International Journal of Information Systems and Software Engineering for Big Companies (IJISEBC)

IJISEBC, 01, I, 2014

Visión General del Desarrollo Global de Software

Overview of Global Software Development

Aurora Vizcaíno[1], Félix García[1], Mario Piattini[1]

[1] Instituto de Tecnologías y Sistemas de Información, Universidad de Castilla-La Mancha, España

Aurora.Vizcaino@uclm.es, Felix.Garcia@uclm.es, Mario.Piattini@uclm.es

RESUMEN. Este artículo presenta una panorámica general del estado del arte y de la práctica del Desarrollo Global de Software (DGS), analizando las principales revisiones sistemáticas de la literatura e identificando un conjunto de áreas de gran interés en la actualidad.
El cual muestra que el DGS es un campo que empieza a alcanzar cierta madurez: cuya evolución ya no se encuentra limitada por factores críticos como las diferencias lingüísticas y culturales, sino que ésta depende más de factores como la motivación personal y las habilidades de los recursos humanos, y de la disponibilidad de funciones y responsabilidades bien definidas; y, al mismo tiempo, presenta nuevos desafíos centrados en importantes líneas de interés como: los Procesos para desarrollo y gestión, la Gestión de Proyectos DGS y los Equipos de Trabajo.

ABSTRACT. This paper presents an overview of the state of the art and the practical of Global Software Development (DGS), analyzing the main systematic reviews of the literature and identifying a set of areas of great interest today.
Which shows that the DGS is a field that begins to reach a certain maturity: whose evolution is no longer limited by critical factors such as language and cultural differences, but it depends more on factors such as personal motivation and skills of resources human, and the availability of clearly defined roles and responsibilities; and at the same time, presents new challenges focused on important areas of interest include: Processes for development and management, DGS Project Management and Task Forces.

PALABRAS CLAVE: Desarrollo Global de Software, DGS, Revisión sistemática, Estado del arte, Estados de la Práctica, Trabajos futuros.

KEYWORDS: Global Software Development, DGS, Systematic review, State of the Art, State of the Practical, Future studies.

Vizcaíno, A., García, F., y Piattini, M. (2014). Visión General del Desarrollo Global de Software. International Journal of Information Systems and Software Engineering for Big Companies (IJISEBC), Vol. 1, Num. 1, pp. 8-22. Consultado el [dd/mm/aaaa] en www.ijisebc.com

1. Introducción

El Desarrollo Global de Software (DGS) se ha consolidado como uno de los aspectos más relevantes en la investigación y práctica de la Ingeniería del Software en la década de 2010 como ya lo predijera (Boehm, 2006). Surge hace más de 20 años con las primeras prácticas de outsourcing, pero se oficializa como tal en 2006 con la celebración de la primera conferencia internacional sobre desarrollo global ICGSE (IEEE International Conference on Global Software Engineering).

En general, el DGS supone para las empresas una manera de disminuir costes de desarrollo intentando mantener el nivel de calidad (Audy et al., 2004). En particular, este tipo de desarrollo permite obtener los siguientes beneficios:

- Contar con profesionales a lo largo y ancho del mundo sin necesidad de afrontar el costo de traslado de esas personas (Kobitzsch et al., 2001). Como resultado se puede por ejemplo reducir el coste de contratación de desarrolladores de software cuyos salarios son más reducidos en ciertos países, mediante la subcontratación de una empresa o la creación de filiales de la misma empresa en otros países (Damian y Moitra, 2006).
- Producir software para clientes remotos sin necesidad de trasladar el equipo de desarrolladores, incrementando de esta forma las posibilidades de introducirse en nuevos mercados (Richardson et al., 2005; Damian y Moitra, 2006).
- Lograr jornadas de trabajo más extensas, y por ende mayor productividad, cuando los programadores se encuentran distribuidos en sitios con amplia diferencia horaria (Carmel y Agarwal, 2001; Ebert y Neve, 2001; Herbsleb y Moitra, 2001).
- Obtener ventajas de la diversidad de experiencias, conocimiento técnico y destrezas de los stakeholders distribuidos (Ebert y Neve, 2001; Richardson et al., 2005).

Existen otros factores que hacen interesante este modelo de desarrollo, como es el aumento de la innovación que nace de la diversidad cultural y de compartir experiencias entre sus miembros.

Como contrapartida el DGS ha producido un profundo impacto en la manera en que los productos software se conciben, diseñan, construyen, prueban y entregan a los clientes (Herbsleb y Moitra, 2001), para lo que se necesitan nuevos métodos de trabajo (Damian et al., 2004). El DGS también presenta una serie de problemas, según (Conchúir, 2010) los principales desafíos encontrados en DGS se pueden agrupar en las llamadas "3C's": desafíos en la comunicación, entendiendo el concepto de la comunicación como el intercambio de conocimientos e información; desafíos en la coordinación, relacionados con la realización de tareas para alcanzar objetivos e intereses comunes y; desafíos en el control, que se centran en la gestión del proyecto (cumplir calendarios de entregas, presupuestos, calidad, estándares, etc.). Dichos desafíos se acentúan debido a lo que se ha llamado en la literatura las tres distancias (Ågerfalk et al., 2005):

- Distancia geográfica, definida como "la medida de esfuerzo que un individuo necesita realizar para visitar otro punto, alejado del primero" (Conchúir, 2010). Por ejemplo, dos lugares dentro del mismo país con un enlace aéreo directo y vuelos regulares, se pueden considerar relativamente cercanos, aunque estén separados por grandes distancias kilométricas. Sin embargo, no se puede decir lo mismo de dos lugares que están cerca geográficamente (separación de pocos kilómetros) pero con poca infraestructura de transporte. Este último caso tendría una elevada distancia geográfica.
- Distancia temporal, definida como "la medida de la deslocalización en tiempo, experimentada por dos individuos que desean interactuar" (Conchúir, 2010). Esta distancia normalmente va unida a la anterior, ya que cuando existe distancia geográfica con frecuencia implica diferentes husos horarios, lo cual puede limitar o incluso impedir la comunicación síncrona porque no haya solape de horario entre dos equipos de trabajo que estén geográficamente distribuidos.
- Distancia socio-cultural definida como "la medida en que un individuo comprende las costumbres (símbolos, normas y valores sociales) y cultura de otro individuo" (Conchúir, 2010). Esta dis-

tancia aparece frecuentemente en DGS, ya que cada miembro del equipo puede tener una nacionalidad y cultura diferente. Este tipo de distancia puede provocar conflictos y malentendidos entre los diferentes miembros de los equipos de desarrollo y suele ser también la causa de retrasos en las entregas de productos, es por ello que es uno de los temas que con más frecuencia se trata en la literatura (Huang, 2007).

Como consecuencia de lo anterior en los estudios sobre DGS se reportan numerosos problemas, entre los que se encuentran la falta de solidez teórica (Betz, 2010), el desconocimiento de los riesgos que este tipo de desarrollos implican (Betz, 2010), el uso de los mismos métodos, procesos y herramientas usadas en desarrollos tradicionales (da Silva et al., 2011) y requisitos incompletos o pobremente especificados (Islam et al., 2009). Otros problemas que las empresas han encontrado en su aplicación son (Eskeli y Maurolagoitia, 2011): la curva de aprendizaje, de modo que las personas que no están familiarizadas con las nuevas tecnologías DGS se resisten al aprendizaje; la pobre interoperabilidad entre herramientas; y los roles y responsabilidades no definidos claramente, entre otros.

En este artículo se presenta una panorámica del estado del arte sobre DGS, analizando los avances realizados hasta la fecha e identificando los desafíos de futuro. Para ello se analizan las principales revisiones sistemáticas de la bibliografía y se resumen algunas de las líneas principales de interés. El artículo se estructura del siguiente modo: en el apartado 2 se presentan las principales revisiones sistemáticas sobre DGS. En el apartado 3 se analizan las principales áreas de interés identificadas a partir de las revisiones sistemáticas y de la experiencia práctica de los autores en proyectos con industria. Finalmente se presentan las principales conclusiones obtenidas.

2. Revisiones Sistemáticas sobre DGS

Debido a la actual tendencia hacia la globalización del desarrollo de software, se ha incrementado tanto el número de revisiones sistemáticas (SLRs: Systematic Literature Reviews) como de estudios de mapeo (Mapping Studies) que se consideran una manera de sintetizar la investigación existente de un modo riguroso e imparcial (Genero et al., 2014). En la tabla 1 se presenta el conjunto de revisiones sistemáticas que se han publicado desde 2008. En el caso de existir varias SLRs de los mismos autores que suponen una ampliación de la original, se ha considerado la más actual.

Tema	Subtema	Referencias
Investigación general sobre DGS	Análisis de estudios existentes sobre DGS	(Monasor et al., 2009; Alsudairi y Dwivedi, 2010; Kroll et al., 2011; Verner et al., 2012; Raza et al., 2013)
	Estudio empíricos en DGS	(Šmite et al., 2008; Treude et al., 2009; Šmite y Wohlin, 2011; Humayun et al., 2013)
	Soluciones en entornos DGS	(Schneider et al., 2013)
	Ontología sobre DGS	(Borges et al., 2013)
Arquitectura	Gestión de conocimiento: identificación del conjunto de entidades DDS y sus relaciones	(Nour, 2010)

Proceso de desarrollo	Scrum, factores de riesgo y estrategias	(Hossain et al., 2009; Almehida et al., 2011; Hossain et al., 2011)
	Cadena de búsqueda o *snowballing* como primer paso en el desarrollo del proceso software	(Jalali y Wohlin, Systematic literature studies: Database searches vs. backward snowballing, 2012)
	Casos prácticos sobre el uso de metodologías ágiles	(Junius et al., 2011; Šteinberga y Šmite, 2011; Jalali y Wohlin, "Global Software Engineering and Agile Practices: A Systematic Review," 2012)
	Gestión de Riesgos en DDS	(Jiménez Monasor y Piattini, 2009; Persson, 2010; Šmite y Wohlin, 2011)
	Modelos y Procesos	(Jiménez Monasor y Piattini, 2009; Prikladnicki y Audy, 2010; Rocha et al., 2011)
Gestión de proyectos	Herramientas para dar soporte al DGS	(Costa et al., 2010; Cataldo y Herbsleb, 2011; da Silva et al., 2011; Chauhan y Ali Babar, 2012; Portillo-Rodriguez et al., 2012)
	¿Qué factores hay que tener en cuenta para gestionar un proyecto software en DGS?	(Cataldo y Herbsleb, 2011)S31
	Barreras en gestión de proyectos DGS	(Niazi et al., Challenges of project management in Global Software Development: Initial results, 2013)
	Estimación	(Britto et al., 2014)
	Gestión de riesgos en DGS	(Verner et al., 2014)
Ingeniería de requisitos	¿Cuáles son los principales riesgos del análisis de requisitos en DDS? Modelos, técnicas y herramientas que dan soporte	(Ebling et al., 2009)
	Riesgos y desafíos para el análisis de requisitos en contextos de DGS. ¿Qué propuestas existen para solventarlos?	(Lopez et al., 2009; Khan et al., Situational factors affecting Requirement Engineering process in Global Software Development, 2013)
Gestión de la configuración	Problemas, soluciones y estudios realizados sobre la gestión de configuración.	(Fauzi et al., 2010; Costa y Murta, 2013)
Outsourcing	Selección de Proveedores: Barreras y factores que influyen.	(Prikladnicki et al., 2008; Khan et al., 2009; 2011)
	Factores que pueden ayudar a proveedores en la gestión exitosa de las diferentes actividades durante las etapas de un contrato	(Khan y Khan, 2014)
	Factores de éxito en colaboraciones	(Ali y Khan, 2014)

Colaboración, comunicación, control y distancias	Barreras, Riesgos y Factores que influyen	(Jiménez Monasor y Piattini, 2009; Khan et al., "Communication Risks and Best Practices in Global Software Development during Requirements Change Management: A Systematic Literature Review Protocol," 2013)
	Distancias culturales, geográficas y temporales. Control y coordinación.	(Noll et al., 2010; Conradi et al., 2012)
	Barreras que afectan a la colaboración entre equipos distribuidos. Soluciones para solventar estas barreras	(Nurdiani et al., 2011)
	Colaboración en el desarrollo software	(Rocha et al., 2011)
	Estudios sobre *awareness* realizados con el objetivo de dar apoyo al DGS	(Steinmacher et al., 2010; Šmite y Wohlin, 2011; Vivian et al., 2011)
	Follow the sun	(Kroll et al., 2013)
	Evidencia empírica sobre el impacto de la dispersión en coordinación, rendimiento de equipo y de proyecto.	(Nguyen-Duc et al., 2014)
	Perspectiva "social" en DGS	(Giuffrida y Dittrich, 2013; Niazi et al., Motivators of Adopting Social Computing in Global Software Development: Initial Results, 2013)
Enseñanza	Técnicas y casos prácticos que se han realizado para tratar de impartir docencia en el contexto del DGS	(Monasor et al., Preparing students and engineers for Global Software Development: A Systematic Review, 2010; Budgen et al., 2012)

Tabla 1. Categorías y subtemas de las revisiones sistemáticas sobre DGS.

Tal como se puede observar en la Tabla 1, las revisiones sistemáticas sobre DGS se han enfocado principalmente en las distintas problemáticas del DGS y en cómo resolverlas o disminuir su impacto. Además, un menor número de revisiones se enfocan en un tema importante cómo la tecnología puede dar soporte a las distintas actividades del DGS. También, se deduce una clara tendencia a la utilización de metodologías ágiles, que pueden suponer una mejora a determinados problemas. Por ello, se deduce que existe cierta madurez principalmente en la investigación sobre los factores que influyen negativamente en el DGS y en las propuestas de soluciones para evitarlos o disminuirlos. Sin embargo, todavía es necesaria más investigación en temas relacionados con las herramientas de apoyo a este tipo de desarrollo y en la validación empírica sobre cómo las metodologías ágiles benefician el DGS comparado con otras metodologías. La descripción de casos reales en empresas son poco frecuentes pero muy necesarios ya que las lecciones aprendidas en estos casos son más relevantes que las obtenidas por experimentos o casos de estudios realizados con alumnos.

Con todo ello, en el siguiente apartado se resume un conjunto de líneas representativas de interés en DGS.

3. Síntesis del Estado del Arte y de la Práctica sobre DGS

En este apartado se sintetizan los resultados sobre las principales áreas de interés en DGS a partir del análisis de las revisiones sistemáticas y de la bibliografía relacionada con cada área.

3.1. Procesos ágiles para gestión y desarrollo global de software

Una de las áreas que más interés suscita en DGS en los últimos años es la definición de procesos adecuados en entornos DGS. La tendencia se centra la búsqueda de una exitosa combinación de métodos ágiles y desarrollo de software distribuido, tal como se presenta en diversas propuestas (Kussmaul et al., 2004; Ambler, 2009; Batra, 2009; Lee y Yong, 2009).

En la revisión sistemática realizada por (Hossain et al., 2009), que abarca los artículos que tratan la aplicación de Scrum en DGS desde 2003 a 2008, se concluye que hay un creciente interés en la realización de estudios empíricos para validar Scrum en la práctica del DGS pero se deben considerar una serie de retos y posibles limitaciones, como la necesidad de considerar los factores del proyecto (complejidad, presupuesto, etc..) a la hora de extraer conclusiones, ya que los resultados de la aplicación de Scrum pueden estar condicionados por dichos factores; los retos a los que se enfrentan los equipos Scrum y la necesidad de soporte de herramientas para facilitarles la aplicación de prácticas de Scrum en un entorno global; la necesidad de extender o modificar las prácticas de Scrum para dar soporte a DGS. En la revisión sistemática realizada por (Jalali y Wohlin, "Global Software Engineering and Agile Practices: A Systematic Review," 2012) se establece como las prácticas ágiles más aplicadas en DGS son las reuniones diarias de Scrum, así como el desarrollo iterativo o en forma de sprints, seguido por la integración continua, la planificación de sprints y las reuniones de retrospectiva. En cuanto a la combinación de métodos de desarrollo y de gestión para DGS se observa un mayor número de estudios que reportan: XP-Equipos distribuidos, Ágil-Offshore, Scrum-Equipo distribuido y Scrum-Offshore. También destacan la necesidad de mayor colaboración academia-industria al identificarse distintas percepciones de ambos en los estudios reportados.

En estos últimos años se ha avanzado en algunos de los desafíos comentados anteriormente, tanto en el desarrollo de herramientas web de soporte a Scrum, como en propuestas de solución de las limitaciones. Por ejemplo (del Nuevo et al., 2011), proponen la metodología "Scrum4D" cuyo objetivo es proporcionar guías para la gestión y desarrollo de software en un entorno distribuido utilizando las ventajas proporcionadas por la integración de métodos ágiles como Scrum y metodologías tradicionales como el Proceso Unificado de Desarrollo. Más recientemente, la tendencia se centra en la optimización de sprints en base al valor de las historias de usuario en proyectos DGS (Sobiech et al., 2014), la adaptación de las tareas del propietario del producto y del Scrum Master cuando trabaja a grandes empresas en proyectos a gran escala (Bass, 2013; 2014); y la gestión de cadenas de equipos scrum co-dependientes (Vlietland y Van Vliet, 2014).

3.2. Congruencia Socio-Técnica

Los problemas de coordinación y comunicación en DGS son, probablemente, uno de sus mayores desafíos. Con el fin de controlar, o al menos poder medir, estas dos variables surge lo que en inglés se llama "Socio Technical Congruence" (STC) que podemos traducir como congruencia socio-técnica. La idea principal del STC proviene de la ley de Conway que afirma que la estructura de un producto software refleja la estructura física de la organización. La STC se suele definir como la comparación entre el esfuerzo de coordinación requerido en un determinado proyecto de desarrollo de software con la coordinación que realmente se está llevando a cabo. Actualmente las empresas intentan conseguir un buen nivel de congruencia entre los requisitos de coordinación y las actividades de coordinación que actualmente se realizan. Teóricamente, un alto nivel de congruencia implicaría una mejora en la productividad y en la calidad del software. Sin embargo, también puede traer asociado un incremento de riesgo y coste. Además, un elevado número de interacciones puede provocar una sobrecarga de trabajo. Por lo tanto es conveniente que las empresas traten de encontrar el equilibrio entre el número de interacciones y el tiempo que se requiere para realizarlas.

Existen estudios empíricos que demuestran los beneficios de medir la STC. Concretamente en (Cataldo et al., 2006) se indica que controlando la congruencia se puede reducir el tiempo destinado a realizar determinadas tareas, como por ejemplo las modificaciones. Además, midiendo la STC se pueden detectar la falta de comunicación y evitar los efectos negativos que esto puede conllevar, por ejemplo, se ha comprobado que cuando hay falta de comunicación se incrementa el número de cambios que hay que realizar en el código (Ehrlich et al., 2008). Por su parte, los jefes de proyecto pueden beneficiarse del conocimiento de esta medida para controlar distintos aspectos, por ejemplo, comprobar el alineamiento entre la coordinación que existe en su equipo con las dependencias técnicas (Kwan et al., 2009). También, puede utilizarse para realizar un ranking de las tareas de coordinación que han faltado, analizando cual es la más importante y prioritaria de solucionar (Kwan et al., 2009).

Para medir la congruencia se requiere obtener información sobre: la estructura del equipo, las interacciones de comunicación, los procesos y prácticas de trabajo, la coordinación que se ha producido bien personalmente o a través de herramientas, el conocimiento tácito, la asignación de tareas y la localización de las personas, entre otros aspectos (Sarma et al., 2008). Actualmente, existen herramientas que ayudan a obtener parte de esta información, por ejemplo los foros, la mensajería instantánea, los gestores de correo o los repositorios de software. Otra información deberá obtenerse mediante encuestas o entrevistas. Las técnicas para medir la congruencia socio-técnica suelen comparar los requisitos de coordinación con la coordinación que realmente se está dando en el proyecto. Esta comparación se suele realizar usando matrices o redes sociales. En el caso de (Cataldo et al., 2006; Cataldo et al., 2008) se calcula comparando la que se llama Matriz de Requisitos de Coordinación (Cr) con la Matriz de Coordinación Real (Ca). La primera es una matriz persona a persona en la que cada celda representa qué debe coordinar un trabajador con otro. La Ca es calculada multiplicando la Matriz de Dependencia de Tareas (Td) en la cual cada celda representa la dependencia entre dos tareas y la Matriz de Tareas Asignadas (Ta) en la que cada celda representa el hecho de que un trabajador es asignado a una determinada tarea.

(Kwan et al., 2009; Kwan et al., 2011) se basan en el enfoque de Cataldo pero añaden un peso en las celdas. Ambos enfoquen ayudan a detectar falta de interacción entre dos personas que debe coordinarse.

Otro enfoque es el de (Ehrlich et al., 2008), en este caso se centran en detectar falta de comunicación en lugar de falta de coordinación. Para calcular la congruencia utilizan información de la comunicación obtenida de las redes sociales y la traducen a un grafo.

En (Portillo-Rodríguez et al., 2014) se presenta una arquitectura completa multiagente diseñada para medir y gestionar la STC con el objetivo de mejorar la coordinación y comunicación en DGS. Para ello se utiliza una arquitectura de agentes encargada de detectar de forma autónoma problemas de coordinación, calcular su importancia y notificar al jefe de proyecto sobre los problemas detectados y su importancia.

3.3. Estimación de Proyectos DGS

La estimación de proyectos software constituye un aspecto fundamental dentro de la gestión de proyectos ya que permite ir desde el inicio del ciclo de vida tomando decisiones más adecuadas en cuanto a la distribución de recursos a lo largo del proyecto. Los métodos de estimación paramétricos tradicionales que más se han aplicado en el campo de estimación del desarrollo y mantenimiento de software son Puntos Función (estimación de tamaño) y COCOMO II (estimación de tamaño y esfuerzo). Estos métodos, junto con otras técnicas de estimación tradicionales basadas juicio de expertos o ágiles como Planning Poker se siguen aplicando en proyectos DGS (Peixoto et al., 2010). De aquí se deriva la necesidad desarrollar métodos de estimación específicos para DGS, dada la heterogeneidad de métodos usados en proyectos DGS, lo que dificulta la comparación entre los resultados de estimación.

Sin embargo, estas técnicas tradicionales no se pueden aplicar tal cual para realizar estimaciones en Desarrollo Global de Software, ya que surgen nuevos retos que se deben resolver, en especial relacionados con la forma de distribuir el trabajo entre los distintas localizaciones (Lamersdorf et al., 2009) o nodos, así como nuevos factores de complejidad que pueden surgir. También es importante reutilizar el conocimiento de asignación de tareas en proyectos anteriores de desarrollo global.

En esta línea de interés, es importante considerar como factor clave la necesidad en DGS de realizar una asignación de trabajo sistemática entre las distintas localizaciones, que tenga en cuenta el coste de dicha asignación para seleccionar las mejores alternativas. A partir de lo anterior, (Lamersdorf et al., Estimating the Effort Overhead in Global Software Development, 2010; Lamersdorf et al., A Rule-Based Model for Customized Risk Identification in Distributed Software Development Projects, 2010) proponen: un modelo de asignación de trabajo en base a las características de los proyectos, sus objetivos y recursos disponibles; un modelo de estimación de costes para evaluar cada alternativa de asignación que considera las características

del sitio o localización y sus relaciones con otras tareas asignadas en otras localizaciones; y un modelo de identificación de riesgos para valorar y predecir los riesgos que pueden surgir de dichas asignaciones de trabajo.

Por su parte, se observa del análisis de la bibliografía que los modelos paramétricos de estimación tradicionales deben ser extendidos para considerar factores adicionales de complejidad en los que cubran las características especiales de DGS. En particular, COCOMO II ya considera un factor de coste denominado "Multisite development (SITE)" que tiene en cuenta el desarrollo distribuido, pero está enfocado en la perspectiva de infraestructura de soporte a la comunicación, lo que requiere incorporar más factores específicos para este tipo de desarrollos. En este sentido, (Keil et al., 2006) proponen nuevos factores a incorporar en el modelo COCOMO II así como una adaptación de alguno de los existentes para encajar mejor en la naturaleza de proyectos DGS. (Madachy, 2007) propone una extensión a las fórmulas del modelo COCOMO II que consideren el esfuerzo específico por fase, la distribución de trabajo a los equipos que trabajan en múltiples localizaciones y los atributos de cada equipo. Por su parte en (Vizcaíno et al., 2014) se propone un modelo de estimación que extiende la técnica de puntos función considerando nuevos factores de complejidad a tres niveles (factores a nivel global, factores del sitio, factores entre sitios).

En definitiva, la estimación aplicada a proyectos DGS se ha convertido en una línea de trabajo de gran interés y se observan ciertos avances aunque queda un importante camino por recorrer sobre todo a la hora de validar y calibrar los modelos propuestos para su aplicación en las empresas. Ello se confirma en estudios como el realizado por (Britto et al., 2014).

3.4. Formación

Una de las principales estrategias que permitirían minimizar los problemas que aparecen en el Desarrollo Global de Software (DGS), consiste en proporcionar una formación adecuada. Dicha formación debe permitir al alumno adquirir conocimientos, actitudes, habilidades y destrezas, en resumen, competencias para afrontar estos retos de manera eficaz. Concretamente, la literatura se describen numerosas dificultades a las que se han de enfrentar los ingenieros en DGS (Monasor et al., 2009; Nunamaker et al., 2009) derivadas de la interacción con equipos multiculturales y multilingües, menos oportunidades para que los miembros del equipo puedan comunicarse con la consecuente reducción de conversaciones informales (Palacio et al., 2009), la toma de decisiones requiere más tiempo (Wainfan, 2005) y crece el miedo por parte de algunos participantes a intervenir en las reuniones (Casey y Richardson, 2008; Casey, 2010), entre otras.

Para paliar lo anterior, se requiere adquirir una serie de competencias generales que son:

- Resolución de conflictos, que incluye entre otras la capacidad para hacer frente a situaciones difíciles y conflictivas (Niederman y Tan, 2011); el diagnóstico temprano de conflictos en el equipo virtual (Wainfan, 2005); actitud positiva y capacidad de motivación (Parvathanathan et al., 2007; Ocker et al., 2009).
- Trabajo en equipo, considerando entre otras, la capacidad para pensar desde la perspectiva del interlocutor (Favela y Peña-Mora, 2001) o la habilidad para ganar la confianza del equipo (Nguyen et al., 2006; Parvathanathan et al., 2007), el uso de estructuras de gratificación y recompensa o respuesta apropiada ante críticas o sugerencias (Niederman y Tan, 2011).
- Destrezas comunicativas, como el dominio de la lengua común empleada por la organización (Ebert y Neve, 2001) o el conocimiento de protocolos de comunicación y costumbres de las diferentes culturas (Richardson et al., 2007; Gotel et al., 2008), entre otras.

Por su parte, reproducir entornos DGS en contextos educativos es una tarea compleja. La mayoría de los estudios que se han encontrado en la literatura describen problemas organizativos cuando se trata de lograr la colaboración entre estudiantes de diferentes países. Por otra parte, los estudiantes que participan en estas actividades, poseen diferentes niveles de conocimiento o habilidades, lo que hace que sea necesario ofrecerles diferentes estrategias de entrenamiento (Soller, 2001). La cultura y lenguaje nativo de cada participante debe,

igualmente tenerse en cuenta en el diseño de dicha formación (Clear y Kassabova, 2008). Otro de los problemas destacados en la literatura es la ineficacia de la comunicación a través de medios como el correo electrónico o el chat, en ocasiones, relacionados con cuestiones técnicas (Daniels et al., 1998), lo que generalmente conlleva al incumplimiento de los plazos fijados.

Con el fin de proporcionar algunas soluciones a los problemas antes mencionados y formar adecuadamente a los desarrolladores de software en destrezas comunicativas en entornos globales, existen propuestas de simuladores como el entorno VENTURE (Virtual ENvironment for Training cUlture and language problems in global softwaRe dEvelopment) (Monasor et al., A Framework for Training Skills for Global Software Development, 2010). Este entorno presta especial atención a las diferencias culturales y lingüísticas, pero también es posible diseñar simulaciones de escenarios que permitan entrenar destrezas de trabajo en equipo, resolución de conflictos, así como escenarios específicamente enfocados a las distintas actividades que se pueden dar en DGS, tales como la elicitación de requisitos, la gestión del proyecto, la resolución de problemas, etc., y que impliquen una comunicación entre los participantes implicados.

3.5. Herramientas de soporte al DGS

Como ya hemos señalado, los retos que deben afrontar las organizaciones cuando los proyectos se llevan a cabo en un entorno global están principalmente relacionados con la distancia, ya sea geográfica, temporal o socio-cultural, lo que provoca un descenso importante de interacciones personales dificultando la colaboración, control y coordinación. Para minimizar los efectos negativos de la distancia, actualmente, existen tecnologías y aplicaciones que tratan de dar soporte a equipos de desarrollo que trabajan de manera distribuida, como en el caso de un proyecto de desarrollo global (Dubey y Hudepohl, 2013). Sin embargo, no todas las tecnologías y aplicaciones disponibles son capaces de ofrecer las mismas características para hacer frente a la colaboración en un entorno distribuido. Teniendo en cuenta que el desarrollo software está dividido en fases, cada aplicación deberá ofrecer características adaptadas a la fase de desarrollo para la que ha sido diseñada y además ofrecer características que mejoren la colaboración distribuida.

Con todo ello a continuación se ofrece una visión general de las herramientas software disponibles para DGS. En primer lugar, es importante considerar el conjunto de características que se deberían tener en cuenta a la hora de diseñar aplicaciones para DGS:

- Soporte al awareness. Significa que las herramientas deben incluir mecanismos a través de los cuales el usuario pueda conocer (ser consciente de) las acciones o eventos relevantes que sus compañeros de trabajo están realizando o produciendo.
- Soporte a la comunicación informal. En un entorno como el global donde los trabajadores prácticamente no pueden realizar charlas "cara-a-cara" con los compañeros que se encuentran en otra localización geográfica, es importante que las herramientas ofrezcan mecanismos para poder realizar charlas de una manera informal, directa y cómoda.
- Soporte al control y a la coordinación. Es importante que las herramientas proporcionen opciones de seguimiento de las tareas, errores, cambios, etc. e integren toda la información de manera que se pueda controlar el estado del proyecto.
- Análisis de dependencias socio-técnicas, mediante herramientas que consideren las relaciones sociales entre los miembros del equipo para realizar un análisis socio-técnico que ayude a mejorar la coordinación.
- Integración de datos. Debido al gran número de actividades realizadas durante el desarrollo software, es necesario usar diferentes tipos de herramientas para cubrir las necesidades de todas las actividades. Así, por ejemplo, es normal hacer uso de herramientas de control de versión, de seguimiento de errores, de peticiones de cambios o de generación de documentación entre otras.
- Soporte a la gestión del conocimiento, dado que en DGS el conocimiento se encuentra distribuido a través de los miembros de los distintos equipos. Esto hace necesario el uso de sistemas que ayuden

a capturar y distribuir dicho conocimiento como por ejemplo a través de Wikis o comunidades de práctica.

- Uso de versiones web de herramientas. Dado el alto grado de distribución en un entorno global, se debe permitir la interacción entre miembros del equipo desde cualquier lugar.

En (Alarcos, 2014) se presentan un conjunto de herramientas representativas que pueden ser útiles para DGS, clasificadas, de acuerdo a los procesos de ciclo de vida de ISO/IEC 12207, en las áreas de Planificación del Proyecto, Control y Aseguramiento del Proyecto, Análisis de Requisitos, Diseño del Software, Construcción del Software, Pruebas del Software, Gestión de la Documentación y Gestión de la Configuración. Por su parte, (García et al., 2011) presenta un conjunto de herramientas representativas de soporte a Ingeniería de Procesos que pueden ser de interés en DGS.

También resulta de interés considerar las herramientas que han sido desarrolladas en proyectos de investigación, y que dan también soporte a distintos procesos o actividades del ciclo de vida, como:

- Diseño distribuido: SYSIPHUS (Bruegge et al., Sysiphus: Enabling informal collaboration in global software development, 2006).
- Gestión de la Configuración: ADAMS (Bruegge et al., Supporting Distributed Software Development with fine-grained Artefact Management, 2006), Augur (Froehlich y Dourish, 2004), RepoGuard (Legenhausen et al., 2009), WikiDev (Bauer et al., 2009).
- Supervisión de Proyecto y actividades: WorldView (Al-Ani et al., 2008), WorkSpace Activity Viewer (Al-Ani et al., 2008), IssuePlayer (Garousi y Leitch, 2010), Implementación Syde (Hattori y Lanza, 2010), Share (Assogba y Donath, 2010).
- Clasificación de sistemas: MUDABlue (Kawaguchi et al., 2006).
- Gestión de Documentación DOCTOR (Krishnamurthy y Subramani, 2008) 4everedit (Meisinger et al., 2006).
- Gestión de Procesos XCHIPS (Fernández et al., 2004), GENESIS (Aversano et al., 2004).
- Gestión del Conocimiento: iBistro (Braun et al., 2002).

4. Conclusiones y Trabajos Futuros

En este artículo se ha presentado una panorámica general del estado del arte y de la práctica del DGS, analizando las principales revisiones sistemáticas de la literatura e identificando un conjunto de áreas de gran interés en la actualidad.

Una importante conclusión obtenida del estudio, es que el DGS es un paradigma muy dinámico que está en constante evolución. Una prueba de ello es por ejemplo el estudio realizado por (Vizcaíno et al., 2013), en el que se analizan los factores que afectan al DGS. Tradicionalmente se ha considerado como factores más críticos para los expertos las diferencias lingüísticas y culturales, así como la distancia geográfica (Carmel, 1999; Ågerfalk et al., 2005; Damian et al., 2005; Herbsleb, 2007; Cuppen et al., 2010). Sin embargo en base a los resultados de dicho estudio estos factores no son percibidos ya como clave por los expertos. En cambio, actualmente se consideran como los factores más importantes la motivación personal y las habilidades de los recursos humanos, al igual que disponer de funciones y responsabilidades bien definidas. Este resultado puede ser debido a que hoy en día estas distancias se han reducido gracias al uso de la tecnología adecuada y mediante la mejora del proceso software en entornos deslocalizados. Ello implica que el DGS es un campo que empieza a alcanzar cierta madurez y al mismo tiempo presenta nuevos desafíos centrados en importantes líneas de interés como:

- **Procesos para desarrollo y gestión**. Se observa que la tendencia es hacia la aplicación de métodos de gestión ágiles, como Scrum, pero adaptados a las particularidades y especial complejidad de desarrollos DGS, lo que supone retos importantes para que un equipo que trabaja de forma des-localizada

pueda funcionar como un equipo cohesivo y auto-organizado para el cumplimiento de sus objetivos.

- **Gestión de Proyectos DGS**, enfocada en una adecuada estimación de esfuerzo y costes del proyecto DGS considerando dimensiones adicionales así como estrategias efectivas de asignación de trabajo en las distintas localizaciones o a los distintos equipos distribuidos.

- **Equipos de Trabajo**, aspecto muy importante que debe permitir la formación de equipos muy cohesionados que alcancen adecuadamente sus objetivos en un entorno en el que hay que mejorar su comunicación, coordinación y control. Hoy en día se ha mejorado en infraestructura tecnológica que facilita lo anterior y se plantean desafíos constantes en la mejora de la congruencia socio-técnica en dichos equipos así como la importancia de una adecuada formación, donde los simuladores de entrenamiento pueden aportar importantes soluciones. También cobra mucho interés la mejora de las capacidades de los equipos que trabajan de forma distribuida. Ello motiva la necesidad de evolucionar al desarrollo de comunidades de práctica, como puede verse en (Monasor et al., 2013), donde los profesionales pueden compartir experiencias, lecciones aprendidas o "buenas prácticas" (Davila y Oktaba, 2013).

Agradecimientos

Este artículo ha sido financiado por los proyectos: GEODAS-BC (Ministerio de Economía y Competitividad y Fondo Europeo de Desarrollo Regional FEDER, TIN2012-37493-C03-01); SDGear (TSI-100104-2014-4), enmarcado en la iniciativa ITEA 2 (Call 7), y cofinanciado por el Ministerio de Industria, Energía y Turismo dentro del Plan Nacional de Investigación Científica, Desarrollo e Innovación Tecnológica 2013-2016 y por el Fondo Europeo de Desarrollo Regional (FEDER); INGENIOSO (Junta de Comunidades de Castilla La Mancha, PEII11-0025-9533); y GLOBALIA (Junta de Comunidades de Castilla La Mancha, PEII11-0291-5274).

Cómo citar este artículo / How to cite this paper

Vizcaíno, A., García, F., y Piattini, M. (2014). Visión General del Desarrollo Global de Software. International Journal of Information Systems and Software Engineering for Big Companies (IJISEBC), Vol. 1, Num. 1, pp. 8-22. Consultado el [dd/mm/aaaa] en www.ijisebc.com

Referencias

Ågerfalk, P. J., et al. (2005). A framework for considering opportunities and threats in distributed software development International Workshop on Distributed Software Development. A. C. Society. Paris, France 47-61.

Al-Ani, B., et al. (2008). Continuous coordination within the context of cooperative and human aspects of software engineering. the International Workshop on Cooperative and Human Aspects of Software Engineering. Leipzig, Germany, ACM: 1-4.

Alarcos (2014). "Global Software Development Tools." from https://sites.google.com/site/toolsgsd/.

Ali, S. y Khan, S. U. (2014). Critical Success Factors for Software Outsourcing Partnership (SOP): A Systematic Literature Review. International Conference on Global Software Engineering (ICGSE 2014). Shanghai, China: 153-162.

Almehida, L. E., et al. (2011). Applying multi-criteria decision analysis to global software development with scrum project planning. International Conference on Rough Sets and Knowledge Technology (RSKT 2011). Banff, Canada.

Alsudairi, M. A. y Dwivedi, Y. K. (2010). "A multi-disciplinary profile of IS/IT outsourcing research." Journal of Enterprise Information Management 23(2): 215-258.

Ambler, S. W. (2009). "The Distributed Agile Team." Dr. Dobb's Journal 34(1): 45-47.

Assogba, Y. y Donath, J. (2010). Share: a programming environment for loosely bound cooperation. Proceedings of the 28th international conference on Human factors in computing systems. Atlanta, Georgia, USA, ACM: 961-970.

Audy, J., et al. (2004). Distributed Analysis The Last Frontier? the 37th Annual Hawaii International Conference on Systems Sciences (HICSS), Big Island (Hawaii).

Aversano, L., et al. (2004). "Managing coordination and cooperation in distributed software processes: the GENESIS environment."

Software Process: Improvement and Practice 9(4): 239-263.
Bass, J. M. (2013). Agile Method Tailoring in Distributed Enterprises: Product Owner Teams. International Conference on Global Software Engineering (ICGSE 2013): 154-163.
Bass, J. M. (2014). Scrum Master Activities: Process Tailoring in Large Enterprise Projects. International Conference on Global Software Engineering (ICGSE 2014): 6-15.
Batra, D. (2009). "Modified agile practices for outsourced software projects." Communications of the ACM 52(9).
Bauer, K., et al. (2009). WikiDev 2.0: discovering clusters of related team artifacts. the Conference of the Center for Advanced Studies on Collaborative Research. Ontario, Canada, ACM: 174-187.
Betz, S. (2010). Knowledge Transfer in IT Offshore Outsourcing Projects: An Analysis of the Current State and Best Practices. 2010 5th IEEE International Conference on Global Software Engineering. A. Oberweis and Stephan, R. Princeton, New Jersey, USA. 0: 330-335.
Boehm, B. (2006). A view of 20th and 21st century software engineering. Proceedings of the 28th International Conference on Software Engineering (ICSE 2006). Shanghai, China: 12-29.
Borges, A., et al. (2013). Ontologies supporting the distributed software development: a systematic mapping study. Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering, Porto de Galinhas, Brasil.
Braun, A., et al. (2002). iBistro: A Learning Environment for Knowledge Construction in Distributed Software Engineering Courses. the Ninth Asia-Pacific Software Engineering Conference, IEEE Computer Society: 197-203.
Britto, R., et al. (2014). Effort Estimation in Global Software Development: A Systematic Literature Review. International Conference on Global Software Engineering (ICGSE 2014): 135-144.
Bruegge, B., et al. (2006). Sysiphus: Enabling informal collaboration in global software development. International Conference on Global Software Engineering (ICGSE'06) Florianopolis, Brazil 139-148.
Bruegge, B., et al. (2006). Supporting Distributed Software Development with fine-grained Artefact Management. Proceedings of the IEEE international conference on Global Software Engineering, IEEE Computer Society: 213-222.
Budgen, D., et al. (2012). What scope is there for adopting evidence-informed teaching in SE? International Conference on Software Engineering (ICSE 2012).
Carmel, E. (1999). Global software teams: collaborating across borders and time zones. Upper Saddle River, NJ (USA), Prentice Hall PTR.
Carmel, E. y Agarwal, R. (2001). "Tactical Approaches for Alleviating Distance in Global Software Development." IEEE Software 18(2): 22-29.
Casey, V. (2010). "Developing trust in virtual software development teams." Special Issue on Trust and Trust Management of the Journal of Theoretical and Applied Electronic Commerce Research 5(2): 41-58.
Casey, V. y Richardson, I. (2008). The Impact of Fear on the Operation of Virtual Teams. IEEE International Conference on Global Software Engineering, IEEE Computer Society: 163-172.
Cataldo, M. y Herbsleb, J. (2011). Factors leading to integration failures in global feature-oriented development: an empirical analysis. Proceedings of the 33rd International Conference on Software Engineering (ICSE 2011): 161-170.
Cataldo, M., et al. (2008). Socio-technical congruence: a framework for assessing the impact of technical and work dependencies on software development productivity. Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement. Kaiserslautern, Germany, ACM: 2-11.
Cataldo, M., et al. (2006). Identification of coordination requirements: implications for the Design of collaboration and awareness tools. the 20th Anniversary Conference on Computer Supported Cooperative Work. Banff, Alberta, Canada, ACM: 353-362.
Clear, T. y Kassabova, D. (2008). A Course in Collaborative Computing: Collaborative Learning and Research with a Global Perspective. Proceedings of the 39th ACM Technical Symposium on Computer Science Education. M. Guzdial and Fitzgerald, S. Portland, Oregon, USA, ACM: 63-67.
Conchúir, E. (2010). Global Software Development: A Multiple-Case Study of the Realisation of the Benefits. Limerick (Ireland), University of Limerick: 262.
Conradi, R., et al. (2012). Dispersion, coordination and performance in global software teams: a systematic review. Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement: 129-138.
Costa, C., et al. (2010). Models and tools for Managing Distributed Software Development: A systematic literature review. International Conference on Evaluation and Assessment in Software Engineering (EASE 2010). Keele University, UK.
Costa, C. y Murta, L. (2013). Version Control in Distributed Software Development: a Systematic Mapping Study. International Conference on Global Software Engineering (ICGSE 2013). Bari, Italia.
Cuppen, E., et al. (2010). "Q methodology to select participants for a stakeholder dialogue on energy options from biomass in the Netherlands." Ecological Economics 69(3): 579-591.
Chauhan, M. A. y Ali Babar, M. (2012). Cloud infrastructure for providing tools as a service: quality attributes and potential solutions. Proceedings of the WICSA/ECSA 2012: 5-13.
da Silva, F. Q. B., et al. (2011). "Research and practice of distributed software development project management: A systematic mapping study." Information and System Technology.
Damian, D., et al. (2005). "Requirements Engineering and Downstream Software Development: Findings from a Case Study." Empirical Software Engineering 10(3): 255-283.
Damian, D., et al. (2004). Workshop Introduction. 3rd International Workshop on Global Software Development. Co-located with ICSE 2004, Edinburgh (Scotland).
Damian, D. y Moitra, D. (2006). "Guest Editors' Introduction: Global Software Development: How Far Have We Come?" IEEE Software 23(5): 17-19.

Daniels, M., et al. (1998). RUNESTONE, an International Student Collaboration Project. IEEE Frontiers in Education Conference, Tempe, Arizona, IEEE.
Davila, M. y Oktaba, J. (2013). Run-Through Practice as a Collaboration Facilitator in Inter-organizational Software Construction. PARIS Workshop en International Conference on Global Software Engineering. Bari (Italy): 31-40.
del Nuevo, E., et al. (2011). Scrum-based Methodology for Distributed Software Development. International Conference on Global Software Engineering (ICGSE 2011): 66-74.
Dubey, A. y Hudepohl, J. P. (2013). Towards Global Deployment of Software Engineering Tools. International Conference on Global Software Development (ICGSE 2013): 129-133.
Ebert, C. y Neve, P. D. (2001). "Surviving Global Software Development." IEEE Software 18(2): 62–69
Ebling, T., et al. (2009). A Systematic Literature Review of Requirements Engineering in Distributed Software Development Environments. ICEIS '09, Milan, Italy.
Ehrlich, K., et al. (2008). "An Analysis of Congruence Gaps and Their Effect on Distributed Software Development." Mining Software Repositories.
Eskeli, J. y Maurolagoitia, J. (2011). Global Software Development: Current challenges and solutions. the 6th International Conference on Software and Data Technologies (ICSOFT), Seville (Spain).
Fauzi, S. S. M., et al. (2010). Software Configuration Management in Global Software Development: A Systematic Map. 17th Asia Pacific Software Engineering Conference, Sydney, Australia.
Favela, J. y Peña-Mora, F. (2001). "An Experience in Collaborative Software Engineering Education." IEEE Software 18(2): 47-53.
Fernández, A., et al. (2004). "Guided support for collaborative modeling, enactment and simulation of software development processes." Software Process: Improvement and Practice 9(2): 95-106.
Froehlich, J. y Dourish, P. (2004). Unifying Artifacts and Activities in a Visual Tool for Distributed Software Development Teams. the 26th International Conference on Software Engineering, IEEE Computer Society: 387-396.
García, F., et al. (2011). "Process Management Tools." IEEE Software 28(2): 15-18.
Garousi, V. y Leitch, J. (2010). "IssuePlayer: An extensible framework for visual assessment of issue management in software development projects." Journal of Visual Languages & Computing 21(3): 121-135.
Genero, M., et al. (2014). Métodos de Investigación en Ingeniería del Software, Ra-Ma.
Giuffrida, R. y Dittrich, Y. (2013). "Empirical Studies on the Use of Social Software in Global Software Development - a Systematic Mapping Study." Information and Software Technology: 23.
Gotel, O., et al. (2008). Integration Starts on Day One in Global Software Development Projects. IEEE International Conference on Global Software Engineering (ICGSE'08). Bangalore, India, IEEE Computer Society: 244-248.
Hattori, L. y Lanza, M. (2010). Syde: a tool for collaborative software development. the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2. Cape Town, South Africa, ACM: 235-238.
Herbsleb, J. D. (2007). Global Software Engineering: The Future of Socio-technical Coordination. International Conference on Software Engineering: Future of Software Engineering (FOSE'07). Minneapolis, MN, USA, IEEE Computer Society: 188-198.
Herbsleb, J. D. y Moitra, D. (2001). "Global software development." IEEE Software 18(2): 16-20.
Hossain, E., et al. (2009). Using Scrum in Global Software Development: A Systematic Literature Review. Fourth IEEE International Conference on Global Software Engineering (ICGSE'09). Limerick, Ireland, IEEE Computer Society: 175-184.
Hossain, E., et al. (2011). Towards an understanding of tailoring scrum in global software development: a multi-case study. Proceedings of the 2011 International Conference on Software and Systems Process (ICSSP 2011): 110-119.
Huang, H. (2007). Cultural influences and globally distributed information systems development: experiences from Chinese IT professionals. the ACM SIGMIS CPR Conference on Computer personnel research: The global information technology workforce. St. Louis, Missouri (USA): 36-45.
Humayun, M., et al. (2013). An empirical study on investigating the role of KMS in promoting trust within GSD teams. Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering (EASE 2013): 207-211.
Islam, S., et al. (2009). Goal and Risk Factors in Offshore Outsourced Software Development from Vendor's Viewpoint. the 4th IEEE International Conference on Global Software Engineering (ICGSE 2009).
Jalali, S. y Wohlin, C. (2012). "Global Software Engineering and Agile Practices: A Systematic Review." Journal of Software: Evolution and Process 24(6): 643-659.
Jalali, S. y Wohlin, C. (2012). Systematic literature studies: Database searches vs. backward snowballing. ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM: 29-38.
Jiménez Monasor, M. y Piattini, M. (2009). A Systematic Review of Distributed Software Development: Problems and Solutions. Software Engineering Approaches for Offshore and Outsourced Development (SEAFOOD 2008). Zurich, Switzerland.
Junius, B. A., et al. (2011). The impact of media selection on stakeholder communication in agile global software development: a preliminary industrial case study. Proceedings of the 49th SIGMIS annual conference on Computer personnel research: 131-139.
Kawaguchi, S., et al. (2006). "MUDABlue: An automatic categorization system for Open Source repositories." Journal of Systems and Software 79(7): 939-953.
Keil, P., et al. (2006). Cost estimation for global software development. Proceedings of the 2006 International Workshop on Economics Driven Software Engineering Research (EDSER '06). ACM. New York, NY, USA: 7-10.
Khan, A. A., et al. (2013). "Communication Risks and Best Practices in Global Software Development during Requirements Change Management: A Systematic Literature Review Protocol." Research Journal of Applied Sciences, Engineering and Technology 6(19): 3514.
Khan, A. W. y Khan, S. U. (2014). Critical challenges in execution of offshore software outsourcing contract from vendors' perspective: A systematic literature review. International Conference on Information and Communication Systems (ICICS 2014).

Khan, H. H., et al. (2013). Situational factors affecting Requirement Engineering process in Global Software Development. IEEE Conference on Open Systems (ICOS),: 118-122.
Khan, S. U., et al. (2009). Critical Success Factors for Offshore Software Development Outsourcing Vendors: A Systematic Literature Review. Fourth IEEE International Conference on Global Software Engineering (ICGSE'09). Limerick, Ireland, IEEE Computer Society: 207-216.
Khan, S. U., et al. (2011). "Barriers in the selection of offshore software development outsourcing vendors: An exploratory study using a systematic literature review." Information and Software Technology 53(7): 693-706.
Kobitzsch, W., et al. (2001). "Outsourcing in India." IEEE Software 18(2): 78-86.
Krishnamurthy, T. y Subramani, S. (2008). Ailments of Distributed Document Reviews and Remedies of DOCTOR (DOCument Tree ORganizer Tool) with Distributed Reviews Support. IEEE International Conference on Global Software Engineering (ICGSE 2008). Bangalore, India: 210-214
Kroll, J., et al. (2013). A Systematic Literature Review of Best Practices and Challenges in Follow-the-Sun Software Development. International Conference on Global Software Engineering Workshops (ICGSEW 2013). Bari, Italia: 18 - 23.
Kroll, J., et al. (2011). Mapping the Evolution of Research on Global Software Engineering - A Systematic Literature Review. Proceedings of the 13th International Conference on Enterprise Information Systems (ICEIS 2011). Beijing, China. 3.
Kussmaul, C., et al. (2004). Outsourcing and Offshoring with Agility: A Case Study (Experience Paper). Proceedings of XP/Agile Universe: 147-154.
Kwan, I., et al. (2009). A Weighted Congruence Measure. Workshop on SocioTechnical Congruence 1-4.
Kwan, I., et al. (2011). "Does Socio-Technical Congruence Have An Effect on Software Build Success ? A Study of Coordination in a Software Project." IEEE Computer Society: 1-20.
Lamersdorf, A., et al. (2009). A Survey on the State of the Practice in Distributed Software Development: Criteria for Task Allocation. International Conference on Global Software Engineering, Limerick, Ireland.
Lamersdorf, B. A., et al. (2010). Estimating the Effort Overhead in Global Software Development. 5th International Conference on Global Software Engineering (ICGSE 2010): 267-276.
Lamersdorf, B. A., et al. (2010). A Rule-Based Model for Customized Risk Identification in Distributed Software Development Projects. 5th International Conference on Global Software Engineering (ICGSE 2010): 209-218.
Lee, S. y Yong, H. S. (2009). "Distributed agile: project management in a global environment." Empirical Software Engineering: 1-14.
Legenhausen, M., et al. (2009). RepoGuard: A Framework for Integration of Development Tools with Source Code Repositories. the Fourth IEEE International Conference on Global Software Engineering, IEEE Computer Society: 328-331.
Lopez, A., et al. (2009). Risks and Safeguards for the Requirements Engineering Process in Global Software Development. International Conference on Global Software Engineering (ICGSE 2009), Limerick, Ireland.
Madachy, R. (2007). "Distributed Global Development Parametric Cost Modeling."
Meisinger, M., et al. (2006). "4everedit — Team-based Process Documentation Management." Software Process: Improvement and Practice 11(6): 627-642.
Monasor, M. J., et al. (2009). "Challenges and Improvements in Distributed Software Development: A Systematic Review." Advances in Software Engineering: 1-16.
Monasor, M. J., et al. (2010). A Framework for Training Skills for Global Software Development. International Conference on Global Software Development (ICGSE 2010), Princeton, NJ, USA.
Monasor, M. J., et al. (2010). Preparing students and engineers for Global Software Development: A Systematic Review. the International Conference on Global Software Development (ICGSE 2010), Princeton, NJ, USA.
Monasor, M. J., et al. (2013). Towards a Global Software Development Community Web: Identifying Patterns and Scenarios. PARIS Workshop en International Conference on Global Software Engineering. Bari (Italy): 41-46.
Nguyen-Duc, A., et al. (2014). "The impact of global dispersion on coordination, team performance and software quality – A systematic literature review." Information and Software Technology.
Nguyen, P. T., et al. (2006). Critical factors in establishing and maintaining trust in software outsourcing relationships. the 28th International Conference on Software Engineering. Shanghai, China, ACM: 624-627.
Niazi, M., et al. (2013). Challenges of project management in Global Software Development: Initial results. Science and Information Conference (SAI): 202-206.
Niazi, M. K., et al. (2013). Motivators of Adopting Social Computing in Global Software Development: Initial Results. Proceedings of the World Congress on Engineering 2013. London, U.K. 1: 1-5.
Niederman, F. y Tan, F. (2011). "Managing global IT teams: considering cultural dynamics." Commun. ACM 54(4): 24-27.
Noll, J., et al. (2010). "Global software development and collaboration: barriers and solutions." Magazine ACM Inroads 1(3): 66-78.
Nour, A. (2010). Architectural Knowledge Management in Global Software Development: A Review. 5th IEEE International Conference Global Software Engineering (ICGSE). Princeton, NJ, USA. 0: 347-352.
Nunamaker, J., et al. (2009). "Principles for effective virtual teamwork." Commun. ACM 52(4): 113-117.
Nurdiani, I., et al. (2011). Risk Identification and Risk Mitigation Instruments for Global Software Development: Systematic Review and Survey Results. Sixth IEEE International Conference on Global Software Engineering Workshop (ICGSEW).
Ocker, R., et al. (2009). "Training Students to Work Effectively in Partially Distributed Teams." ACM Transactions on Computing Education (TOCE) 9(1): 1-24.
Palacio, R., et al. (2009). "Providing Support for Starting Collaboration in Distributed Software Development: A Multi-agent Approach." WRI World Congress on Computer Science and Information Engineering 7: 397-401.
Parvathanathan, K., et al. (2007). Global Development and Delivery in Practice: Experiences of the IBM Rational India Lab, IBM Press.

Peixoto, C. E. L., et al. (2010). Effort Estimation in Global Software Development Projects: Preliminary Results from a Survey. 5th IEEE International Conference on Global Software Engineering (ICGSE 2010): 123-127.
Persson, J. S. (2010). A Process for Managing Risks in Distributed Teams. L. Mathiassen. 27(1): 20-29.
Portillo-Rodríguez, J., et al. (2012). "Tools used in Global Software Engineering: A systematic mapping review." Journal Information and Software Technology 54(7): 663-685.
Portillo-Rodríguez, J., et al. (2014). "Using agents to manage Socio-Technical Congruence in a Global Software Engineering project." Inf. Sci. 264: 230-259.
Prikladnicki, R. y Audy, J. L. N. (2010). "Process models in the practice of distributed software development: A systematic review of the literature." Inf. Softw. Technol. 52(8): 779-791.
Prikladnicki, R., et al. (2008). Patterns of Evolution in the Practice of Distributed Software Development: Quantitative Results from a Systematic Review. 12th International Conference on Evaluation and Assessment in Software Engineering (EASE) University of Bari, Italy.
Raza, B., et al. (2013). Topics and treatments in global software engineering research - A systematic snapshot. International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2013): 85-96.
Richardson, I., et al. (2005). Global Software Development – the Challenges. SERC Technical Report 278, University of Limerick, Ball State University: 10.
Richardson, I., et al. (2007). Globalizing Software Development in the Local Classroom. the 20th Conference on Software Engineering Education & Training, IEEE Computer Society: 64-71.
Rocha, R., et al. (2011). "Collaboration Models in Distributed Software Development: a Systematic Review." CLEI Electronic Journal.
Sarma, A., et al. (2008). Challenges in Measuring, Understanding, and Achieving Social-Technical Congruence. CMU-ISR-08-106.
Schneider, S., et al. (2013). "Solutions in Global Software Engineering: A Systematic Literature Review." International Journal of Information Management 33(1): 119-132.
Šmite, D., et al. (2008). Reporting Empirical Research in Global Software Engineering: A Classification Scheme. IEEE International Conference Global Software Engineering (ICGSE), Bangalore, India.
Šmite, D. y Wohlin, C. (2011). "A Whisper of Evidence in Global Software Engineering." IEEE Software 28(4): 15-18.
Sobiech, F., et al. (2014). On iteration optimization for non-cross-functional teams in Scrum. Proceedings of the Conference on Research in Adaptive and Convergent Systems (RACS 2014): 266-271.
Soller, A. (2001). "Supporting Social Interaction in an Intelligent Collaborative Learning System." International Journal of Artificial Intelligence in Education (IJAIED) 12: 40-62.
Šteinberga, L. y Šmite, D. (2011). Towards Understanding of Software Engineer Motivation in Globally Distributed Projects. Proceedings of the 2011 IEEE Sixth International Conference on Global Software Engineering Workshop (ICGSE 2011): 117-119.
Steinmacher, I., et al. (2010). Awareness Support in Global Software Development: A Systematic Review Based on the 3C Collaboration Model. International Conference on Collaboration and Technology (CRIWG 2010). Maastricht, The Netherlands: 185-201.
Treude , C., et al. (2009). Empirical Studies on Collaboration in Software Development: A Systematic Literature Review.
Verner, J. M., et al. (2012). Systematic literature reviews in global software development: A tertiary study. 16th International Conference on Evaluation & Assessment in Software Engineering (EASE 2012), Ciudad Real (Spain).
Verner, J. M., et al. (2014). "Risks and risk mitigation in global software development: A tertiary study." Information & Software Technology 56(1): 54-78.
Vivian, R. L. M. H., Elisa Hatsue, et al. (2011). Context-awareness on software artifacts in distributed software development: a systematic review. International Conference on Collaboration and Technology (CRIWG 2011): 30-44.
Vizcaíno, A., et al. (2014). Desarrollo Global de Software, Ra-Ma.
Vizcaíno, A., et al. (2013). "Applying Q-methodology to analyse the success factors in GSD." Information and Software Technology 55(7): 1200-1211.
Vlietland, J. y Van Vliet, H. (2014). "Towards a governance framework for chains of Scrum teams." Information & Software Technology 57: 52-65.
Wainfan, L. (2005). Challenges in Virtual Collaboration: Videoconferencing Audioconferencing and Computer--Mediated Communications, RAND Corporation.