Principios, Roles y Métricas en alineamiento estratégico de nuevos requerimientos utilizando DevOps

Alberto Belalcázar[1], Javier Díaz[1], Lía Molinari[1], Christian Rodríguez[1]

[1]LINTI – Facultad de Informática – Universidad Nacional de La Plata, UNLP
albertob@tesista.linti.unlp.edu.ar, javierd@info.unlp.edu.ar, lmolinari@info.unlp.edu.ar, car@info.unlp.edu.ar

Abstract. En un entorno de usuarios cada vez más exigentes y ansiosos de los servicios que les brinda la tecnología, las áreas de tecnología de la información y las comunicaciones (TICs) adquieren un papel estratégico en las organizaciones debido al apoyo y aporte que brindan a los usuarios. Las aplicaciones que utiliza cotidianamente, en forma ubicua, obliga a las áreas de TICs a dinamizar y acelerar los modelos tradicionales de desarrollo y entrega de aplicaciones, con estrategias que logren disminuir la brecha de alineamiento entre las áreas de desarrollo y operaciones. DevOps[1] ha surgido como una nueva tendencia orientada hacia la obtención de una relación de trabajo y colaboración entre áreas de desarrollo y operaciones de TICs. En este documento se presenta una hoja de ruta que permita alinear Organización-DevOps-Usuarios, y asegurar que los beneficios de aplicar las buenas prácticas de DevOps ayuden a superar la brecha entre operaciones con desarrollo. En este contexto se analizan principios, métricas, fronteras y roles y se presenta un caso de estudio realizado en estudiantes del ámbito universitario.

Keywords: DevOps, Integración Continua, Entrega Continua, Despliegue Continuo, Tiempos y Procesos Estratégicos, Procesos Sensibles.

1 Introducción

En una nueva concepción integradora promovida por los trabajos colaborativos entre desarrollos/operaciones, la automatización de tareas es una de las metas más deseadas y complejas.

El flujo de trabajo tradicional de desarrollo y operaciones en TI, en el que interviene personal con sus roles y responsabilidades se resume de la siguiente manera, haciendo una pronunciada simplificación: desarrollo produce sus programas, luego, estos son testeados y desplegados a producción sea en forma manual o automática.

Internamente el ciclo desarrollo-testing producción/despliegue puede estar formal o informalmente definido y ser efectivo en cuanto a la determinación de las funciones y responsabilidades involucradas entre desarrolladores y operadores.

[1] http://devops.com/

Sin embargo, cuando la aplicación se encuentra en producción, el usuario puede encontrar fallos o que no cumple con lo esperado por diferentes características que pueden ser la accesibilidad, el tiempo de respuesta, o relacionadas con la ubicuidad.

La detección temprana de los incidentes y una respuesta ágil desde TI, permitiría reducir el tiempo para solucionar el incidente y cerrarlo a satisfacción del usuario.

Al no existir una comunicación fluida entre las áreas de desarrollo de aplicaciones y operaciones, se pueden presentar fallos que ponen en riesgo la imagen de TI.

Para definir una hoja de ruta que permita alinear Organización-DevOps-Usuarios, es necesario realizar un análisis conceptual de componentes DevOps, identificando su ubicación y alcance en el marco de desarrollo y despliegue de software. La granularidad de componentes DevOps que permita identificar los Principios y las Fronteras de los Roles de DevOps, ayuda a determinar las actividades que pueden retrasar el avance de los procesos y flujos de trabajo colaborativo involucrados en el desarrollo y despliegue de software. Las métricas que se toman como referencia son parte de los conceptos DevOps cuyo análisis en un estudio de caso propuesto ayuda a establecer el factor determinante que tiene la relación de apoyo colaborativo de DevOps en el cumplimiento de objetivos con el desarrollo y despliegue de automatización de tareas corporativas.

2 Componentes DevOps

DevOps (Developer-Operators) es un conjunto de prácticas que permiten fomentar la colaboración y relación de trabajo entre desarrollo y operaciones de TI. DevOps es una mezcla de patrones destinados a mejorar la colaboración entre el desarrollo y operaciones [1][2][3][4][5][6][14]. El objetivo de esta colaboración es mejorar los tiempos de respuesta del ciclo de requerimiento-satisfacción del usuario, mediante aplicaciones y medios que ayuden a al inearlos y cerrar las brechas entre desarrolladores y operadores, aplicando prácticas ágiles en las áreas de desarrollo y operaciones[10][12].

La arquitectura en los que se desenvuelve DevOps es la siguiente:

- Componentes de la Gestión de Desarrollo de Software
- Componentes de la Gestión de Data Center
- Herramientas DevOps como apoyo al Desarrollo y Despliegue de un Proyecto

2.1 Relación de Componentes de la Gestión de Desarrollo de Software y Componentes de la Gestión de Data Center en base a DevOps

DevOps sincroniza la labor del área de desarrollo y operaciones en forma continua, y lo hace desde la etapa de planificación de las tareas a au tomatizar, hasta el despliegue de aplicaciones al usuario.

Existen dos instancias en las que se desenvuelve la arquitectura DevOps:

- Desarrollo de Aplicaciones
- Despliegue de Aplicaciones

Para definir la relación tiene DevOps en el Desarrollo y Despliegue de aplicaciones, es necesario que cada instancia sea granulada en subcomponentes, con el fin de saber los límites en los roles y principios de influencia.

DevOps requiere para el **Desarrollo de Aplicaciones** metodologías ágiles de desarrollo de software. La guía de metodología ágil se lo analizará en base a Scrum.

En la siguiente tabla se visualiza los roles que intervienen en Scrum:

Tabla 1. Roles de Scrum

Roles	Scrum Máster
	Dueño del Producto
	Equipo de Desarrollo
	Backlog del Producto

DevOps asume los roles de Scrum y para realizar labores colaborativas incluye a los operadores como parte del equipo de desarrollo y se crea el equipo de desarrollo DevOps. Por lo tanto el equipo de Desarrollo DevOps tiene los siguientes roles:

Tabla 2. Roles de DevOps

Roles	Scrum Máster
	Dueño del Producto
	Equipo de Desarrollo
	Backlog del Producto
	Operadores

Scrum tiene definido la siguiente metodología de desarrollo:

Tabla 3. Metodología o ciclo de Desarrollo de Software en base a Scrum

Metodología de Desarrollo de Scrum	Planificación de Iteración
	Inicio Iteración
	Desarrollo Iteración
	Versión Candidata

DevOps asume el ciclo de vida de Scrum, y lo hace por las siguientes pautas:

- Maneja iteraciones cortas. De 2 a 4 semanas por iteración
- Scrum cumple con los principios de cultura de DevOps, desarrollo y operaciones miran en un solo camino, que es la satisfacción del servicio brindado al usuario.
- Realiza entregas parciales del producto a ser automatizado, a través del trabajo colaborativo del equipo de desarrollo. Scrum es dependiente de la automatización end to end [16].

Las pautas de manejo de iteraciones cortas se encuentran directamente relacionadas con las métricas de DevOps.

La Métrica se define como la medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado [18].

Los atributos de las métricas se dan por acuerdo de un protocolo entre directivos y personal operativo, en base a un valor que mide la gestión de un área en un tiempo determinado.

El marco colaborativo en el que se desarrolla DevOps rompe el silo en el que cada una de las áreas siga viendo sus métricas de acuerdo al paradigma de medición de rendimiento acostumbrado.

Gartner ha definido las siguientes métricas para ambiente DevOps:

- Colaboración
- Tiempos de Ciclo
- Frecuencia
- Satisfacción del usuario

Estas métricas tienen relación directa con las pautas de ciclo corto de DevOps.

La **métrica de Colaboración** es parte del concepto de DevOps, desarrollo y operaciones trabajan en forma colaborativa, para entregar un producto en tiempo corto y como único objetivo satisfacer el servicio brindado al usuario.

La **métrica de Tiempo de Ciclo** es un i ndicador que nos medir el nivel de eficiencia que tiene un desarrollo desde el inicio de la iteración hasta el despliegue de la versión candidata.

La **métrica de Frecuencia** nos indica el tiempo repetitivo en el cual se debe entregar cada iteración.

La **métrica Satisfacción al Usuario**, nos permite medir el grado de satisfacción del usuario en base a la eficiencia de la automatización de las aplicaciones.

DevOps maneja principios que son parte de la estructura colaborativa que son utilizados en toda la etapa de desarrollo y despliegue de aplicaciones. Los principios en los que se desenvuelve DevOps son los siguientes [1][2][17]:

- Integración Continua
- Entrega Continua
- Despliegue Continuo

La **Integración Continua** es la forma en la que el equipo de desarrollo de software integra su trabajo parcial o total, en un determinado tiempo establecido por el equipo de trabajo [17].

La Integración Continua permite estrecha colaboración entre el equipo DevOps que ayude a desarrollar una iteración en un ciclo de tiempo definido en forma satisfactoria.

La Integración Continua para cada iteración con objetivos satisfactorios para el usuario, permitirá frecuencias exactas planificadas en la iteración.

La Integración continua requiere de herramientas de automatización que son únicas para todo el equipo de desarrolladores. Estas herramientas ayudan a integrar en forma

continua partes de código que son validados por pruebas automáticas, lo cual vuelve más eficiente el trabajo del equipo de desarrollo, ya que permite detectar fallos en etapas tempranas del ciclo de desarrollo, lo cual va a favorecer en los tiempos y calidad del producto.

La **Entrega Continua** consiste en desplegar la versión candidata que contiene los requerimientos corporativos automatizados a ser usados por el usuario.

Los tiempos de ciclo definen los periodos de entregas de versiones candidatas.

El **Despliegue Continuo** es la configuración del ambiente de trabajo, que permita un funcionamiento efectivo de las versiones candidatas por parte de los usuarios. Se empieza con una pre-configuración durante todo el proceso de desarrollo y una configuración final antes que se termine la versión candidata.

El Despliegue Continuo visualiza tempranamente los fallos que puedan existir en la aplicación. El momento en que el usuario interactúa más tempranamente con la aplicación, revisa sus requerimientos y se puede volver atrás en el desarrollo.

La segunda instancia en la que se desenvuelve DevOps es el **Despliegue de Aplicaciones**.

El análisis de los componentes del Data Center es crucial para entender la relación que tienen con los componentes del desarrollo de software. Los componentes de software del Data Center, deben ser granulados a un nivel de detalle que permita tener un control preciso de los servicios que están relacionados con las aplicaciones que sirven al usuario.

Desde la etapa de inicio del desarrollo del proyecto, en el que intervienen directamente los desarrolladores, estos se pueden soportar en herramientas que ayuden a dar una verdadera integración continua, para cada una de las iteraciones que se planifiquen para dar solución a los requerimientos a ser automatizados.

2.2 Herramientas DevOps como apoyo al Desarrollo y Despliegue de un Proyecto

El objetivo de la colaboración de DevOps es actualizar y mantener los sistemas de producción estables en todo momento [1]. Las herramientas sirven de apoyo para superar la brecha de desarrollo y operaciones, permiten ser utilizadas en la etapa de desarrollo en forma exclusiva como Dev, y también en la etapa de despliegue como Ops. Tanto para Dev como Ops existen herramientas con software, código, scripts o secuencias de comandos que ayudan a controlar de manera eficiente el ambiente de desarrollo y despliegue de aplicaciones.

Las herramientas DevOps son parte de la estrategia de una buena gestión de administración de recursos. En los ambientes de desarrollo de software las herramientas ayudan a conseguir integración continua, control de versiones de programas, pruebas automáticas, y despliegue continuo. La gestión de operaciones se soporta en herramientas para despliegue automático de aplicaciones, configuración de máquinas virtuales, ejecución automática de scripts.

DevOps se posiciona desde el inicio de la planificación de la iteración como parte de la integración conjunta de conocimientos de desarrolladores y operadores de los objetivos de las etapas del proyecto.

Mientras la aplicación desarrollada se encuentre en producción, el equipo DevOps será el responsable de atender los cambios en las versiones de las aplicaciones.

3. Análisis de Fronteras en los Roles de DevOps

Las áreas de desarrollo y operaciones por las funciones divergentes que tienen establecidas están acostumbradas a trabajar como silos, DevOps ayuda a romper estas divergencias desde las etapas de planificación de la iteración.

En las etapas de desarrollo, pruebas e implementación de una iteración el trabajo colaborativo debe sostenerse con prácticas que ayuden a establecer el trabajo efectivo en la definición de tareas prioritarias del usuario, por lo cual se necesita regular las funciones de las dos áreas.

DevOps no quita ni aumenta funciones, las regula, a través de las siguientes actividades que son anotadas en [1][15]:

- Cultura
- Compartición
- Automatización
- Métricas

El concepto de dar valor a los requerimientos del usuario, es el principal objetivo del componente **Cultura** [1].

La Compartición, es un elemento de DevOps, donde la gente comparte conocimiento, ideas, innovación, procesos y herramientas.

La automatización permite que las tareas manuales, sean realizadas por componentes computacionales de acuerdo a los requerimientos del usuario [1][17].

Las métricas permiten a las áreas de desarrollo de software y operadores, determinar indicadores que le sirve para calificar la eficacia de su labor en base a valores de sus atributos o procesos, pero en forma aislada para cada área.

4. Caso de Estudio

El siguiente caso de estudio fue realizado en la Universidad Central del Ecuador.

Colaboraron en él, 40 estudiantes de pregrado, de la Carrera de Ingeniería Informática.

El objetivo del trabajo es analizar la relación de métricas con principios y fronteras y roles de componentes en el Desarrollo y Despliegue de Software con herramientas DevOps.

El trabajo a realizar es el desarrollo de un sistema que permita el ingreso, actualización y consulta de nota de los estudiantes. El límite de tiempo para desarrollar fue de 100 minutos. Las computadoras donde se realizaron los programas fueron las mismas ya que en los 4 laboratorios existen los mismos tipos de equipos y software para el efecto.

El software para desarrollo de los programas fue Java y Mysql, considerando que los alumnos tenían el conocimiento y experiencia adquiridos en otras asignaturas.

El software para integración continua y control de versiones fue capacitado a 30 estudiantes. Jenkins, Git, GitHub.

Se capacitó a 30 estudiantes en Chef y Puppet.

Se capacitó a 30 estudiantes en Nagios.

Considerando que los alumnos iban a ser distribuidos en diferentes equipos, la composición de los equipos se hizo de forma tal de garantizar que cada equipo estaba capacitado para realizar el desarrollo solicitado con las herramientas requeridas. El uso de controles y herramientas fue discrecional.

4.1 Flujo de tareas y actividades a seguir y resultados obtenidos

- 40 estudiantes formaron 8 equipos (A, B, C, D, E, F, G, H) de 5 desarrolladores cada uno con su líder de equipo.

Tabla 4. Resultados obtenidos con tiempos por cada equipo de trabajo

Flujo de Tareas realizadas por los estudiantes	Equipo							
	A	B	C	D	E	F	G	H
Organización de equipo de trabajo	Si	Si	Si	Si	Si	Si	Si	Si
Estructurar el equipo de trabajo y designar un líder de equipo. Incluir al equipo al dueño del producto	Si	Si	Si	Si	Si	Si	Si	Si
Definición de Procesos manuales y automatizados con los que interactúa el dueño del producto	Si	Si	Si	Si	Si	Si	Si	Si
Planificación y Definición de Tareas a automatizar	Si	Si	Si	Si	Si	Si	Si	Si
Planificación y Definición de Tareas Prioritarias. Definición del ciclo y frecuencia de la iteración	Si	Si	Si	Si	Si	Si	Si	Si
Planificación por equipo de trabajo para que exista Integración, Entrega y Despliegue Continuo de las aplicaciones a realizar	Si	Si	No	No	Si	Si	Si	Si
Identificación de Componentes de la Gestión de Despliegue en Data Center - Servidor identificado, sistemas operativo de servidor, manejador de base de datos, software de frameworks	Si	Si	Si	Si	Si	Si	Si	Si
Identificación de Herramientas DevOps como apoyo al Desarrollo, Java, Mysql	Si	Si	Si	Si	Si	Si	Si	Si
Identificación de Herramientas DevOps como apoyo al Despliegue continuo de las aplicaciones a realizar. Uso de Jenkins y plugin de GIT y repositorio GitHub para Integración Continua y Control de Versiones.	Si	Si	No	No	Si	Si	Si	Si
Uso de Vagrant para despliegue de máquinas virtuales donde van a residir las aplicaciones en servidores.	Si	Si	No	No	Si	Si	Si	Si
Uso de Chef Solo, para configurar y desplegar software Centos, Java, Mysql en el	Si	Si	No	No	Si	No	Si	No

servidor.								
Uso de Puppet, para configurar y desplegar software Centos, Java, Mysql en el servidor.	No	No	No	No	No	Si	No	Si
Instalación de Nagios para monitoreo de servicios, tanto de sistema operativo como de base de datos.	Si	Si	No	No	Si	Si	Si	Si
Uso de monitoreo de Nagios sin haberlo instalado	No	No	Si	Si	No	No	No	No
Análisis de cada equipo de la capacitación en herramientas y procesos a automatizar	Si	Si	Si	Si	Si	Si	Si	Si
Análisis y planificación por equipo que permita compartir en tiempos cortos ideas innovadoras, procesos, herramientas	Si	Si	Si	Si	Si	Si	Si	Si
Identificación de conceptos de métricas DevOps a cumplir	Si	Si	Si	Si	Si	Si	Si	Si
Colaboración. Se analizó si en cada grupo existió relaciones colaborativas que apoyen el avance del trabajo	Si	Si	Si	Si	Si	Si	Si	Si
Tiempo de Ciclo. Total 5 iteraciones	20	20	20	20	20	20	20	20
Frecuencia de Implementación	100	100	100	100	100	100	100	100
Frecuencia real (minutos) utilizado en realizar 5 programas, por los equipos que utilizaron herramientas de integración continua	75	75			75	75	75	
Frecuencia real utilizado en realizar 5 programas, por los equipos que no utilizaron herramientas de integración continua			82	133				
Frecuencia real (minutos) utilizado en realizar 5 programas, por equipos que utilizaron herramientas de integración continua por cada programa								55
Fallos luego de una semana de monitoreo de las aplicaciones instaladas	No	No	No	No	No	No	No	No
Satisfacción del Usuario de resultados con herramientas de integración continua	Si	Si	No	No	Si	Si	Si	Si
Satisfacción del Usuario con Frecuencia real (minutos) utilizado en realizar 5 programas por cada equipo, utilizando herramientas de integración continua por cada programa.	No	No	No	No	No	No	No	Si

4.2 Conclusiones del trabajo realizado

El caso de estudio, con el trabajo de cada equipo permite concluir que:

- Los equipo C y D al no cumplir con la aplicación de conceptos de integración continua, demoraron más tiempo en desplegar sus aplicaciones.
- Los equipos A, B, E. F y G, a pesar que utilizaron conceptos y herramientas de integración continua en su desarrollo, no fue suficiente para satisfacer las necesidades del usuario.
- La Frecuencia real (minutos) utilizado en realizar 5 programas por el equipo H, por equipos que utilizaron conceptos y herramientas de integración continua por cada programa, establece que los 55 minutos.

- Se ha cumplido con los objetivos ya que la sincronización de los equipos de trabajo DevOps, que permitió el Análisis de Fronteras en los Roles de DevOps, como cultura, compartición, automatización y métricas y ayudaron a establecer en forma clara el factor determinante que tiene la relación de apoyo colaborativo de DevOps en todas las etapas del desarrollo de software, para alcanzar los objetivos corporativos en tiempos cortos.

5. Conclusiones

Se ha realizado un análisis de los Principios DevOps como Integración Continua, Entrega Continua, Despliegue Continuo, que permite visualizar la cultura como paradigma colaborativo en el desarrollo de software, y que junto a metodologías ágiles producen resultados eficaces en tiempos cortos y ordenados.

Se ha estudiado los Componentes en los que se desenvuelven las Fronteras y Roles DevOps con sus actividades: Cultura, Compartición, Automatización y Métricas, que permiten dar una idea clara de las buenas prácticas que hay que estructurar para llevar adelante en forma eficaz el desarrollo y despliegue de software en forma continua.

Se han definido los límites de DevOps que ayudan a e stablecer desde donde comienza la labor colaborativa del equipo de trabajo y hasta donde llega la responsabilidad luego del despliegue del todo el proyecto. Esto permite tener claro las responsabilidades de recuperación de fallos de las aplicaciones.

Se ha analizado los alcances del concepto de DevOps, que ayudan a entender los nuevos paradigmas de colaboración entre las áreas de desarrollo y operadores, para todo el proceso de desarrollo de software. De esta manera se logra que cada etapa de la metodología ágil en base a DevOps finalice en forma eficiente en tiempos cortos.

Se realizó una experiencia para analizar la aplicación de DevOps en cuanto a desarrollo y despliegue en alumnos de nivel preuniversitario con resultados interesantes en cuanto a la adopción de nueva forma trabajo, una nueva cultura, para muchos.

Se concluye que la elaboración de una hoja de ruta debe incluir estos pasos:

- Identificación de Componentes DevOps
  - Componentes de la Gestión de Desarrollo de Software con DevOps
    - ✓ Principios de DevOps
    - ✓ Integración, Entrega, Despliegue Continuo
  - Componentes de la Gestión de Despliegue en Data Center
  - Herramientas DevOps como apoyo al Desarrollo y Despliegue de un Proyecto
- Análisis de Fronteras en los Roles de DevOps
  - Cultura
  - Compartición
  - Automatización

Métricas

Referencias

1. Michael Hüttermann, DevOps of Developers, Integrate Development and Operations, The Agile Way, Available: Book of Apress
2. Jez Humble, David Farley, Continuous Delivery, Reliable Software, Releases Through Build, Test, and Deployment Automatic, Foreword y Martin Fowler. Available: https://buildrelease.googlecode.com/hg-history/1998cd1d530b35b79740d7bf93f8915548136c25/Trunk/BreBooks/Continuous%2520Delivery.pdf
3. Udo do Pracht, Menschen und IT, DevOps - Gemeinsam produktiv werden, Udo Pracht über Leute, Computer, Prozesse und i hr Zusammenspiel. Available: https:// menschenundit.wordpress.com/2011/05/25/devops/
4. Benno Luthiger, DevOps bei den ID Build-Automatisierung statt Silo-Betrieb, SWS Entwicklertreffen vom 1.10.2015
5. Robert Schulze, Cezar Adam, Consulting AG, Gegensätze Ziehen Sich An Oder Wie Man Mit Devops Erfolgreich Brücken Bauen Kann. Available: https://udf.de/files/it-consulting-unternehmensberatung/06-presse/02-veroeffentlichungen/mit-devops-erfolgreich-bruecken- bauen.pdf
6. OOP Software Meets Business (2012), DevOps, Available: http://www.sigs.de/download/oop_2012/files/Di2-2__Pracht_DevOps.pdf
7. Markus Guske, Michael Kloss, Release a D ay, Mythos oder Realität, Available: http://docplayer.org/1524009-Sonntag-4-oktober-2009- release-a-day-mythos-oder-realitaet.html.
8. Thomas Rümmler, Christian Schlag, DevOps und C ontinuous Delivery: sich gemeinsam kontinuierlich verbessern, Online Themenspecial DevOps 2014. Available: http://www.aitgmbh.de/fileadmin/user_upload/pdf/DevOpsUndContinousDelivery.pdf
9. Peter Gfader, Use Scrum + Continuous Delivery to build the right thing, Scrum.org. Available:https://www.scrum.org/Portals/0/Documents/Community%20Work/Scrum.org%20Whitepaper_Continuous%20Delivery.pdf
10. Christian Rodríguez, Lia Molinari, Francisco Javier Díaz (2013), The Hard Way to Virtual Machine, Administration: towards DevOps, A Bridge between Developers and IT Operators, LINTI (Laboratory of Investigation in New Information Technologies) National University of La Plata UNLP, December 2013.
11. Carlos Lobos Medina, Auditoría de Controles usando Cobit 5 e ISO 27002, Académico Universidad Diego Portales, CISA, CISM, Conferencia Latinoamericana CASC/ISRM 2014.
12. Jez Humble, David Farley, Continuous Delivery, R eliable Software Releases Through Build, Test, and Deployment Automation., Foreword by Martin Fowler, 2011
13. OOP Software Meets Business (2012), DevOps, Available: http://www.sigs.de/download/oop_2012/files/Di2-2__Pracht_DevOps.pdf
14. New Relic, Navigating Devops, What it is and why it matters to you and your business, ebook. 2014
15. Udo Pracht, Meine oder deine Architektur? Von der Entwicklung zum Betrieb mit DevOps, 2011
16. Peter Gfader, Use Scrum + Continuous Delivery to build the righ thing, Scrum.org, 2012
17. Michael Hûtterman, Agile ALM, Lightweight tools and Agile strategies, 2011
18. IEEE Software Engineering Standards, Standard 610.12-1990, 1993