan definir algunos componentes generales a reutilizar por todos, no es igual lo que puede reutilizarse de una multimedia, que de un software de salud, o de inteligencia artificial por ejemplo. Por este motivo, una de las acciones importantes para llevar adelante la reutilización en la UCI, es hacer un estudio para definir de cada uno, qué es posible incluir en la biblioteca.

Para ejemplicar el caso de las multimedias o software educativo, se entrevistó a la MSc. Yadenis Piñero Perez, Directora de Software Educativo de la UCI. Considera que algunos de los componentes que pueden tener mayor nivel de reutilización son:

- Componentes de ejercicios comprobatorios de conocimientos como:
  - Arrastrar.
  - Enlazar.
  - Seleccionar la respuesta correcta.
  - Completar información en un texto.
  - Crucigramas.
- Línea de tiempo.

Teniendo en cuenta el concepto definido de componente o activo, algunos de los que se pueden reutilizar de forma general son:

- Una especificación de requisitos.
- Un modelo de negocio.
- Una especificación de diseño.
- Un algoritmo.
- Un segmento de código.
- Un patrón de diseño.
- Una arquitectura de dominio.
- Un esquema de base de datos.
- Una especificación de prueba.
- La documentación de un sistema.
- Un plan.

Para que un componente sea reutilizable resulta imprescindible que esté bien documentado, garantizando así que se comprenda por cualquier persona que lo consulte y de esta manera,

defina si cumple con los requisitos que necesita para el sistema que está desarrollando y pueda entonces utilizarlo. Es necesario que la Dirección de Calidad de la Universidad, establezca normas para la documentación que acompaña cada uno de los componentes, de forma tal que sea redactada, teniendo en cuenta los mismos parámetros y requisitos, para una buena comprensión de los activos incluidos en la biblioteca por todos los usuarios.

La documentación que acompaña a cada componente debe tener un grupo de elementos que a consideración de los autores no se deben dejar de tener en cuenta:

- Historia o registro de uso.
- Clasificación del activo.
- Descripción del activo, donde se explique de forma tal, que se pueda prescindir para su uso del equipo o persona que lo realizó.
- Mantenimiento y soporte, ayuda en línea.

## 2.4 Conclusiones.

En el capítulo se definieron conceptos importantes para comenzar a impulsar la reutilización de componentes en la UCI, como son el de biblioteca de componentes reutilizables y el de componente o activo de software. Se describieron además las características generales de la biblioteca que se propone desarrollar en la Universidad, así como sus objetivos.

También se mencionaron factores organizativos que son importantes para la implantación de la biblioteca. Se definieron elementos básicos que la forman y los componentes a incluir para su reutilización.

# Capítulo 3: Propuestas para la implantación de una biblioteca de componentes de software reutilizables en la UCI.

## 3.1 Introducción.

En el capítulo se realizan dos propuestas para la implantación de una biblioteca de componentes reutilizables de software en la UCI. La primera para comenzar a utilizar de inmediato el Gforge y la segunda para que paralelamente a la implantación de la primera, se desarrolle la propia biblioteca de la UCI. Además se cómo llevar a la práctica cada una de estas propuestas.

## 3.2 El Gforge como propuesta inmediata, pero no como solución permanente.

Como se mencionó en el Capítulo 1, hace alrededor de un año que en la UCI se comenzaron a dar los primeros pasos para estimular la reutilización de componentes de software y con ese fin se instaló un servidor con el Gforge, herramienta basada en plataforma libre para el desarrollo de software en forma comunitaria, que permite organizar y administrar grandes cantidades de proyectos y proporciona un conjunto integrado de herramientas que facilitan el trabajo en colaboración. La herramienta no funciona a plenitud porque no se dieron los pasos organizativos necesarios para lograr el éxito de la misma, pues a pesar de que están registrados, según el sitio del Gforge UCI, 94 proyectos, se detectaron navegando en él los siguientes aspectos:

- La herramienta no es utilizada periódicamente por todos los proyectos registrados.
- Proyectos que no poseen ningún componente ni documentación publicada.
- Proyectos que publicaron algún elemento cuando se registraron y no han actualizado más su página.
- Proyectos que solo han utilizado el foro para intercambiar ideas sobre algún tema relacionado con el mismo.
- No está actualizado el ranking que se publica en la página de inicio.

Tomando como punto de partida esta información se realiza un análisis para utilizar el Gforge como una solución inmediata para la reutilización, teniendo en cuenta las ventajas que ofrece al proceso productivo y los pasos que son necesarios dar para revertir la situación ilustrada anteriormente, pero no como solución permanente, por los inconvenientes que presenta, si se toman como referencia las características particulares que posee dicho proceso en la UCI.

### 3.2.1 El Gforge como propuesta inmediata. Ventajas de su aplicación en la UCI.

Se propone comenzar a utilizar el Gforge en la UCI porque su aplicación trae consigo algunas ventajas, importantes para el buen desarrollo de la reutilización en la Universidad.

La herramienta está basada en plataforma libre, por lo que posee código abierto y esto posibilita que esté disponible para revisarlo, modificarlo, mejorarlo y adaptarlo como se requiera; además de que no está a expensas de que, al salir una versión nueva, este pueda inutilizarse, como puede suceder con facilidad con herramientas propietarias.

Resulta importante el hecho de que la tecnología ya esté disponible en la UCI, pues como se ha mencionado anteriormente, se instaló un servidor con el Gforge hace alrededor de un año. Por este motivo, no se requiere gastar nuevos recursos tecnológicos ni esfuerzos extras para su implantación. De esta manera para comenzar a dar pasos en la implantación de una biblioteca, se propone utilizar la tecnología que ya está disponible para todos, el Gforge, y que solo requiere que esté complementada, para el éxito de su uso, por una buena organización, por una política establecida desde la Dirección General de la Infraestructura Productiva de la Universidad.

Como consecuencia de que se haya trabajado con el Gforge en la UCI, a pesar de que no se han logrado sus objetivos a plenitud, se evidencia que existe personal preparado para el manejo de la herramienta, que la conocen, han estudiado, utilizado o administrado. Por tanto, no es necesario utilizar tiempo en la capacitación del personal que debe operar la misma, solo apoyarse en las mismas personas que ya están listas. Serían estas propias personas las encargadas de instruir, de forma organizada, a los proyectos que comiencen a utilizar la herramienta y a los grupos de calidad que deban certificar que los componentes están listos para ser publicados.

Otra arista importante de su utilización, es que se comience a crear una cultura alrededor de la reutilización, incipiente en la Universidad, y muy necesaria para llevar a la práctica satisfactoriamente la implantación de una biblioteca de componentes de software reutilizables en la UCI. Para ello, resulta esencial definir cómo se utilizará el Gforge de forma inmediata en la Universidad, propuesta que se desarrolla en el presente capítulo.

### 3.2.2 Pero no como solución permanente.

El Gforge, a pesar de ser una buena opción en estos momentos para impulsar el inicio de la reutilización de componentes en la UCI, como ilustran las ventajas explicadas anteriormente, se considera que no es la solución más óptima para implantar una biblioteca de componentes de

software reutilizables, que se adapte a las características del proceso productivo en la Universidad. A esta conclusión los autores han llegado, después del estudio realizado durante la investigación y las razones que la justifican son:

- **No posee una wiki:** no posee un espacio donde se pueda crear una base del conocimiento que apoye el proceso de Formación desde la Producción, que ofrezca una herramienta como una de las vías para gestionar el conocimiento tácito concebido en la actividad productiva-investigativa y que facilite a los usuarios de la red, tener la información necesaria sobre un tema determinado que se ha generado en el entorno universitario, tan solo al alcance de un click. Desde la wiki se puede acceder a las Comunidades de Desarrollo, a través de las cuales se agrupan las fuerzas, promoviendo el intercambio entre los desarrolladores de una misma tecnología; se propicia además la socialización de conocimientos y el desarrollo colaborativo.

- **No brinda la seguridad necesaria:** a pesar de que la herramienta permite controlar los niveles de acceso a los recursos publicados, estos no se adaptan al entorno de la Universidad, teniendo en cuenta, a partir de cómo está organizado el proceso de producción en la UCI, la manera en que se deben gestionar los componentes en la biblioteca, de tal forma que se pueda garantizar la confiabilidad, integridad y calidad de los activos que se publican, con el fin de que logren utilizarse en nuevos sistemas por cualquier persona. Para obtener mayor seguridad de la información guardada en la biblioteca, se deben tener en cuenta las vulnerabilidades que posee esta herramienta, pues ha sido hackeada en lugares donde ha estado instalada por debilidades que posee.

  La seguridad se convierte además en un elemento esencial, si se tiene en cuenta que la documentación de los proyectos, según el concepto de componente definido por los autores, también es reutilizable, y esta es información muy sensible de los mismos, por lo que se requiere altos grados de confiabilidad, integridad y disponibilidad.

- **No posee una interfaz que se adapte proceso de desarrollo UCI:** es una herramienta de desarrollo colaborativo, que no se adapta completamente a los parámetros de medición que se establecen en el avance de la producción, como el hecho de que se trabaja con estudiantes y profesores, los cuales tienen horarios determinados. Tampoco concuerda con las métricas definidas en la Universidad, donde se trabaja por casos de uso, con conceptos de producto y procesos específicos para la UCI, recursos asignados a los proyectos, hitos de desarrollo y otras particularidades del proceso, que son importantes para poder llevar a la práctica la implantación de una biblioteca.

A pesar de estas limitaciones, se propone comenzar a reutilizar componentes de software a través de la herramienta, debido a que su aplicación inmediata posee un grupo de ventajas, que hacen favorable la opción del Gforge como punto de partida de la reutilización en la UCI, las cuales fueron abordadas en el subepígrafe anterior.

### 3.3 Propuesta de cómo se debe comenzar a utilizar el Gforge en la UCI.

La producción en la Universidad se está organizando por Polos Productivos, agrupando de esta forma los proyectos por su área temática dentro de las propias facultades. Teniendo en cuenta esta organización, la propuesta es implementar el Gforge en uno de los polos, como una experiencia que sea monitoreada, con el fin de demostrar la factibilidad de la reutilización de componentes en la UCI de manera organizada desde la Dirección de la Producción y que sea un experimento del que se puedan obtener resultados positivos y negativos, que contribuyan a la posterior implantación de la reutilización generalizada a todos los proyectos. A pesar de que se propone la experiencia en uno de los polos, es posible que en la medida que avance la misma, se puedan ir incorporando otros proyectos que no pertenezcan a dicho polo.

La selección de qué Polo Productivo es más favorable para llevar adelante la experiencia, está relacionada con un grupo de elementos afines con el funcionamiento y organización de los mismos, los cuales están actualmente en proceso de definición por la máxima dirección de la producción en la Universidad. Pero sí se evidencian de forma general un grupo de ventajas y desventajas, que trae consigo el hecho de aplicar la experiencia en un polo. Se considera que es el lugar más idóneo para obtener resultados satisfactorios.

**Fortalezas de implementar la experiencia en un polo:**

- Los proyectos tienen un perfil similar y esto hace que la reutilización se facilite.
- El polo pertenece a una facultad, por lo que la estructura del Vicedecano de Producción es la responsable del impulso y control de la experiencia.
- Se comienza a instaurar la reutilización ya adaptada a la forma en que debe estar organizada la producción de la UCI completamente, en un futuro no muy lejano.
- Se inicia una cultura entorno a la reutilización de componentes de software.

**Debilidades de su aplicación:**

- Poca madurez de los polos.
- Insuficiente organización o integración de los proyectos como un polo.

- Todavía no están claras un grupo de definiciones dentro de los mismos, como podría ser el uso de las tecnologías, herramientas, etc.

Para llevar adelante la propuesta, se hace necesario desarrollar un grupo de acciones:

- Revisar el servidor en el que está instalado el Gforge UCI y hacer las adaptaciones necesarias, para facilitar la utilización de la herramienta en el polo que sea seleccionado, teniendo en cuenta como aspecto fundamental, los requisitos de hardware necesarios para que su funcionamiento sea óptimo.
- Activar un ranking, teniendo en cuenta las opciones que brinda el Gforge, con el fin de lograr un sistema de información constante y actualizado, además de la posibilidad de establecer una emulación entre los proyectos, que incentive la reutilización de los componentes.
- Seleccionar un Polo Productivo, después de analizar cuál es el que está en mejores condiciones para implementar la experiencia.
- Definir el grupo de calidad responsable de certificar que los componentes están listos para ser incluidos en el repositorio. En el caso de que el proyecto tenga su propio grupo, ellos deben ser los responsables y en los casos que no tienen, es trabajo del grupo de calidad de la facultad.
- Explicar la importancia y ventajas que aporta al proceso productivo de la UCI comenzar a reutilizar componentes de software, a los proyectos y los grupos de calidad que sean seleccionados para llevar adelante la experiencia.
- Seleccionar el personal encargado de capacitar a los estudiantes y profesores involucrados.
- Capacitar en el uso de la herramienta y en cómo documentar los componentes de manera que estén aptos para ser incluidos en el repositorio, a los miembros de los proyectos y los grupos de calidad. Explicarles en detalle cómo va a funcionar la experiencia, las políticas a seguir para la subida de los componentes y los conceptos esenciales de la reutilización ya definidos en el presente Trabajo de Diploma para la UCI.

Para que estas acciones se lleven a la práctica es esencial la exigencia, control y el entendimiento a plenitud por parte de los directivos de la facultad, de la importancia del tema de la reutilización para la UCI como naciente industria del software.

Es muy importante para lograr el posterior éxito de la implantación de una biblioteca de componentes de software reutilizables, que se adapte completamente al proceso productivo de

la UCI y desarrollada con recursos tecnológicos y humanos propios, en la cual interactúen todos los proyectos involucrados en el mismo, que la experiencia propuesta de utilizar de manera inmediata el Gforge en uno de los Polos Productivos sea positiva, para demostrar con resultados palpables y rotundos que la reutilización en la UCI es realmente objetiva y viable.

## 3.4 Propuesta para el desarrollo de una biblioteca de componentes de software reutilizables en la UCI.

La solución más idónea para la implantación de una biblioteca de componentes de software reutilizables en la UCI, después de todos los análisis realizados anteriormente, es el desarrollo de una biblioteca propia, adaptada a las características del proceso productivo de la Universidad y utilizando recursos tanto tecnológicos como humanos de la UCI. De esta forma, se favorece y facilita la reutilización de componentes de software, logrando un producto que se ajusta a las necesidades de reusabilidad de la Universidad.

Para el desarrollo de esta biblioteca, de forma paralela a la implementación de la experiencia del Gforge en uno de los polos productivos, se hacen necesarias como acciones organizativas inmediatas:

1. **Definir a qué facultad se le asigna el proyecto.** Teniendo en cuenta el perfil de las facultades y además las tecnologías que se utilizan en cada una de ellas, se debe seleccionar la facultad adecuada para el desarrollo de una biblioteca de componentes reutilizables para la UCI. De esta manera, el proyecto creado es responsabilidad de la máxima dirección de la facultad y puede seguirse y controlarse el desarrollo de la misma favorablemente. Es muy importante explicarle a la facultad seleccionada la importancia y objetivos del desarrollo de este proyecto, y la necesidad de hacer chequeos de su avance y desarrollo periódicamente, cuestión que debe ser monitoreada desde la Dirección de Producción de la Universidad, específicamente se propone que sea desde la Dirección de Informatización por las características del proyecto.
2. **Crear un grupo de desarrollo.** Una vez definida la facultad debe procederse a la selección del equipo de estudiantes y profesores, que serán los responsables del desarrollo de la biblioteca. Se debe definir el líder del proyecto y los demás roles a desempeñar dentro del mismo por cada uno de los integrantes. A partir de este momento se comienza a desarrollar la biblioteca partiendo del levantamiento de requisitos, lo que conlleva a las siguientes etapas en la vida de un software como son el análisis, diseño, implementación, prueba, etc. Es importante la preparación de los desarrolladores en las herramientas en las que se debe desarrollar la biblioteca, así como en aspectos relacionados propiamente con las bibliotecas

de componentes de software reutilizables, para que se pueda comprender la importancia del trabajo.

La biblioteca que se propone desarrollar consiste en una aplicación web cliente-servidor que integre básicamente las herramientas:

- Repositorios de componentes.
- Base de Datos de Conocimiento (Wiki).
- Control de versiones.
- Foros de discusión.
- Comunicación mediante listas de distribución de correo.
- Gestión y monitoreo de la planificación de tareas.

### 3.4.1 Requisitos Funcionales.

Los Requerimientos Funcionales especifican acciones básicas que el sistema debe ser capaz de realizar, sin tomar en consideración ningún tipo de restricción física. Para su captura se emplearon técnicas como la investigación y la revisión de documentos. En la descripción de los casos de uso se explican mejor cada uno de ellos. Estos, son requisitos funcionales básicos de la biblioteca, pero debe realizarse un análisis más profundo de la misma, por parte de los analistas del equipo de desarrollo seleccionado, para determinar cuáles son todos los requisitos funcionales del sistema.

1. **Gestionar Usuarios del Sistema.**
    - 1.1 Adicionar Usuario.
    - 1.2 Eliminar Usuario.
    - 1.3 Buscar Usuario.
    - 1.4 Modificar Usuario.
2. **Gestionar Componentes.**
    - 2.1 Descargar Componentes.
    - 2.2 Adicionar Componente.
        - 2.2.1 Enviar a responsable de calidad datos del activo a publicar.

**3. Buscar Componente**

3.1 Buscar por Palabra Clave.

3.2 Buscar por Temática.

3.3 Buscar por Lenguaje.

3.4 Buscar por Plataforma.

3.5 Buscar por Idioma.

3.6 Buscar por Polo Productivo.

3.7 Buscar por Facultad.

3.8 Buscar por Proyecto.

**4. Gestionar Calidad de Activo.**

4.1 Publicar Activo.

4.1.1 Enviar correo al usuario con la validez de publicación de Activo.

4.2 Eliminar Activo.

4.2.1 Enviar correo electrónico al usuario con las causas por las que fue rechazado el componente.

**5. Generar Reportes.**

5.1 Generar Reporte de Componentes Publicado por un Usuario.

5.2 Generar Reporte de Componentes por Lenguaje.

5.3 Generar Reporte de Componentes por Plataforma.

5.4 Generar Reporte de Componentes Multiplataforma.

5.5 Generar Reporte de Componentes Publicados.

5.6 Generar Reporte de Componentes No Publicados.

5.7 Generar Reporte de Componentes por Idioma.

5.8 Generar Reporte de Componentes por Polo Productivo.

5.9 Generar Reporte de Componentes por Facultad.

5.10 Generar Reporte de Componentes por Proyecto.

5.11 Generar Reporte de Componentes por Temática.

5.12 Generar Reporte de Componentes por Palabras Clave.

5.13 Generar Reporte de Usuarios Registrados en la Biblioteca.

5.14 Generar Reporte de Componente más Utilizado.

**6. Mostrar Ranking de Publicaciones.**

6.1 Mostrar Ranking de Proyectos.

6.2 Mostrar Ranking de Lenguaje.

6.3 Mostrar Ranking de Plataforma.

6.4 Mostrar Ranking de Componentes.

6.5 Mostrar Ranking de Usuarios.

**7. Registrar Petición de Mejora.**

7.1 Notificar al autor del componente una Solicitud de Mejora de forma automática.

7.2 Notificar a los usuarios por correo el problema detectado.

**8. Autenticar Usuario.**

**9. Importar Componentes.**

### 3.4.2 Requisitos no funcionales.

Los requerimientos no funcionales añaden funcionalidad al producto, pues hacen que este sea fácil de usar, seguro e interactivo. Sin embargo, la razón fundamental de que esta funcionalidad sea parte del producto, es brindarle las características deseadas. Forman una parte significativa de la especificación. Son propiedades o cualidades que el producto debe tener, las cuales hacen que el mismo sea atractivo, usable, rápido o confiable.

**1. Seguridad.**

1.1 Confidencialidad. Se establecerán diferentes niveles de acceso para los usuarios que interactúen con el sistema, para garantizar que la información manejada esté protegida de acceso no autorizado.

1.2 Integridad. Debido a la importancia de la información manejada, resulta de extremo cuidado su protección, por lo que la trasmisión de datos por la red se realizará a través de un protocolo seguro.

1.3 Disponibilidad. La información estará disponible las 24 horas del día para el trabajo de los usuarios, así como para las acciones de mantenimiento.

**2. Legales.**

2.1 El sistema se ajustará a lo planteado en la versión vigente de la Resolución No. 132.

2.2 Estará regido por el manual de normas y procedimientos de la Resolución No. 132/2004.

**3. Software.**

3.1 Sistema Operativo Linux.

3.2 Servidor Web Apache. El servidor principal tendrá la aplicación web y se encargará de todos los pedidos y envíos. Existirán dos servidores donde estén los datos, tipo espejo.

3.3 Lenguaje de programación PHP.

3.4 Sistema gestor de bases de datos PostgreSQL.

3.5 Antivirus Kaspersky.

**4. Hardware.**

4.1 Procesador P4 3.0 GHz.

4.2 2 GB RAM.

4.3 160 GB SATA HD.

### 3.4.3 Definición de los actores del sistema.

| **Actores** | **Justificación** |
|---|---|
| Usuario | Es la persona que tiene permiso a la manipulación de los activos. |
| Responsable de Calidad | Es la persona que se encarga de garantizar la calidad de los activos y su publicación. |
| Administrador | Es la persona encargada de administrar todas las actividades que se realizan en el sistema. |

### 3.4.4 Definición de los casos de uso.

| | |
|---|---|
| CU1 | Gestionar Usuario |
| **Actor** | Administrador |
| **Descripción** | El caso de uso comienza cuando el actor hace una petición para realizar alguna actualización de los usuarios o eliminar un usuario. |
| **Referencia** | RF1 |

| | |
|---|---|
| CU2 | Gestionar Componentes |
| **Actor** | Usuario |
| **Descripción** | El caso de uso comienza cuando el actor adiciona o descarga un componente. |
| **Referencia** | RF2 |

| | |
|---|---|
| CU3 | Buscar Componente |
| **Actor** | Usuario |
| **Descripción** | El caso de uso comienza cuando el actor solicita la búsqueda de un activo. |
| **Referencia** | RF3 |

| | |
|---|---|
| CU4 | Gestionar Calidad de Activo |
| **Actor** | Responsable de Calidad. |
| **Descripción** | El caso de uso se inicia cuando el responsable de calidad aprueba el componente y lo publica o lo elimina. |
| **Referencia** | RF4 |

| | |
|---|---|
| CU5 | Gestionar Reportes |
| **Actor** | Usuario |

| | |
|---|---|
| **Descripción** | El caso de uso comienza cuando un actor solicita un reporte sobre un tema específico. |
| **Referencia** | RF5 |

| | |
|---|---|
| CU6 | Gestionar Reportes de Calidad. |
| **Actor** | Responsable de Calidad. |
| **Descripción** | El caso de uso comienza cuando un actor solicita un reporte sobre un tema de calidad |
| **Referencia** | RF5 |

| | |
|---|---|
| CU7 | Gestionar Reportes de Administración |
| **Actor** | Administrador |
| **Descripción** | El caso de uso comienza cuando un actor solicita un reporte sobre un tema administrativo |
| **Referencia** | RF5 |

| | |
|---|---|
| CU8 | Gestionar Reportes de Usuario |
| **Actor** | Usuario |
| **Descripción** | El caso de uso comienza cuando un actor solicita ver un reporte. |
| **Referencia** | RF5 |

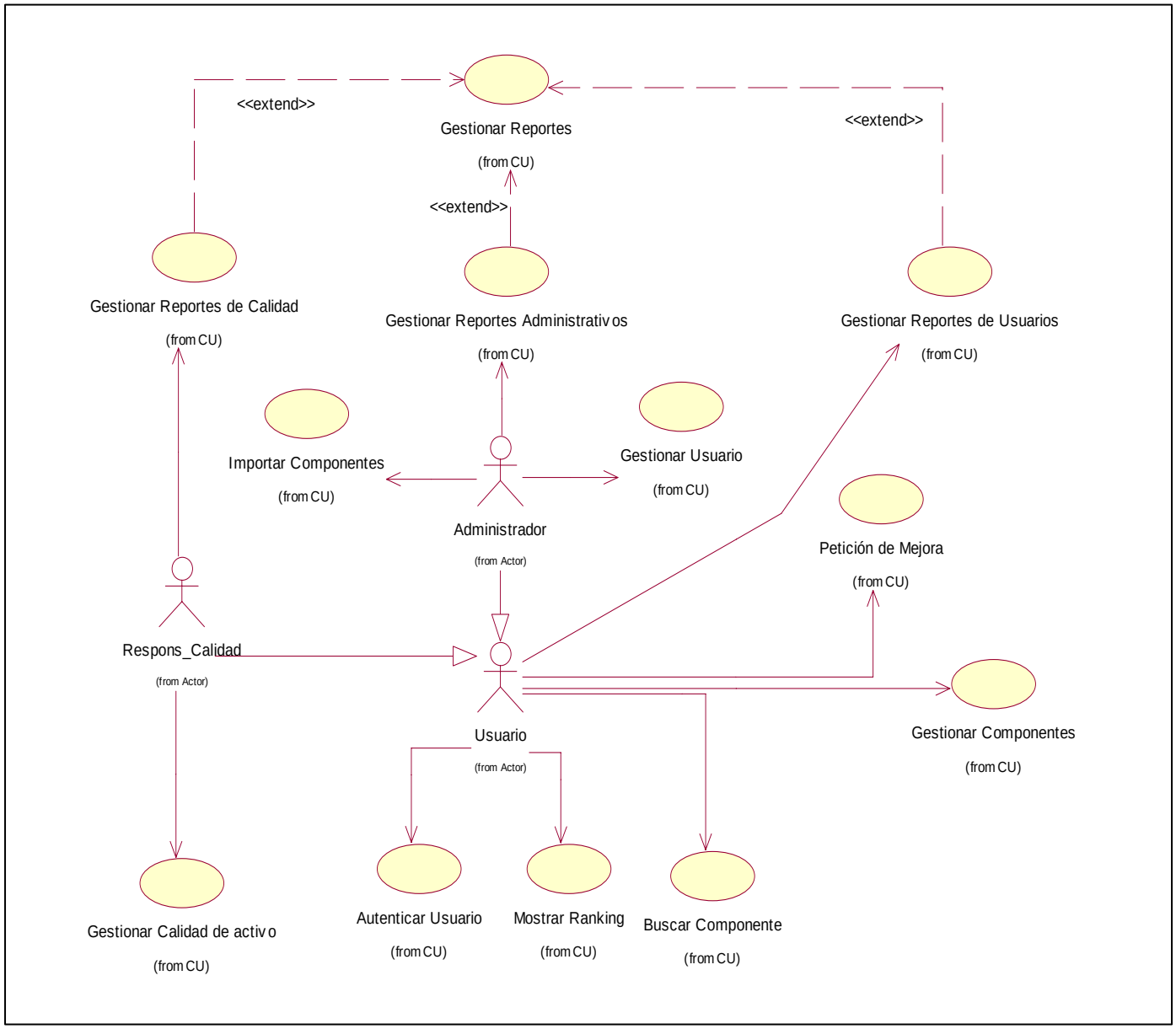
| | |
|---|---|
| CU9 | Mostrar Ranking |
| **Actor** | Usuario |
| **Descripción** | El caso de uso comienza cuando un actor accede a la biblioteca. |
| **Referencia** | RF6 |

| CU10 | Petición de Mejora |
|---|---|
| **Actor** | Usuario |
| **Descripción** | El caso de uso comienza cuando el actor hace una petición de mejora de componente. |
| **Referencia** | RF7 |

| CU11 | Autenticar Usuario |
|---|---|
| **Actor** | Usuario |
| **Descripción** | El caso de uso comienza cuando el actor se autentifica para acceder a la biblioteca. |
| **Referencia** | RF8 |

| | |
|---|---|
| CU12 | Importar Componentes |
| **Actor** | Administrador |
| **Descripción** | El caso de uso comienza cuando el actor importa componente almacenados en otra base de datos. |
| **Referencia** | RF9 |

### 3.4.5 Diagrama de Caso de Uso del Sistema.



### 3.4.6 Casos de Uso Expandidos.

<table>
<tr><th colspan="3">Caso de uso</th></tr>
<tr><td>CU1</td><td colspan="2">Gestionar Usuario</td></tr>
<tr><td>Propósito</td><td colspan="2">Adicionar, Modificar o Eliminar un usuario.</td></tr>
<tr><td colspan="3">Actores: Administrador</td></tr>
<tr><td colspan="3">Resumen: Este caso de uso comienza cuando el actor desea adicionar, modificar o eliminar un usuario, el sistema adiciona el usuario o realiza la búsqueda del usuario a modificar o eliminar y después guarda los cambios.</td></tr>
<tr><td>Referencias</td><td colspan="2">RF1</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td colspan="2">1. El actor envía la solicitud de:<ul><li>Modificar Usuario</li><li>Eliminar Usuario</li><li>Adicionar Usuario</li><li>Buscar Usuario</li></ul></td><td>2. En caso que se reciba:<ul><li>Modificar Usuario: ver sección “Modificar Usuario”</li><li>Eliminar Usuario: ver sección “Eliminar Usuario”</li><li>Adicionar Usuario: ver sección “Adicionar Usuario”</li><li>Buscar Usuario: ver sección “Buscar Usuario”</li></ul></td></tr>
<tr><td colspan="3">Sección: Modificar Usuario</td></tr>
<tr><td colspan="2"></td><td>3. Muestra una interfaz para que el actor realice la búsqueda del usuario que se desea modificar.</td></tr>
<tr><td colspan="2">4. Selecciona un criterio de búsqueda e introduce los datos para realizarla.</td><td>5. Realiza la búsqueda y muestra una lista con los resultados.</td></tr>
<tr><td colspan="2">6. Selecciona el usuario al que se le va a modificar los datos.</td><td></td></tr>
<tr><td colspan="2">7. Introduce los cambios y realiza la operación “Guardar”.</td><td>8. Analiza los datos que se han cambiado y si son válidos los actualiza en el servidor central.</td></tr>
<tr><td colspan="2"></td><td>9. Envía un mensaje de confirmación.</td></tr>
<tr><td colspan="3">Sección: Eliminar Usuario</td></tr>
<tr><td colspan="2"></td><td>3. Muestra una interfaz para que el actor realice la búsqueda del usuario que se desea</td></tr>
</table>

| | |
|---|---|
| | eliminar. |
| 4. Selecciona un criterio de búsqueda e introduce los datos para realizarla. | 5. Realiza la búsqueda y muestra los resultados. |
| 6. Selecciona el usuario y realiza la operación “Eliminar”. | 7. Elimina al usuario seleccionado. |
| | 8. Envía un mensaje de confirmación. |
| **Sección: Adicionar Usuario** | |
| | 3. Muestra la Interfaz para que el actor introduzca los datos del nuevo usuario. |
| 4. El actor introduce los datos del usuario y realiza la opción de aceptar. | 5. Verifica que el usuario no exista y actualiza la base de datos en el servidor central. |
| | 6. Envía un mensaje de confirmación. |
| **Sección: Buscar Usuario** | |
| | 3. Muestra la Interfaz para que el actor introduzca los datos usuario que desea buscar. |
| 4. El actor introduce los datos del usuario y realiza la opción Buscar. | 5. Muestra los resultados obtenidos de la búsqueda. |
| **Sección: Modificar Usuario** | |
| **Flujo alternativo 1** | |
| **Acción del actor** | **Respuesta del sistema** |
| | 5. En caso de que no existan resultados para la búsqueda se informa al actor y termina el caso de uso. |
| **Flujo alternativo 2** | |
| | 8. En caso de que no se seleccione ningún usuario o los datos introducidos no sean válidos se envía un mensaje de error. |
| **Sección: Eliminar Usuario** | |
| **Flujo alternativo 1** | |
| | 5. En caso de que no existan resultados para la búsqueda se informa al actor y termina el caso de uso. |
| **Flujo alternativo 2** | |

| | |
|---|---|
| | 7. En caso de que no se seleccione ningún usuario o que no se pueda eliminar, se envía un mensaje de error. |
| **Sección: Adicionar Usuario** | |
| **Flujo alternativo 1** | |
| | 5. En caso de existir el usuario, se envía un mensaje diciendo que usuario ya esta registrado y termina el caso de uso. |
| | |
| **Puntos de extensión.** | |
| | |

| **Caso de uso** | |
|---|---|
| CU2 | Gestionar Componentes |
| **Propósito** | Permitir Descargar o Adicionar componentes. |
| **Actores:** Usuario | |
| **Resumen:** Este caso de uso se inicia cuando el actor desea descargar o adicionar componentes, para descargar se puede buscar por un criterio de búsqueda el componente deseado y descargarlo y para adicionar basta con suministrar los datos necesarios. | |
| **Referencias** | RF2, CU3 |

| **Acción del actor** | **Respuesta del sistema** |
|---|---|
| 1. El actor envía la solicitud de:<br>• Adicionar Componente<br>• Descargar Componente | 2. En caso que se reciba:<br>• Adicionar Componente: ver sección “Adicionar Componente”<br>• Descargar Componente: ver sección “Descargar Componente” |
| **Sección: Adicionar Componente** | |
| | 3. Muestra una interfaz para que el actor introduzca los datos del componente que va a adicionar. |
| 4. Introduce los datos del componente y acepta. | 5. Si los datos son válidos envía un mensaje de confirmación al actor. |

| | |
|---|---|
| | 6. Envía un correo electrónico al especialista de calidad que debe revisar el componente. |
| **Sección: Descargar Componente** | |
| | 3. Muestra una interfaz para que el actor realice la búsqueda del componente que desea descargar. |
| 4. Selecciona un criterio de búsqueda para realizarla. | 5. Realiza la búsqueda y muestra los resultados. |
| 6. Selecciona el componente que desea y realiza la operación descargar. | 7. El sistema comienza la descarga del componente. |
| **Sección: Adicionar Componente** | |
| **Flujo alternativo 1** | |
| **Acción del actor** | **Respuesta del sistema** |
| | 5. En caso que los datos no sean válidos envía un mensaje con el error detectado. |
| **Sección: Descargar Componente** | |
| **Flujo alternativo 1** | |
| | 5. En caso de que no existan resultados para la búsqueda se informa al actor y termina el caso de uso. |
| **Flujo alternativo 2** | |
| | 7. En caso de que no se seleccione ningún componente se envía un mensaje de error. |
| | |
| **Puntos de extensión.** | |
| Línea 5: Busca el componente según criterio. Ver CU3. | |

| **Caso de uso** | |
|---|---|
| CU3 | Buscar Componentes |
| **Propósito** | Permitir Buscar componente solicitado |
| **Actores:** Usuario | |

| | |
|---|---|
| **Resumen:** Este caso de uso se inicia cuando el actor desea un componente en el repositorio. | |
| **Referencias** | RF3 |
| **Acción del actor** | **Respuesta del sistema** |
| 1. Envía la solicitud de búsqueda de componente. | 2. Muestra una interfaz para que el actor realice la búsqueda del componente |
| 3. Selecciona un criterio de búsqueda. | 4. Realiza la búsqueda y muestra los resultados. |
| | |
| **Puntos de extensión.** | |
| | |

| | |
|---|---|
| **Caso de uso** | |
| CU4 | Gestionar Calidad de Activo |
| **Propósito** | Permitir la publicación de los componentes que puedan ser reutilizables o eliminar los que no se puedan reutilizar. |
| **Actores: Responsable de Calidad** | |
| **Resumen:** El caso de uso se inicia cuando el responsable de calidad aprueba el componente y lo publica o elimina, en caso de que no cumpla los requisitos normados. | |
| **Referencias** | RF2, CU3 |
| **Acción del actor** | **Respuesta del sistema** |
| 1. El actor envía la solicitud de:<br>• Publicar Activo | 2. En caso que se reciba:<br>• Publicar Activo: ver sección “Publicar Activo” |
| **Sección: Publicar Activo** | |
| | 3. Muestra una interfaz con los activos que están sin revisar. |
| 4. El actor revisa el componente y realiza la operación publicar activo. | 5. Publica el activo. |
| | 6. Envía un correo electrónico al autor del activo notificando que su activo fue publicado. |

| **Flujo alternativo 1** | |
|---|---|
| **Acción del actor** | **Respuesta del sistema** |
| 4. El actor realiza la operación eliminar activo. | 5. El sistema elimina el activo y termina el CU. |
| | |
| **Puntos de extensión.** | |
| Línea 5: Busca el componente según criterio. Ver CU3. | |

| **Caso de uso** | | |
|---|---|---|
| CU5 | Gestionar Reportes | |
| **Propósito** | Permitir al actor acceder a reportes generados por la aplicación | |
| **Actores:** Usuario | | |
| **Resumen:** El caso de uso comienza cuando un actor solicita un reporte. El CU verifica los privilegios del actor y le muestra los reportes a los que tiene acceso. | | |
| **Referencias** | RF5, CU6, CU7, CU8 | |
| **Acción del actor** | | **Respuesta del sistema** |
| 1. Realiza la acción visualizar reportes | | 2. Verifica los privilegios del actor. |
| | | 3. Muestra una interfaz con los reportes que este puede consultar. |
| | | |
| **Puntos de extensión.** | | |
| | | |

| **Caso de uso** | | |
|---|---|---|
| CU6 | Gestionar Reportes de Calidad | |
| **Propósito** | Permitir al actor acceder a reportes generados por la aplicación | |
| **Actores:** Responsable de Calidad. | | |
| **Resumen:** El caso de uso comienza cuando un actor solicita un reporte y se muestran los resultados del mismo. | | |
| **Referencias** | RF5, CU6 | |
| **Acción del actor** | | **Respuesta del sistema** |
| 1. El actor selecciona el reporte que le | | 2. Hace la consulta a la base de datos y |

| | |
|---|---|
| interesa y realiza la acción mostrar | muestra el reporte solicitado. |
| | |
| **Puntos de extensión.** | |
| | |

| **Caso de uso** | |
|---|---|
| CU7 | Gestionar Reportes de Administración |
| **Propósito** | Permitir al actor acceder a reportes generados por la aplicación |
| **Actores:** Administrador. | |
| **Resumen:** El caso de uso comienza cuando un actor solicita un reporte y se muestran los resultados del mismo. | |
| **Referencias** | RF5, CU7 |

| **Acción del actor** | **Respuesta del sistema** |
|---|---|
| 1. El actor selecciona el reporte que le interesa y realiza la acción mostrar | 2. Hace la consulta a la base de datos y muestra el reporte solicitado. |
| | |
| **Puntos de extensión.** | |
| | |

| **Caso de uso** | |
|---|---|
| CU8 | Gestionar Reportes de Usuario |
| **Propósito** | Permitir al actor acceder a reportes generados por la aplicación |
| **Actores:** Usuario | |
| **Resumen:** El caso de uso comienza cuando un actor solicita un reporte y se muestran los resultados del mismo. | |
| **Referencias** | RF5, CU8 |

| **Acción del actor** | **Respuesta del sistema** |
|---|---|
| 1. El actor selecciona el reporte que le interesa y realiza la acción mostrar | 2. Hace la consulta a la base de datos y muestra el reporte solicitado. |

| |
|---|
| **Puntos de extensión.** |
| |

| **Caso de uso** | |
|---|---|
| CU9 | Mostrar Ranking |
| **Propósito** | Permitir que el actor pueda ver de una manera sencilla un ranking con diversos criterios. |
| **Actores:** Usuario | |
| **Resumen:** El caso de uso comienza cuando un actor accede a la página principal y en esta se muestra un ranking con criterios de interés. | |
| **Referencias** | RF6 |

| **Acción del actor** | **Respuesta del sistema** |
|---|---|
| 1. Accede a la página principal. | 2. Realiza una consulta a la Base de Datos. |
| | 3. Muestra la interfaz principal con el ranking actualizado y termina el caso de uso. |
| | |
| **Puntos de extensión.** | |
| | |

| **Caso de uso** | |
|---|---|
| CU10 | Petición de Mejora |
| **Propósito** | Permitir corregir cualquier error que se detecte en un activo de software. |
| **Actores:** Usuario | |
| **Resumen:** El caso de uso comienza cuando el actor hace una petición de mejora de un componente por alguna deficiencia detectada. | |
| **Referencias** | RF7 |

| **Acción del actor** | **Respuesta del sistema** |
|---|---|
| 1. Realiza la acción solicitud de mejora | 3. Muestra una interfaz para que el actor introduzca los datos del componente a mejorar. |
| 4. Introduce datos del componente y problema a resolver, después realiza la | 5. Si los datos son válidos envía un mensaje |

| | |
|---|---|
| acción aceptar | de confirmación al actor. |
| | 6. Envía un correo electrónico al autor del componente con la solicitud de mejora. |
| | 7. Notifica a los responsables de calidad del problema detectado. |
| **Flujo alternativo 1** | |
| **Acción del actor** | **Respuesta del sistema** |
| | 5. En caso que los datos no sean válidos envía un mensaje de error. |
| | |
| **Puntos de extensión.** | |
| | |

| **Caso de uso** | |
|---|---|
| CU11 | Autenticar Usuario |
| **Propósito** | Permitir el control de acceso a la biblioteca. |
| **Actores:** Usuario | |
| **Resumen:** El caso de uso comienza cuando el actor se autentifica para acceder a la biblioteca. | |
| **Referencias** | RF8 |
| **Acción del actor** | **Respuesta del sistema** |
| 1. Solicita la URL de la biblioteca. | 2. Muestra una interfaz para escribir nombre de usuario y contraseña. |
| 3. Introduce nombre de usuario y contraseña del dominio. | 4. Si los datos son válidos muestra la interfaz principal de la biblioteca. |
| **Flujo alternativo 1** | |
| **Acción del actor** | **Respuesta del sistema** |
| | 4. En caso que los datos no sean válidos muestra un mensaje de acceso denegado. |
| | |
| **Puntos de extensión.** | |
| | |

| **Caso de uso** | |
|---|---|
| CU12 | Importar Componentes |
| **Propósito** | Permitir importar componentes que se encuentren en otro base de datos. |
| **Actores:** Administrador | |

| | |
|---|---|
| **Resumen:** El caso de uso comienza cuando el actor selecciona los componentes que desea importar y posteriormente realiza la acción importar. | |
| **Referencias** RF9 | |
| **Acción del actor** | **Respuesta del sistema** |
| 1. Solicita importar componentes. | 2. Muestra una interfaz correspondiente a la acción realizada. |
| 3. Busca los componentes que desea importar. | |
| 4. Selecciona los componentes que desea importar y realiza la acción importar. | 5. Muestra una interfaz solicitando datos de los componentes que faciliten su búsqueda posterior por los usuarios de la biblioteca. |
| 6. Introduce datos de los componentes y realiza la acción aceptar. | 7. Se guardan los componentes en la BD. |
| **Flujo alternativo 1** | |
| **Acción del actor** | **Respuesta del sistema** |
| 6. No introduce datos de los componentes. | 7. Muestra un mensaje de error y termina el caso de uso. |
| | |
| **Puntos de extensión.** | |
| | |

### 3.5 Conclusiones.

En el capítulo se explican las dos propuestas para la implantación de una biblioteca de componentes de software reutilizables, una de forma inmediata, utilizando la herramienta Gforge como experimento en uno de los polos productivos de la UCI, y la otra, que se desarrolle a la par, una biblioteca adaptada al proceso productivo de la Universidad, con recursos tecnológicos y humanos propios. Se mencionan también las acciones organizativas necesarias para la puesta en práctica de las dos variantes.

Además se exponen los requisitos funcionales y no funcionales básicos que debe tener la biblioteca que se desarrollará como propuesta idónea para la UCI. Se realiza el Diagrama de Casos de Uso del Sistema y se describen los Casos de Uso que intervienen.

## Conclusiones

Después de analizar el proceso productivo de la UCI, las características de una biblioteca de componentes de software reutilizables que se adapte al mismo y dando respuesta a las preguntas científicas planteadas al inicio del trabajo, se arriba a las siguientes conclusiones:

1. La reutilización de componentes de software es posible en la UCI, si se tienen en cuenta todas las condiciones de la Universidad para organizarlo y potenciarlo. Pero más allá de ser posible, es necesaria, como una de las vías para optimizar el proceso de producción en la Universidad y lograr convertirla en una industria.
2. Una biblioteca de componentes reutilizables, es una aplicación web cliente servidor, donde se integran un grupo de herramientas, como son los repositorios de componentes, control de versiones, foros de discusión, comunicación mediante listas de distribución de correos, gestión de monitoreo de la planificación de las tareas, entre otras herramientas, que pueden incluirse en la biblioteca, teniendo en cuenta las características del entorno en el que de implante.
3. Una biblioteca de componentes reutilizables adaptada al modelo UCI, serían repositorios donde se guardan o conservan todos aquellos componentes o activos, que participan en el ciclo de vida de un software, que permitan el trabajo colaborativo de equipos de desarrollo localizados en diferentes lugares geográficos, utilizando las facilidades que ofrecen las Intranet e Internet, de manera tal que los componentes o activos puedan ser aportados, utilizados o consultados.
4. Las posibles soluciones para la implantación de una biblioteca de componentes de software reutilizables en la UCI son: la implementación de una experiencia de forma inmediata pero no permanente con el Gforge, en uno de los polos productivos de la Universidad y desarrollar a la par, una biblioteca que se adapte a las características del proceso productivo de la UCI, utilizando recursos tecnológicos y humanos propios.

## Recomendaciones

Se recomienda para impulsar la reutilización de componentes de software:

1. Alcanzar la representación estándar de los activos, mediante el establecimiento de principios o premisas para el uso de las arquitecturas, plataformas, lenguajes de programación y herramientas para el desarrollo de software en la UCI.
2. Establecer normas para la documentación que acompaña los componentes incluidos en la biblioteca, por parte de la Dirección de Calidad de la UCI.
3. Incluir la reutilización de componentes de software como una nueva disciplina dentro de la Ingeniería del Software, de tal forma que desde de la Formación de los estudiantes se contribuya con el tema de la reutilización, uno de los objetivos esenciales que se comienza a impulsar paulatinamente por parte del dirección de Producción de la UCI.
4. Organizar dentro del grupo de investigación “Arquitecturas de Software”, líneas que estén relacionadas con la reutilización de software.
5. Realizar un análisis profundo de la herramienta de colaboración Gforge, como punto de partida para el desarrollo de una biblioteca de componentes de software reutilizables en la UCI.
6. Implementar la experiencia con el Gforge en uno de los Polos Productivos de la Universidad.
7. Crear y organizar un grupo de proyecto, que sea responsable de comenzar el desarrollo de la biblioteca de componentes de software reutilizables de la UCI.
8. Importar a la biblioteca desarrollada por la UCI los componentes posibles a reutilizar, que se encuentran en repositorios ya existentes tanto dentro como fuera de la Universidad.

## Bibliografía

(1) KRUEGER, Charles W. *Software Reuse, ACM Computing Surveys*. Volumen 24 no. 2, Junio de 1992.

(2) PRIETO-DÍAZ, Ruben. *Implementing Faceted Classification for Software Reuse, Comunications of the ACM.* Volumen 34 no. 5, Mayo de 1991.

(3) PACIFICO, C. *Conceptos de Reusabilidad de Software*. III Jornadas Internacionales de Administración e Informática, Universidad Nacional de Entre Ríos, Facultad de Ciencias de la Administración, 2005. p.

(4) Diccionario Microsoft® Encarta® 2007. © 1993-2006 Microsoft Corporation.

(5) UNESCO. *Manifiesto de la Biblioteca Pública de la UNESCO*. Nov. de 1994. [En línea][Citado el: 5 de abril de 2007]

http://www.nl.gob.mx/creb/boletin/unescobp.htm.

(6) Wikipedia. *La Enciclopedia Libre*. [En línea][Citado el: 10 de abril de 2007]

http://es.wikipedia.org/wiki/Biblioteca_digital.

(7) SAMETINGER, J. Sofware *Engineering with Reusable Components.* Part II: Software Components. Berlin: Springer-Verlag. 1997.

(8) MEYER, B. *Construcción de Software Orientado a Objetos*. Segunda Edición. Madrid, 1999. p. 84-8322-040-7.

(9) SODHI, J. and SODHI, P. *Software reuse: Domain analysis and design processes*. New York, McGraw-Hill, 1999. p. 0070579237.

(10) SOMMERVILLE, Ian. *Software Engineering.* Octava Edición. Pearson Education, 2006. p.

(11) GHEZZI, Carlo; JAZAYERI, Mehdi, and MANDRIOLI, Dino. *Fundamentals of Software Engineering*. Prentice Hall. 1991.

(12) CLEMENTS, P. and NORTHROP, L. *Software Products Line: Practices and Patterns.* Addison Wesley, 2002.

(13) GOMMA, H. *Designing Software Product Lines with UML: From Use cases to pattern-based Software Architectures.* Addison Wesley. 2004.

(14) KRUEGER, Charles W. *Introduction to Software Products Lines.* [On-line] http://www.softwareproductline.com. 2006.

(15) BROWN, A. W. *Large-Scale, Component-Based Development.* Prentice Hall PTR. 2000.

(16) Microsoft Architects Journal. *The Case for Software Factories*. [On-line] http://msdn.microsoft.com/architecture/journ/default.aspx?pull=/library/en-us/dnmaj/html/aj3softfac.asp.

(17) GRISS, M.L. and WOSER, M. *Making Reuse Work at Hewlett-Packard*. IEEE Software. January 1995.

(18) PALACIO, J. G*estión y procesos en empresas de software.* Volumen 19, 2005.

(19) OKTABA, Hanna. Modelo de Procesos para la Industria de Software, MoProSoft, Por Niveles de Capacidad de Procesos. Versión 1.3, Agosto 2005.

(20) MACHADO, Alejandro Gabriel. *Presentación: La producción en la UCI, "Mi proyecto, EL PROYECTO FUTURO"*. Claustro de Profesores de la UCI. Abril 2007.

## Glosario de Términos

[1] **Crisis del software:** Creciente desbalance entre la oferta y la demanda, que ocasionan deficiencias en la productividad del desarrollo de software, escasez en la oferta de profesionales del área y poco desarrollo en las metodologías y técnicas para la producción.

[2] **Herramientas CASE:** Sigla en inglés que significa en español: "Ingeniería Asistida por Computadora", se le da este nombre a herramientas que buscan de automatizar los aspectos clave de todo el proceso de desarrollo de un sistema informático, desde el principio hasta el final.

[3] **Retorno de la Inversión (ROI):** Return of investmentes es el beneficio que se obtiene por cada unidad monetaria invertida en tecnología durante un período de tiempo. Suele utilizarse para analizar la viabilidad de un proyecto y medir su éxito.

ROI= (Beneficios/Costes) x100.

[4] **XSDs:** Una más nueva lengua del esquema de XML, descrita por el W3 C como el sucesor de DTDs, es XML Schema, o referido más informal en términos de initialism para los casos de XML Schema, XSD (XML Schema Definition). XSDs es más de gran alcance lejano que DTDs en describir idiomas de XML. Utilizan un sistema datatyping rico, permiten apremios más detallados en la estructura lógica de un documento de XML, y se requieren para ser procesadas en un marco más robusto de la validación.

**[5] GNU GPL (General Public License o licencia pública general):** es una licencia creada por la Free Software Foundation a mediados de los 80, y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

**[6] Multihilo:** Un hilo de ejecución, en sistemas operativos, es una característica que permite a una aplicación realizar varias tareas simultáneamente. Los distintos hilos de ejecución comparten una serie de recursos tales como el espacio de memoria, los archivos abiertos, situación de autentificación, etc. Esta técnica permite simplificar el diseño de una aplicación que debe llevar a cabo distintas funciones simultáneamente.

**[7] Transacción:** es una unidad lógica de trabajo. Una serie de actualizaciones que dejan otra vez la Base de Datos en un estado coherente.

**[8] ROLLBACK:** ocurre debido a algún fallo en la transacción, deshace la misma y el sistema mantiene un archivo con el estado de la base de datos antes del inicio de la transacción.

[9] **Service-Oriented Architecture (SOA):** Es un modelo de componente que interrelaciona unidades funcionales diferentes de una aplicación, denominado servicios, a través de interfaces y contratos bien definidos entre estos servicios. La interfaz se define de una manera neutral que debe ser independiente de la plataforma de hardware, del sistema operativo y del lenguaje de programación en los que se implemente el servicio. Esto permite que los servicios, construidos en una variedad de tales sistemas, interactúen entre sí de una manera uniforme y universal.

[10] **Parche:** En informática, un parche es una actualización de un programa usado para solucionar problemas o la usabilidad de una versión previa de la aplicación. Esto puede incluir cualquier programa desde un procesador de texto, videojuego hasta un sistema operativo. El parche se puede aplicar a un binario ejecutable o al código fuente de un programa.

[11] **Consejo Técnico de Producción:** Es presidido por el Director Técnico de Producción de la Infraestructura Productiva de la UCI. Tiene como principal propósito la toma de decisiones, en cuestiones tecnológicas y de arquitecturas, para el desarrollo de los proyectos productivos de las 10 Facultades. Sus competencias son:

- Garantiza la armonía, equilibrio e integración de las 10 Facultades, en las cuestiones técnicas, relacionadas con el desarrollo de software.
- Aprueba las arquitecturas de software de los proyectos productivos.
- Aprueba las solicitudes de adquisición de tecnología y licencias de software.
- Aprueba y asigna proyectos de investigación aplicada, relacionados con las líneas de investigación en las cuestiones técnicas de la producción.
- Estudia los programas de las asignaturas de la especialidad de los diferentes planes de estudio, y propone cambios en dependencia de las necesidades y objetivos de la producción.
- Propone la incorporación de nuevos cursos optativos y asignaturas al segundo perfil, según las líneas de producción de las facultades.
- Establece las normas para la acreditación de competencias en los roles de arquitecto de software, programador, gestor de configuración, administrador de bases de dato, entre otros.

[12] **Genexus:** Es el producto insignia creado por ARTech del Uruguay. Poderosa herramienta para el diseño y desarrollo de software multiplataforma. Permite el desarrollo incremental de aplicaciones críticas de negocio de forma independiente de la plataforma. Genera el 100% de

la aplicación. Basado en los requerimientos de los usuarios realiza el mantenimiento automático de la base de datos y del código de la aplicación, sin necesidad de programar.

[13] **MoProSoft:** es un Modelo de Procesos para la Industria de Software en México que fomenta la estandarización de su operación a través de la incorporación de las mejores prácticas en gestión e ingeniería de software. La adopción del modelo permitirá elevar la capacidad de las organizaciones para ofrecer servicios con calidad y alcanzar niveles internacionales de competitividad.