Una herramienta de apoyo a la gestión del proceso de desarrollo de software

Enrique P. Latorres
latorres@ort.edu.uy; elatorres@mtop.gub.uy
Ministerio de Transporte
y Obras Públicas
de Uruguay
Universidad ORT Uruguay

Pedro Salvetto
salvetto@ort.edu.uy
Laboratorio de Investigación de
Sistemas de Información
Universidad ORT Uruguay

Uruguay Larre Borges
ularre@adinet.com.uy
Consultor independiente

Juan C. Nogueira
nogueira@ort.edu.uy
Laboratorio de Investigación de
Sistemas de Información
Universidad ORT Uruguay

Resumen

En muchos centros de cómputo de la región no se aplican modelos de proceso de desarrollo de software. Esto tiene varios orígenes, principalmente la falta de recursos y capacitación en las mejores prácticas de la industria. En general, no está extendido el conocimiento y el uso de metodologías de gestión de riesgos, la mayoría de las cuales no son estructuradas y requieren de gerentes con experiencia, un recurso siempre escaso. El intercambio de información y experiencia se hace difícil y sólo es posible compararnos con organizaciones de otros contextos sociales, tecnológicos, económicos y culturales. Como resultado, las opiniones están basadas más en percepciones personales que en información objetiva. Esta investigación propone proveer una herramienta de bajo costo junto con un modelo de implantación, que aporte valor a toda la comunidad informática, y permita implantar y controlar un proceso de desarrollo de software, ágil y flexible, orientado a grupos de desarrollo pequeños y medianos, que desarrollan utilizando GeneXus[1], con control de calidad y proceso de mejora continua, que permita la planificación de la capacitación continua del equipo, controlar los costos de calidad y brindar automáticamente indicadores de gestión de proyectos, poniendo énfasis en el seguimiento del riesgo del proyecto.

Palabras clave: Gestión de proceso de desarrollo de software, Gestión de calidad, Gestión de Riesgo, Métricas automáticas, Informes automáticos, Capacitación, Mejora personal del proceso, Formación para tercerización, Indicadores de gestión, GeneXus.

1 Introducción y objetivo principal

Muchas organizaciones de Uruguay, debido a razones culturales y a las crisis económicas por las que ha transitado la región, hacen hincapié en inversiones en tecnología orientadas principalmente hacia la incorporación de aquellas que mejoren la ecuación de costos fijos y variables de la producción o los servicios que brindan, en desmedro, de los sistemas de información en general. Se descuidan servicios internos de apoyo técnico y de gestión como las inversiones en mejorar los procesos de desarrollo de software. Por esta razón el área de sistemas, en muchas organizaciones, dispone de recursos muy limitados para la mejora de sus procesos y es común que no estén en condiciones de diseñar los suyos propios debido a las permanentes urgencias que deben atender diariamente. Según estudios recientes [2] un 50% de estas organizaciones - en Uruguay - dispone de personal destinado al aseguramiento de la calidad, pero sólo el 26% indica la cantidad de personal asignado

para esa tarea. Sólo el 4% usa un proceso definido para el desarrollo de software. El 20% cuenta con alguna certificación en calidad en la empresa pero solo el 4% cuenta con certificaciones específicas del proceso de desarrollo de software y el 24% de los profesionales no tienen formación en modelos de procesos de desarrollo. Esto, en mayor medida, es debido a que las empresas han preferido la capacitación en herramientas tecnológicas y no en modelos de gestión.

Los problemas con el proceso de desarrollo en el Departamento de Informática del Despacho de Secretaría y Oficinas Dependientes del Ministerio de Transporte y Obras Públicas (MTOP) de Uruguay motivaron varios estudios y una propuesta de "Proceso de Desarrollo de Software" [3] que da origen a esta investigación, la cual toma mayor empuje al integrarse con los trabajos de Pedro Salvetto en gestión de riesgo en proyectos de desarrollo de software, y por último al conformarse con el Laboratorio de Investigación de Sistemas de Información (LISI) de la Universidad ORT Uruguay un equipo de desarrollo con seis estudiantes de la licenciatura en sistemas de información de Universidad ORT del Uruguay como proyecto de grado. El objetivo de este proyecto es proveer una herramienta de bajo costo junto con una recomendación de proceso, que permita implantar y controlar un proceso de desarrollo de software, ágil y flexible, orientado a grupos de desarrollo pequeños y medianos, que desarrollan con la herramienta GeneXus, con control de calidad y proceso de mejora continua, que permita la planificación de la capacitación continua del equipo de trabajo, controlar los costos derivados de la falta de calidad y brindar –automáticamente - indicadores de gestión de proyectos obtenidos a partir de métricas tempranas que permitan un monitoreo continuo del riesgo de fracaso del proyecto, y fomente el intercambio de información objetiva sobre los procesos de desarrollo de software.

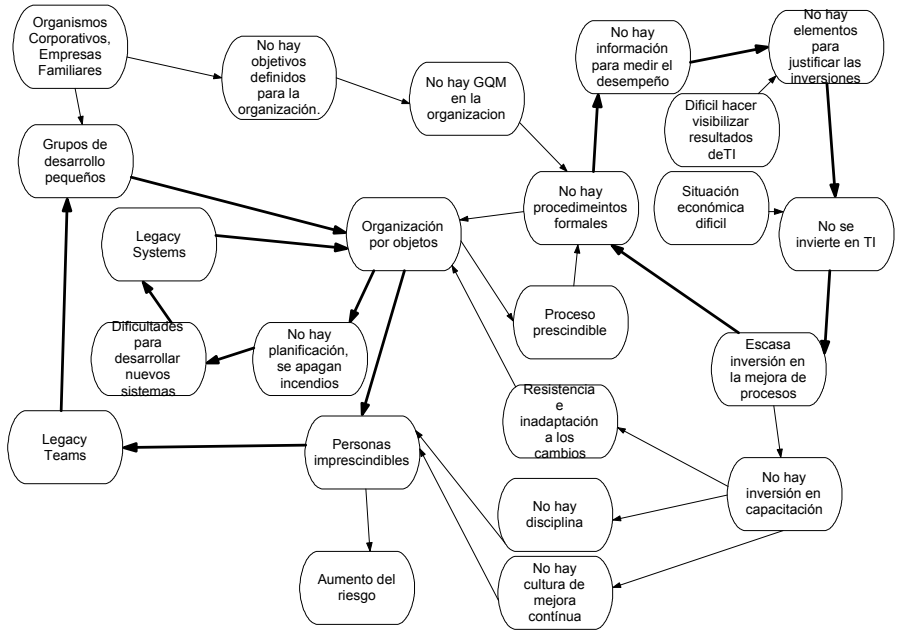
2 Objetivos generales del proyecto

Un objetivo general es contribuir a la mejora de la industria del software en el país y la región 1) facilitando la adopción de las mejores prácticas de la industria, 2) promoviendo el intercambio de información objetiva sobre resultados y soluciones a problemas de TI adaptados a la realidad de nuestro país y la región y 3) brindando un modelo estructurado de evaluación continua de riesgos integrado al IDE.

2.1 Adopción de mejores prácticas

La aplicación de metodologías para el proceso de desarrollo de software se verifica en aproximadamente el 4% de los centros de desarrollo [2], estas estadísticas están por debajo de similares en la región En Uruguay, el 80% de estas organizaciones son centros de desarrollo de software con grupos pequeños de menos de 20 personas, y el 40% con grupos de desarrollo compuestos por 1 a 4 integrantes [4]. Por lo tanto, estamos en un medio donde a la mayoría de las organizaciones que desarrollan software se las puede catalogar de pequeñas o medianas empresas. Muchas de las empresas que disponen de un grupo de desarrollo son organismos corporativos o empresas pequeñas y medianas, que por razones históricas o por necesidades de mercado o área de actividad mantienen sus sistemas, ya que sistemas de plaza son o muy caros o no cumplen fácilmente con las necesidades de la organización. Frecuentemente estas organizaciones, no tienen objetivos claramente definidos para el área de TI. Sin objetivos explícitos no es posible derivar indicadores de desempeño y elementos de apoyo a la toma de decisiones; tampoco indicadores estratégicos, que estén alineados con las políticas y las necesidades de la organización. La falta de esta información hace difícil la aplicación de GQM (Goal Question Metric) [5][6] y por lo tanto el desarrollo de procesos para el control de los parámetros que se definirían a partir de estos. En muchos casos, la organización toda carece de procedimientos definidos. La Figura 1 presenta algunos de estos problemas y sus relaciones. A continuación se comentan algunos de los más importantes círculos viciosos que se identifican. La inexistencia de procedimientos pone en evidencia un desconocimiento de su valor como herramienta para la coordinación y el control automatizados de las actividades de la organización.

Figura 1: Diagrama conceptual de problemas más frecuentes en la gestión de TI y sus relaciones.



La pobre valoración de estas herramientas en la organización, hace que sea difícil obtener apoyo para la incorporación de estos procedimientos en un área de "servicios", tal como se acostumbra a clasificar al área de tecnologías de la información, ya que normalmente no está vinculada a los sistemas de información estratégicos, sino sólo a los sistemas de gestión administrativa contable y de producción. En esas condiciones es complicado obtener apoyo e inversión para la implantación de procedimientos formales en el área de TI en muchas de estas organizaciones. Por ser el área de TI un área de servicios, no se la ve como fundamental dentro de la estructura productiva de la organización, y tiene muy baja prioridad dentro de las disponibilidades de inversión. Por otra parte al no concitar la atención de los directivos, y no disponer de mediciones objetivas en relación a los resultados de la organización, las actividades del área de TI son poco visibles para el resto de esta. En general son percibidas cuando no funcionan bien, de lo contrario son tan solo parte de la infraestructura. La carencia de elementos para justificar -dentro de la organización- la inversión en tecnologías de procesos y TI, se traduce en que no hay inversión en TI ni mejora de procesos, y nos introduce en un círculo vicioso (Figura 1). Este a su vez afecta la inversión en capacitación, que – de existir – se orienta hacia herramientas operativas más que a apoyar nuevos procesos, o al soporte de mejora continua. Se cambiará de un compilador a otro más moderno, o al lenguaje de moda, pero es menos probable que se invierta en herramientas de control de versiones, auditoria, pruebas, o en soporte al proceso de desarrollo. La falta de capacitación en metodologías resulta en personal menos dispuesto a los cambios y a reorganizarse, trabajo desorganizado y carente de disciplina y falta de cultura de mejora continua.

Las organizaciones con grupos de TI pequeños, que como ya vimos son la amplia mayoría en el Uruguay, muchas veces llevan a la organización del trabajo de desarrollo por objetos [7] donde cada persona o pequeño grupo se especializa en una aplicación, informalmente coordinado con otras aplicaciones y personas del equipo, en lugar de hacerlo por actividades o tareas (jerárquica, más habitual) o de acuerdo las mejores prácticas de la industria en que los equipos de desarrollo modernos se organizan por proyectos (o en forma matricial). Esta organización *ad hoc* por objetos refuerza la inamovilidad de las personas. Aparecen los expertos en tal o cual sistema "que solo él o ella saben como hacer funcionar". Las personas se tornan imprescindibles, y hay problemas si se enferman o simplemente se van de vacaciones. Esto fomenta la formación de los "Legacy Teams", equipos de técnicos atrincherados en tecnologías potencialmente perimidas y que no se arriesgan a proponer mejoras pues los resultados no se harán visibles a la organización, les aumentan el esfuerzo y en algunos casos extremos puede que consideren que podría disminuir su "status quo" de "gurús"

de su pequeño dominio de conocimiento. Sin procedimientos bien definidos, la única forma de contrarrestarlo es incrementando la cantidad de personas dentro del equipo; ¡Posiblemente formando más "Legacy Teams"! La ausencia de procedimientos lleva a una carencia de planificación adecuada. Los equipos trabajan en forma reactiva y no hay tiempo disponible para el desarrollo de nuevas aplicaciones y menos de herramientas que apoyen la implantación de metodologías, ni para capacitarse y familiarizarse con nuevas tecnologías. Este es el caso de organizaciones en el nivel 1 CMM [17]. Se comienza a depender fuertemente de los "Legacy Systems". Finalmente los "Legacy Teams" y los "Legacy Systems" se retroalimentan en un doble círculo vicioso. Estos círculos viciosos - a los que se enfrentan los encargados de las áreas de TI de estas organizaciones - aumentan el riesgo en los sistemas de TI, y son muy difíciles de enfrentar con recursos escasos y sin apoyo de la organización. Los gerentes de TI tienen entonces dificultades para salir de ellos y deben atacar varias causas al mismo tiempo a efectos de poder erradicarlos. Una de las soluciones es que las organizaciones se percaten de su error y comprendan la relevancia y utilidad de las TI para la gestión moderna de las organizaciones. Pero es poco probable que el personal de TI pueda, por sí solo, hacer demasiado para conseguirlo. Sin embargo, se puede tratar de cortar esos círculos viciosos con alguna ayuda externa y sin necesidad de grandes inversiones. Esperamos que este enfoque del área de TI pueda ser tomado como ejemplo para incorporar este espíritu y mentalidad de mejora continua al resto de la organización.

La herramienta que estamos desarrollando puede ayudar a romper con estos círculos viciosos, aportando una base para incorporar procedimientos, capacitación en mejora de los procesos, infraestructura para la mejora continua, que se adapte a la idiosincrasia de los equipos de TI y con un costo de implantación muy bajo, o que eventualmente pueda ser auto-implantado. Además esta herramienta puede ser útil a los consultores apoyando la transformación de las organizaciones que asesoran. La mejora de la visibilidad de las actividades del área de TI sumado a otros cambios de actitud [8] puede ayudar a dar la relevancia que corresponde a las actividades de TI dentro de las organizaciones modernas de la región.

2.2 Promover el intercambio de información objetiva sobre resultados y soluciones a problemas de TI, adaptados a la realidad de nuestro país y la región

Más allá de la falta de procedimientos definidos, y en parte como consecuencia de esta, hay una carencia de información sobre el desempeño y los resultados de procesos de desarrollo en la región. Aspectos relacionados con idiosincrasia y legislación hacen que no haya costumbre de informar al público sobre los indicadores de desempeño de las organizaciones. La información contable es secreta, los resultados son reservados, los informes sobre la calidad de los procesos de producción y su producto se manejan con acceso restringido, y el desempeño de la organización con respecto a su mercado no se encuentra disponible y no es interés de las organizaciones darlo a conocer. Estas políticas de restricción de la información se propagan a lo largo de la organización, y se reflejan en la forma de actuar y presentarse de las áreas de TI. La información pública sobre el desempeño de los grupos de TI de la región es escasa. Como contraste existe una encuesta en Brasil muy completa, realizada bianualmente por la Secretaría de Política Informática del Ministerio de Ciencia y Tecnología desde 1995 sobre la calidad en el sector de software [9]. Si algún gerente de TI quiere hacer benchmarking, cuenta sólo con información disponible de grupos, para nada similares, de USA, Europa, o incluso en Asia. Esto restringe severamente también el outsourcing [39] debido a la dificultad de determinar la relación costo/beneficio. Estas políticas son poco convenientes para las áreas de TI, pues las hacen aún más invisibles dentro de la organización y aumentan la incertidumbre de la alta gerencia sobre su desempeño además de restringir las oportunidades de mejora por medio del intercambio de experiencias entre expertos. Uno de nuestros intereses es abrir instancias y mecanismos para el intercambio de información relativa a desempeño, estimación de costos y de tiempos, a soluciones para la mejora continua y patrones de desarrollo, a efectos de poder definir

metodologías adecuadas a las condiciones, tamaños e idiosincrasia de los equipos de TI de la región. Para ello este sistema proveerá de opciones especiales para generar informes de resultados diseñados para ser compartidos con otras organizaciones, cuidando la información que pueda ser considerada confidencial.

2.3 Proveer un modelo estructurado de evaluación continua de riesgos integrado al IDE

La gestión y seguimiento del riesgo de los proyectos se basa en técnicas informales que requieren de gerentes de proyecto con gran experiencia. Los gerentes con experiencia son un recurso escaso y su desempeño pasado no es garantía éxito en el futuro. Además al ser técnicas informales es posible que diferentes evaluadores lleguen a diferentes conclusiones respecto del mismo escenario. El sistema proveerá un modelo estructurado y automático que brinde una medida objetiva –y por lo tanto comparable- del riesgo (ver 3.5).

3 Requisitos

El prototipo se desarrolló en GeneXus, estando previsto generalizarlo a MIS desarrollados en torno a bases de datos relacionales con un proceso de desarrollo evolutivo y en el cual se tomen las métricas asociadas a PSP.

3.1 Métricas e implantación

La propuesta de iniciar la aplicación de métricas dentro del grupo de TI tiene efectos disímiles dentro de los profesionales del equipo. Muchos comprenden la conveniencia de su aplicación para el proceso de desarrollo y las necesidades de responder a sus clientes con precisión sobre los costos y tiempos involucrados en el desarrollo de un proyecto. Otros desconfían y suponen que se aplicarán para controlar y evaluar las productividades individuales, con un objetivo coercitivo. Además, la aplicación de métricas que los propios integrantes del equipo deben registrar, representa un trabajo extra que puede afectar su desempeño al dedicar tiempo y concentración en llenar formularios que no aportan beneficios directos e inmediatos a su trabajo. Estos temores hacen que la captura de métricas donde el propio integrante debe registrar sus actividades pierda validez, ya que o se considera como burocracia innecesaria que obstruye la verdadera creatividad y productividad, o los integrantes embellecen los resultados, o cambian el comportamiento para adaptarse en forma exagerada a los parámetros con los que son evaluados. Varios de estos aspectos son considerados en la literatura y en particular la experiencia de HP [10] constituye un ejemplo interesante. La implantación de un sistema de métricas no es algo trivial, y exige una planificación concienzuda y una campaña de capacitación, formación e información. Una opción más interesante es hacer que las métricas sean registradas de la forma lo más automática posible. Aún así se presenta el problema de correlacionar las definiciones adoptadas para los requisitos con el esfuerzo necesario para implementarlos. También la identificación de código reutilizado o creado y luego eliminado, ya sea por errores o por cambios en los requisitos, es una métrica importante que hace al desempeño del equipo de trabajo.

El ambiente GeneXus de desarrollo de sistemas, basado en una abstracción de alto nivel conceptual, permite una gran productividad y provee varias facilidades que permiten automatizar con relativamente poco esfuerzo la mayor parte de las métricas de productividad. Por ejemplo, firmas con fecha y hora de modificación de cada uno de los objetos definidos en el sistema y entidades de la especificación formal. Además provee un "OleDb provider" que publica la especificación de la base de conocimiento. GeneXus es una herramienta de especificación formal de sistemas de información, que, una vez especificado un sistema, permite generar código para diversas plataformas y lenguajes. A partir de esto nos proponemos identificar métricas que midan la complejidad conceptual de las bases de conocimiento y guarden relación con el esfuerzo para desarrollarlas.

Sobre la incorrecta utilización de las métricas

El uso de las métricas de desempeño como forma de evaluación del personal puede traer aparejado serios problemas que afecten al proyecto y a la aplicación del programa de métricas. Del éxito o fracaso de este programa de métricas depende el resto de la implantación de estas metodologías ya que sin métricas toda otra acción es estéril. Los efectos negativos que esto puede traer aparejado en el comportamiento del equipo, pueden hacer perder la validez de las métricas por diferentes causas y no aportan a una evaluación adecuada del personal [10]. Se podrá además evaluar en forma independiente a los desarrolladores sobre la base de las métricas disponibles en el sistema, aunque se sugiere evitar este uso. Otros modelos muestran que para ciertas situaciones la aplicación de métricas a los individuos puede significar criterios importantes de evaluación de personal. Pero la visión presente en el proyecto es que principalmente aporta elementos para la mejora personal de cada uno de los participantes del equipo, por lo que empleamos métricas tomadas de PSP [11]. En general fomentar el espíritu de equipo tiene más relevancia por la multiplicación de la productividad que brinda la sinergia y la coordinación [12] comparado contra la optimización de los elementos individuales del sistema [13]. Muchos factores que afectan el desempeño del equipo están fuera del alcance de las métricas de productividad que pueda soportar un sistema y es responsabilidad del encargado del proyecto evaluar el desempeño sobre este cúmulo de imponderables [14][15][16]. Es importante que si se consideran estos elementos para la decisión sobre la continuidad o no de un integrante del equipo, no se utilice la justificación de las métricas del sistema por los efectos negativos que en el resto del equipo y en el sistema de métricas puede causar.

3.2 Ciclo de vida y Fases

Dadas las condiciones de los centros de tecnología de la información en las organizaciones del Uruguay [4], las metodologías ágiles han demostrado ser las más convenientes para aplicar a los procesos de desarrollo. La aplicación de otras metodologías de proceso de desarrollo son poco adecuadas para los pequeños grupos de TI que obligan a una enorme multiplicidad de roles y aumentan los procedimientos y registros burocráticos del equipo. Por otra parte metodologías monumentales, como CMM [17] y similares, sólo pueden ser aplicados en grupos de TI de dimensiones importantes de los que existen muy pocos en el país. Buscamos entonces un proceso y ciclo de vida sencillo y flexible, que se pueda adaptar a las necesidades de cada equipo de TI y cada proyecto según su tamaño, experiencia y complejidad. Las definiciones de etapas se fundamentaron en los ciclos evolutivos basados en casos de uso, usando como modelo el propuesto por Craig Larman [18], el cual se adapta bien al modelo XP [19], agregando especificaciones y métricas derivadas de PSP [11] y TSP [20], aunque la secuencia de pasos y ciclos estarán librados a las características de cada proyecto. Se asume que los roles y/o actividades serán múltiples para cada individuo, dado que debe ser posible la cobertura de cualquier integrante del equipo. El gerente del proyecto podrá diseñar un ciclo determinado para el proyecto y asignar un conjunto de técnicos al proyecto.

De XP se extrajeron varias ideas. En particular el desarrollo de a pares promueve el intercambio de información y el aprendizaje entre los integrantes del equipo y la revisión continua del código generado. Por otra parte, al evaluarse parejas, se resuelven algunos de los problemas de evaluación del desempeño personal y los efectos negativos que esto puede traer aparejado en el comportamiento del equipo. Además se promueve la sinergia, la cooperación, el intercambio de ideas y conocimiento y se disminuye el riesgo al haber mayor cantidad de personas participando en cada módulo.

Uno de los objetivos del sistema de gestión del proceso de desarrollo propuesto es que el grupo de trabajo esté en condiciones de obtener un sistema funcionando en poco tiempo. Para esto se debe disponer de un ciclo de vida evolutivo que permita obtener funcionalidad en etapas tempranas sin comprometer el diseño y la funcionalidad futura [18]. Se aprovecha algunas de las cualidades del ambiente de desarrollo elegido. Esto significa, entre otras cosas, el ingreso temprano de las visiones

de los usuarios. Una ventaja es que el ambiente de desarrollo es capaz de reflejar estas visiones de usuario así como sus cambios en la base de datos en forma automática. Con esto se promueve el desarrollo con "refactoring", y el objetivo de obtener en etapas tempranas el sistema con alguna funcionalidad disponible, completamente alineado con la filosofía XP [19].

A partir del diseño temprano de las visiones de usuario se puede implementar en cada ciclo de desarrollo nuevas funcionalidades basadas en las prioridades evaluadas en el análisis de requisitos [18]. Asociados a GeneXus existen estándares de codificación que serán incrementados con las directivas que el grupo de desarrollo defina a partir de su experiencia y de las soluciones que plantee junto con una adecuada gestión del conocimiento sobre el proceso de desarrollo. El sistema permite al desarrollador tomar la métrica de tamaño del sistema en desarrollo antes de comenzar y al terminar. De la diferencia entre el código antes y el código después, más el registro del las horas involucradas en cada etapa y módulo, junto con la indicación de los requisitos involucrados, el sistema puede obtener en forma automática el nivel de reutilización y de producción por unidad de tiempo y etapa. Los ingresos que el desarrollador debe hacer, serán reducidos al mínimo.

3.3 Gestión de Costos

Se dispondrá de elementos adecuados para estimar la cantidad de esfuerzo necesario para minimizar el costo de calidad y la cantidad de defectos. Para ello se asigna a las actividades tipos de costo [21]. Así las diferentes actividades se asocian a costos de prevención, costos de evaluación, costos de defectos internos, y costos de defectos externos [22]. Todos estos costos están evaluados en horas/hombre. Esto permite gestionar los costos de calidad y comprender cuál es la distancia a recorrer para obtener el 100% de calidad [23]. Las métricas registran las actividades de los integrantes del equipo en horas/hombre que se pueden valorar monetariamente. Si a este costo monetario se le agregan otros costos derivados del proyecto, se puede mantener el índice de desempeño de costo (Cost Performance Index) sin gran esfuerzo. Si se ingresan las planificaciones/estimaciones de tiempos del proyecto se puede derivar el índice de desempeño en tiempos (Schedule Performance Index) y el valor ganado (Earned Value), entre otros indicadores de costos y tiempos.

3.4 Ingeniería de Requisitos

La variación de requisitos es un elemento importante para la elaboración de indicadores de riesgo de fracaso en proyectos de software [26]. El sistema registrará los requisitos, referencias cruzadas para análisis de impacto y correlación con módulos y actividades. Debido a que la variación de los requisitos guarda relación con la estabilidad de un sistema en desarrollo, se registrarán sus cambios. Se diferenciarán los requisitos ya envejecidos y estables de los requisitos que son recién ingresados debido a que el proceso de identificación de requisitos es un proceso de aproximaciones sucesivas, donde la información entre el usuario/entrevistado y el analista de requisitos se transmite por un proceso de aprendizaje y verificación. Los requisitos deben llevarse a "contratos" al final de cada etapa de análisis. Una vez contratado un requisito entra en el proceso de control, seguimiento y medición de cambios, aún cuando sea cambiado en una futura etapa de análisis.

Los requisitos pertenecen a distintas familias, estas familias dependen de atributos particulares de los requisitos y del contexto de aplicación. Así hay requisitos del sistema, requisitos del proceso de desarrollo, y requisitos de la funcionalidad, requisitos legales, etcétera. Estas familias de requisitos tienen organizaciones variadas, de relaciones entre ellos, por ejemplo los requisitos de desarrollo estarán organizados por dependencias relativas a la organización del proceso de desarrollo y habrá requisitos relacionados con cada etapa y sub-etapa del proceso. Los requisitos de la organización se podrán ordenar según los procesos internos de la organización que se está modelando en el sistema. En los hechos se debe contar con relaciones entre los requisitos según diversos criterios. Estas relaciones entre los requisitos están dadas por tipos de requisitos (restricción/regla, estructura, proceso/tarea, objeto/materia) y tipos de relaciones (clase-generalización/subclase-particularización, par-

te-de/contenido-en, atributo-de/portador-de, secuencialidad/temporalidad, instancia-de/clase-de). Este esquema define una estructura compleja de relaciones que se irá registrando sobre el dominio de análisis. Esta caracterización de las relaciones permite la extracción relativamente simple de información relevante al proceso de construcción de software. Por ejemplo, el seguimiento de las relaciones clase-generalización/subclase-particularización entre requisitos de tipo objeto/materia nos permite extraer la taxonomía de clases preliminar para el proyecto.

El sistema permitirá navegar y visualizar la estructura de la red de requisitos. Cuando el operador se encuentra visualizando un requisito observará las relaciones con otros requisitos o definiciones y el tipo de esa relación, decidiendo cual otra relación le interesa navegar u observar. Estos requisitos luego son puestos en correspondencia con los módulos o funcionalidades del sistema construido (requisitos del sistema o de funcionalidad), o con actividades específicas del proceso de desarrollo (requisitos del proceso de testing, etc.), ya que cualquier actividad realizada por el grupo de desarrollo debe responder a un requisito que es el que inicia los eventos garantizando la trazabilidad.

3.5 Gestión del Riesgo

A diferencia de los procesos de producción industrial, los procesos de producción de software generan productos intangibles y requieren comunicación y coordinación intensivas lo que contribuye a aumentar los riesgos y dificultar la estimación. La experiencia ha demostrado que la construcción de software por métodos formales contribuye a crear sistemas más baratos, en menor tiempo, y con mayor confiabilidad [24]. Los procesos de desarrollo evolutivos tales como “Software Evolution” [24] o “Spiral” [25] también han mejorado la calidad del software. Sin embargo, todos estos modelos presentan una flaqueza al estimar el riesgo informalmente [26]. Muchas investigaciones han atacado el problema de la estimación del riesgo con un enfoque heurístico [27], [28], [29], [30]. En general todos estos enfoques se basan en listas de verificación, taxonomías de riesgo, prácticas recomendadas, y algunas pocas métricas. La mayoría de estas métricas miden el esfuerzo realizado y no el valor entregado al usuario, la satisfacción del cliente ni el valor generado. La informalidad del proceso constituye una debilidad y hace posible que diferentes individuos lleguen a diferentes conclusiones observando un mismo escenario. Esto representa un riesgo en si mismo ya que se depende de la experiencia, destreza, objetividad y manejo de las presiones políticas por parte del evaluador. Por otra parte, la dificultad en estimar la duración de los proyectos agrega complejidad al evaluar riesgo.

A pesar de que la existencia de modelos de estimación tales como COCOMO, COCOMOII [31], [32], SLIM [33], etc. es muy común que los presupuestos y los tiempos de desarrollo sean subestimados [34], [31]. Existen indicios de que en un alto porcentaje de casos, los desvíos obedecen a problemas organizacionales; y - a pesar de los avances en herramientas CASE - la gerencia de proyectos de software no ha avanzado sustancialmente [28]. Por lo tanto, si se desea mejorar la disciplina es necesario investigar tanto los aspectos organizacionales como técnicos del problema. Estos modelos no explican los fenómenos internos de los procesos de desarrollo de software. Sin este tipo de conocimiento es imposible avanzar cualitativamente en la disciplina. Herramientas de planificación como PERT y CPM no son aplicables a procesos de desarrollo de software evolutivos ya que se basan en grafos dirigidos acíclicos. El desarrollo de software es un caso particular de proyecto que presenta dos características fundamentales. Primero, existe un alto grado de incertidumbre sobre el producto final, su costo, sus riesgos, y el esfuerzo que implica su desarrollo. Segundo, el producto final es bastante intangible y su valor real depende no sólo de su correctitud, sino además del momento en que se pone en servicio, de la calidad apreciada por el usuario, y de su adaptabilidad, mantenibilidad y extensibilidad. El principal desafío que tienen los gerentes es entre tomar decisiones tempranas bajo alta incertidumbre, o mitigar la incertidumbre pagando con tiempo y encareciendo el proyecto. Nogueira [26] resolvió parte de estos problemas. Este trabajo, que se restringió a

desarrollos de sistemas de tiempo real basados en especificaciones formales con generación automática de código, permitió descubrir cuales son las métricas tempranas recolectables automáticamente que puedan usarse como indicadores de riesgo.

En este proyecto se evalúa el riesgo como el producto del resultado obtenido en un determinado escenario por la probabilidad de que este escenario se presente. Esta visión es menos restrictiva que la aplicada habitualmente y es aplicable a amenazas, oportunidades y pérdida de las mismas [26]. En particular deseamos un modelo que para un proyecto dado permita calcular la probabilidad de finalizarlo en determinado tiempo, o dada la probabilidad deseada nos diga cual es el tiempo requerido para tener esta probabilidad de finalizar en este tiempo. La probabilidad de concluir en la fecha prevista es función de la eficiencia de la organización, volatilidad de los requisitos, y la complejidad del proyecto [26]. En el corto plazo la eficiencia de la organización es constante [33], por lo que este modelo será una herramienta útil para evaluar el efecto de la volatilidad de los requisitos y negociar tiempo y requisitos sobre bases objetivas y claras para todas las partes. Como parte de este proyecto se desarrolló una herramienta de obtención de métricas de complejidad de bases de conocimiento [35], [36]. Esta es la primera etapa de una investigación dirigida a desarrollar el modelo antes mencionado. Para elaborar el modelo se comenzará con mediciones de bases de conocimiento post-mortem. Las conclusiones así obtenidas serán refinadas con los resultados de la integración de la captura de métricas al IDE y su captura en tiempo real, permitiendo la mejora del modelo, un monitoreo continuo del riesgo y la generación de información que permita entender mejor el proceso definido e identificar oportunidades de mejora tanto personales como grupales y organizacionales. Esta funcionalidad se integrará a la herramienta, de forma que cuando los valores de probabilidad arrojados se desvíen de los deseados, indicará al gerente del proyecto que deberá procederse a una revisión de las causas y, de estimarse necesario, proceder a realizar los ajustes pertinentes.

3.6 Etapas y Actividades y Proceso Personal de Software

Se plantea un conjunto básico de tipos de etapas basados en la nomenclatura de PSP [11]. Sin embargo el encargado de TI podrá definir la cantidad de etapas que desee con los nombres que desee, aunque cada etapa deberá corresponder a alguno de estos tipos. El sistema permite definir un proceso sencillo y laxo que luego puede ser ajustado a medida que el equipo se acostumbra a los requisitos de cada etapa del proceso de software, en particular al ir incorporando los diferentes niveles de PSP a medida que el grupo de desarrollo adquiere nuevas habilidades. Así se incorporan un conjunto de actividades bien definidas junto con un programa adaptado de PSP para la formación en procesos y la creación de disciplina técnica en el equipo.

3.7 Programación Extrema

Varios aspectos de programación extrema son incorporados a los conceptos de los procesos sugeridos por el sistema. Se destaca la propuesta de desarrollo de a pares, aunque es claro que es opcional la aplicación de esta medida. Algún gerente de TI puede sugerir que la relación costo beneficio no es adecuada. Pero debemos recordar que los módulos de los sistemas deben funcionar coordinadamente con los módulos desarrollados por otros programadores, y que el intercambio de ideas y pareceres es importante para la transferencia de conocimiento y conceptos complejos. Por otra parte una orientación hacia la productividad individual, probablemente descuide algunos aspectos de la calidad y facilidad de integración resultante de los módulos desarrollados individualmente, contra la revisión continua que se presenta en el desarrollo de a pares. Sobre este tópico son importantes los trabajos de Laurie Williams [37] sobre la efectividad del desarrollo de a pares y las experiencias de ambientes que aplican PSP y TSP en ambientes universitarios. Este proyecto identificará ventajas y desventajas comparativas entre el desarrollo de a pares contra el desarrollo individual. Una hipótesis presentada es que en etapas tempranas el equipo tendrá más rendimiento con el desarrollo de a pares hasta que se estabilice el proceso de desarrollo y se llegue a niveles de calidad de casi el 100%. En-

tonces la experiencia, el proceso detalladamente definido, los requisitos no funcionales y el conocimiento generado en patrones de diseño y desarrollo, permitirá que el desarrollo individual sea más efectivo. Por lo tanto, en un equipo de trabajo ya interiorizado en los procesos, sólo los nuevos integrantes deberán pasar un tiempo efectuando "desarrollo de a pares", supervisados por un desarrollador experto del equipo. Esperamos obtener información sobre esto a partir de los datos provenientes luego de la implantación del sistema y su aplicación en algunas organizaciones. Otras prácticas de XP que se pueden aplicar con mayor facilidad en proyectos desarrollados con GeneXus son la entrega rápida de pequeñas versiones, el diseño simple, y la participación activa del cliente en el lugar (apoyada por la facilidad de GeneXus para la prototipación).

3.8 Indicadores de gestión y de control del proceso

El sistema brindará un conjunto de indicadores de gestión del proceso de desarrollo de software y sus productos. Si bien la lista inicial de indicadores es bastante importante [38], esta primer implementación incorpora solo algunos de los indicadores más relevantes para el seguimiento de los proyectos, como, los ya nombrados de "Earned Value", "Cost Performance Index", "Schedule Performance Index" y "Project Risk Index" entre otros. Además brindará un conjunto completo de las métricas de esfuerzo y tamaño de producto de acuerdo con los informes propuestos en PSP, aunque adaptados a las particularidades del ambiente GeneXus. La mayor parte de los informes se producirán en forma automática a partir de los ingresos de los desarrolladores y el encargado del proyecto.

3.9 Gestión de calidad, mejora continua y patrones de diseño

Se plantea un registro de errores que rescata el enfoque presentado en PSP. De esta manera los defectos encontrados en cualquier fase del proyecto son registrados junto con el esfuerzo necesario para resolverlo. Por cuanto la tarea de reparar defectos dentro del proceso de desarrollo está asociada a costos de defectos internos, este costo permite junto a otros evaluar el costo de calidad del proyecto e interpolar para identificar la posición del equipo de desarrollo dentro de la curva de Juran [23]. La caracterización de los defectos encontrados, junto con una evaluación de la posible causa permite disponer de elementos para identificar las causas profundas de múltiples defectos e implantar medidas preventivas o metodologías que las eviten, como la definición de patrones de diseño ya probados que ayuden al desarrollador a no tener que experimentar soluciones en casos determinados. Como consecuencia se podrá implantar un modelo de gestión de conocimiento que apoye al proceso de mejora continua del proceso de desarrollo, basado sobre el mismo modelo que el registro de requisitos como requisitos no funcionales.

3.10 Pruebas, revisión y recomendaciones de mejora del proceso.

En esta primera fase el sistema no incluirá herramientas para testing. Si, mecanismos para registrarlos e ingresar las evaluaciones, y soporte de revisiones formales o por pares. Un aspecto importante para el éxito de un programa de mejora de la calidad y el desempeño dentro de una organización es que todos y cada uno de los integrantes puedan sentirse partícipes de este proceso. Una de las herramientas más importantes para ello es asegurarse de que haya mecanismos de participación de todos los integrantes en las propuestas de mejora. Para esto es necesario que la información de los indicadores de gestión esté disponible para todo el equipo de forma que se vayan educando en el análisis del trabajo propio y grupal. Esta visión de globalidad les permitirá comprender, analizar y criticar su propio trabajo y el del equipo, y proponer cambios que serán menos resistidos por ser los propios integrantes quienes los han sugerido.

3.11 Capacidad de llevar adelante políticas de tercerización.

El área de TI y en particular su equipo de desarrollo deben estar en condiciones de llevar adelante políticas de outsourcing [39]. Para lograr esto el equipo debe dominar las mejores prácticas y debe tener una evaluación clara de sus costos y sus límites relativos a la dimensión de los proyectos que puede llevar adelante. Esto requiere que el grupo de desarrollo disponga de métricas, procedimien-

tos de control de calidad, una experiencia suficiente para controlar y dirigir los trabajos del grupo tercerizado y garantizar el control del proyecto y del conocimiento de la organización. Esta herramienta apoya esta filosofía y puede ser aplicada a la auditoría y el control a los servicios de terceros.

3.12 Intercambio de información

La herramienta dispondrá de mecanismos para que el gerente de proyectos, pueda publicar diferentes niveles de información sobre sus proyectos, lo que permite aumentar el conocimiento de todos los participantes en el programa. Estas vistas podrán ser extraídas en formato texto para que el gerente las pueda examinar y controlar previo a su distribución, asimismo, puede personalizarse la información distribuida garantizando la confidencialidad de información sensible.

3.13 Metodología de trabajo

El ambiente permitirá al supervisor del grupo de desarrollo, crear un proyecto,asignar los desarrolladores, y definir un ciclo de vida para el proyecto. Por cada ingreso al sistema, y se medirá el tiempo involucrado en cada actividad por cada desarrollador o par de desarrolladores. Una pequeña ventana llevará cuenta del tiempo y dispondrá de un botón de start/stop para registrar las interrupciones con un comentario opcional. Al entrar al proyecto se registra también la "base de conocimiento GeneXus" sobre la que el sistema debe registrar los elementos de código generados, modificados y eliminados. Esta información se registrará en una base de datos común, desde la que se podrán extraer los informes que se requiera.

4 Conclusiones, resultados esperados y futuros planes

La implementación se realizará en GeneXus, aprovechando las herramientas públicas del ambiente GXOpen, para el análisis del código de especificación formal. Algunas herramientas desarrolladas en otros ambientes también estarán disponibles en Internet próximamente. Dado que esta herramienta contribuirá a la mejora personal y a facilitar la adopción y medición de procesos de desarrollo de software en pequeñas organizaciones, se espera tener una respuesta importante que genere un ámbito de intercambio de información y de auto-capacitación sobre temas de desarrollo de software. Este intercambio, esperamos pueda fomentar otros tipos de colaboraciones entre los grupos involucrados como ser benchmarking tecnológico y asesoramientos cruzados. Además se espera disponer de información relevante para publicar resultados y afinar los modelos de estimación de costos, tiempos y riesgos en estos ambientes. En una segunda etapa se piensa extender las posibilidades del ambiente de desarrollo con herramientas para automatizar el testing y realizar un seguimiento de los errores detectados.

5 Referencias

[1] Artech, GeneXus. http://www.genexus.com/

[2] Larre Borges, U. Encuesta a 27 empresas. Procesos para el Desarrollo de Software de Calidad. Tesis de Maestría, Facultad de Ingeniería, Universidad de la República. A ser publicado 2003.

[3] Latorres, E., Machuca, E.; Proyecto de Proceso de Desarrollo: Aplicación de técnicas de PSP y XP para un ambiente de desarrollo GeneXus. MTOP-FING-Universidad de la República. 2001.

[4] Informe del Banco Interamericano de Desarrollo (TC-99-10-05-6-UR) sobre la industria del software en el Uruguay, 1999.

[5] Basili V.R.; Caldiera G.; Rombach H.D. "The GQM Approach". In the Encyclopedia of Software Engineering, Wiley, 1994.

[6] Juristo, N.; Moreno A.N.; Basics of Software Experimentation, Kluwer Academic Pub., 2001.

[7] Grupp, B.; La gestión del Departamento de Informática, Editorial Hispano Europea S.A. Barcelona, 1985.

[8] Ouellette L.P.; How to market de I/S Department Internally; AMACOM, New York, 1992.

[9] Secretaria de Política de Informática-Ministério da Ciencia e Tecnología; Qualidade e Produtividade no Setor de Software Braileiro, http://www.mct.gov.br/sepin
[10] Grady, R.B.; Caswell, D.L.; Software Metrics: Establishing a company-wide program, Section 7, Human Factor, Prentice Hall, New Jersey, 1992.
[11] Humphrey, W.S. A discipline for software engineering. Addison-Wesley. May 1995.
[12] Humphrey, W.S.; Chapter 15, Team Synergism. Managing technical people, Addison Wesley Logmann, 1997.
[13] Goldratt, E.; Theory of Constrains, North River Press, Massachusetts, 1990.
[14] Sullivan, E.; section 1, Under pressure and on time, Microsoft Press, Best Practices, 2001.
[15] Chiavenato, I.; Administración de recursos Humanos, sección 8, McGraw-Hill, 1994.
[16] Schvarstein, L.; Evaluación del desempeño: Una perspectiva políticamente incorrecta, en Psicología y Organización del Trabajo, Ediciones Multiplicidades, Montevideo, 2000.
[17] Humphrey, W.S. Managing the software process. Addison-Wesley. 1990.
[18] Larman, C. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process, July 2001.
[19] Beck, K.; Extreme Programming Explained, Adison Wesley, 2000.
[20] Humphrey, W.S. Introduction to the Team Software Process. Addison-Wesley. 1999.
[21] Juran, J.M.; Quality Control Handbook, 3rd Edition, McGraw-Hill, New York, 1974.
[22] Harrington, H.J.; El coste de la mala calidad, Ediciones Díaz de Santos, S.A., Madrid 1990.
[23] Houston, D.;Keats, J.B.; Cost of Software Quality: A Means of Promoting Software Process Improvement, http://www.eas.asu.edu/~sdm/houston/scoqpap1.rtf
[24] Luqi and Goguen, J. Formal Methods: Promises and Problems. IEEE Software. Jaunary, 1997.
[25] Boehm, B. A Spiral Model of Software Development and Enhancement. Computer. May, 1988
[26] Nogueira, J.C. A Formal Risk Assessment Model for Software Projects. Ph.D. Dissertation. Naval Postgraduate School, 2000.
[27] Boehm, B. Software Risk Management. IEEE Computer Society Press. 1989.
[28] Hall, E. Managing Risk. Methods for Software Systems Development. Addison Wesley, 1997.
[29] Karolak, D. Software Engineering Management. IEEE Computer Society Press, 1996.
[30] Software Engineering Institute. Software Risk Management. Technical Report CMU/SEI-96-TR-012. June, 1996.
[31] Boehm, B. Software Engineering Economics. Prentice Hall, 1981.
[32] Boehm, B. et al. Software Cost Estimation with COCOMO II. Prentice Hall, 2000.
[33] Putnam, L. and Myers, W. Industrial Strength Software. Effective Management Using Measurement. IEEE Computer Society Press, 1997.
[34] Luqi. Software Evolution Through Rapid Prototyping.IEEE Computer May, 1989.
[35] Dávila Daniel and Chalar Luis, Estimación de Proyectos, Métricas y Herramientas XII Genexus International Meeting.
[36] Salvetto, P.; Nogueira, J.C.; Size Estimation for Management Information Systems Based on Early Metrics: An Automatic Metric Tool Based in Formal Specifications. International Conference on Computer Science,Software Engineering, Information Technology, e-Business,and Applications (CSITeA 03). Rio de Janeiro, Brasil, 2003.
[37] Williams, L.; Publications, http://collaboration.csc.ncsu.edu/laurie/pubs
[38] Latorres, E.; Indicadores de Gestión para el Departamento de Informática: Directivas para el Plan de Implantación. MTOP/DI, 2002.
[39] Latorres, E.; Tercerización en Informática. Percepciones Nº 6, Revista del Capítulo Montevideo - Uruguay de I.S.A.C.A. A ser publicado 2003.