



Una guía para el

CONOCIMIENTO DE SCRUM (GUÍA SBOK™)

2016 Edición

A Guide to the SCRUM BODY OF KNOWLEDGE (SBOK™ Guide)

Una guía completa para la entrega de proyectos utilizando Scrum



Una guía para el

CONOCIMIENTO DE SCRUM

(Guía SBOK™)

2016 Edición

A Guide to the SCRUM BODY OF KNOWLEDGE (*SBOK™ Guide*)

Una guía completa para la entrega de proyectos utilizando Scrum

© 2016 SCRUMstudy™, una marca de VMEdu, Inc. Todos los derechos reservados.

Library of Congress Cataloging-in-Publication Data

Una guía para el conocimiento de Scrum (Guía SBOK™) - 2016 Edición

Título original: *A Guide to the SCRUM BODY OF KNOWLEDGE (SBOK™GUIDE) 2016 Edition*

Incluye referencia bibliográfica e índice.

ISBN: 978-0-9899252-0-4

1. Marco de Scrum.I. SCRUMstudy™.II. *Guía SBOK™*

2013950625

ISBN: 978-0-9899252-0-4

Publicado por:

SCRUMstudy™, una marca de VMEdu, Inc.

410 N 44th Street, Suite 240

Phoenix, Arizona 85008 USA

Teléfono: +1-480-882-0706

Fax: +1-240-238-2987

Correo Electrónico: sbok@scrumstudy.com

Sitio Web: www.scrumstudy.com

"SBOK", el logotipo de SCRUMstudy, "SDC", "SMC", "AEC", "SPOC", y "ESM" son marcas comerciales de SCRUMstudy™ (una marca de VMEdu, Inc.) Para obtener una lista completa de las marcas SCRUMstudy™, póngase en contacto con el Departamento Legal SCRUMstudy™.

Se proporciona *Una guía para el conocimiento de Scrum (Guía SBOK™)* con fines educativos. SCRUMstudy™ o VMEdu, Inc. no garantizan que es conveniente para ningún otro propósito y no hace ninguna garantía expresa o implícita de ningún tipo y no asume ninguna responsabilidad por errores y omisiones. No se asume responsabilidad por daños incidentales o consecuentes relacionados con o que surjan de la utilización de la información aquí contenida.

SCRUMstudy™ da la bienvenida a las correcciones y los comentarios sobre sus libros. No dude en enviar comentarios sobre errores tipográficos, de formato, u otros errores. Usted puede hacer una copia de la página correspondiente del libro, marcar el error, y enviarlo a la dirección antes mencionada o enviar un correo electrónico a sbok@scrumstudy.com.

Ninguna parte de este trabajo puede ser reproducido o transmitido en cualquier forma o por cualquier medio, ya sea electrónico, manual, fotocopia, grabación o por cualquier sistema de almacenamiento y recuperación, sin el permiso previo y por escrito de la editorial.

PRÓLOGO

Una guía para el conocimiento de Scrum (Guía SBOK™) proporciona directrices para la aplicación con éxito de Scrum-el desarrollo de productos Agile más popular y la metodología de ejecución de proyectos. Scrum, tal como se define en la *Guía SBOK™*, es un marco que se aplica a *portfolios*, programas o proyectos de cualquier tamaño y complejidad; y se puede aplicar de manera efectiva en *cualquier* industria para crear un producto, servicio, o cualquier otro resultado.

La *Guía SBOK™* se desarrolló como una guía estándar para profesionales y organizaciones de negocios que deseen implementar Scrum, así como para los que ya lo estén utilizando y quieran aun mejorar el proceso. Está diseñado para ser utilizado como referencia y guía de conocimiento tanto por profesionales con experiencia en Scrum, para profesionales de productos o servicios de desarrollo, así como para personas que no tengan experiencia previa o conocimiento de Scrum o cualquier otra metodología de entrega del proyecto.

La *Guía SBOK™* se basa en el conocimiento y la visión combinada obtenida de miles de proyectos a través de una variedad de organizaciones e industrias. Además, las contribuciones han sido hechas por los expertos que han enseñado Scrum y cursos de entrega de proyectos a más de 400.000 profesionales en 150 países. Su desarrollo ha sido verdaderamente un esfuerzo de colaboración de un gran número de expertos en una variedad de disciplinas. En particular, me gustaría dar las gracias a los diecisiete autores y expertos en la materia y los veintiocho correctores que contribuyeron en gran medida a la creación de la *Guía SBOK™*.

La amplia adopción del marco *Guía SBOK™* ayudará a estandarizar la aplicación de Scrum a todo tipo de proyecto a través de las organizaciones a nivel mundial, al igual que ayudará a mejorar significativamente su *return on investment*. Además, promoverá una mayor reflexión y deliberación sobre la aplicación de Scrum para muchos tipos de proyectos, que a su vez contribuirá a ampliar y enriquecer el acervo de conocimientos y consecuentemente actualizaciones futuras de esta guía.

Aunque la *Guía SBOK™* es una guía completa y marco para la entrega de proyectos que utilizan Scrum, su contenido está organizado para una referencia fácil, independientemente de los conocimientos previos del lector sobre el tema. Espero que cada lector pueda aprender y disfrutar tanto como los autores y correctores aprendieron y disfrutaron del proceso de cotejo de los conocimientos y la sabiduría colectiva aquí contenida.



Tridibesh Satpathy,

Autor principal, *Guía SBOK™*

TABLA DE CONTENIDO

1.	INTRODUCCIÓN	1
1.1	Información general de Scrum.....	2
1.1.1	Breve historia de Scrum	3
1.2	¿Por qué utilizar Scrum?	4
1.2.1	Escalabilidad de Scrum	5
1.3	Propósito de la <i>Guía SBOK™</i>	6
1.4	Marco de la <i>Guía SBOK™</i>	7
1.4.1	¿Cómo usar la <i>Guía SBOK™</i> ?	8
1.4.2	<i>Scrum Principles</i>	9
1.4.3	<i>Aspectos de Scrum</i>	11
1.4.4	Procesos de Scrum	16
1.5	Scrum vs gestión de proyectos tradicional.....	20
2.	PRINCIPIOS	21
2.1	Introducción	21
2.2	Guía de Roles	22
2.3	<i>Control del proceso empírico</i>	22
2.3.1	<i>Transparencia</i>	22
2.3.2	<i>Inspection</i>	24
2.3.3	<i>Adaptation</i>	24
2.4	Auto-organización.....	27
2.4.1	Beneficios de Auto-organización.....	27
2.5	Colaboración.....	30
2.5.1	Beneficios de Colaboración en Proyectos Scrum	30
2.5.2	La importancia de <i>Colocation</i> en Colaboración.....	32
2.6	Prioritization basado en valor.....	33
2.7	Tiempo asignado	34
2.7.1	<i>Scrum Time-boxes</i>	35

2.8	<i>Desarrollo iterativo</i>	38
2.9	Scrum vs Gestión tradicional de proyectos.....	40
3.	ORGANIZACIÓN.....	41
3.1	Introducción	41
3.2	Guía de los roles.....	42
3.3	Roles de un proyecto Scrum.....	42
3.3.1	Core Roles.....	43
3.3.2	Non-core Roles	44
3.4	<i>Propietario del producto</i>	45
3.4.1	<i>Voice of the Customer (VOC)</i>	47
3.4.2	Jefe Propietario del producto.....	47
3.5	Scrum Master.....	48
3.5.1	Jefe Scrum Master.....	49
3.6	<i>Equipo Scrum</i>	50
3.6.1	Selección de Personal.....	51
3.6.2	Tamaño del <i>Equipo Scrum</i>	52
3.7	Scrum en Proyectos, Programas, y Portfolios.....	52
3.7.1	Definición de Proyecto, Programa, y Portfolio.....	52
3.7.2	Scrum en Proyectos	53
3.7.3	Scrum en <i>Porfolios y Programs</i>	55
3.7.4	El mantenimiento de la participación de los <i>Socios</i>	57
3.8	Resumen de responsabilidades.....	58
3.9	Scrum vs gestión de proyectos tradicional.....	59
3.10	Las teorías de HR populares y su relevancia para Scrum.....	60
3.10.1	Modelo de Dinámica de Grupo de Tuckman.....	60
3.10.2	Manejo de conflictos	61
3.10.3	Técnicas de <i>Conflict Management</i>	61
3.10.4	Estilos de liderazgo	64
3.10.5	La Teoría de Jerarquía de Necesidades de Maslow	66
3.10.6	<i>Theory X and Theory Y</i>	67
4.	JUSTIFICACIÓN DEL NEGOCIO	68

4.1	Introducción	68
4.2	Guía de Roles	69
4.3	<i>Entrega basada en valor</i>	69
4.3.1	Las responsabilidades del <i>Propietario del producto</i> en la creación de <i>Justificación del negocio</i>	71
4.3.2	Responsabilidades de los otros roles de Scrum en <i>Justificación del negocio</i>	71
4.4	Importancia de <i>Justificación del negocio</i>	72
4.4.1	Factores utilizados para determinar <i>Justificación del negocio</i>	72
4.4.2	<i>Justificación del negocio</i> y el ciclo de vida del proyecto	73
4.5	Técnicas de <i>Justificación del negocio</i>	75
4.5.1	Estimación del valor del proyecto	75
4.5.2	<i>Planning for Value</i>	78
4.5.3	<i>Relative Prioritization Ranking</i>	80
4.5.4	<i>Story Mapping</i>	80
4.6	<i>Continuous Value Justification</i>	80
4.6.1	<i>Earned Value Analysis</i>	81
4.6.2	Cumulative Flow Diagram (CFD).....	84
4.7	Confirmar la realización de beneficios	85
4.7.1	Prototipos, simulaciones y demostraciones.....	85
4.8	Resumen de responsabilidades.....	86
4.9	Scrum vs gestión de proyectos tradicional.....	87
5.	CALIDAD (<i>QUALITY</i>)	88
5.1	Introducción	88
5.2	Guía de los roles.....	89
5.3	Definición de Calidad (<i>Quality</i>)	89
5.3.1	Calidad y Alcance (<i>Qualiaty and Scope</i>)	90
5.3.2	Calidad y Valor empresarial.....	90
5.4	<i>Acceptance Criteria</i> y <i>Prioritized Product Backlog</i>	91
5.4.1	Escribiendo <i>Acceptance Criteria</i>	92
5.4.2	<i>Minimum Acceptance Criteria</i>	93
5.4.3	Definición de <i>Done</i> (Terminado)	95

5.4.4	Aceptación o rechazo de elementos de <i>Prioritized Product Backlog</i>	95
5.5	<i>Quality Management</i> en Scrum.....	96
5.5.1	<i>Quality Planning</i>	96
5.5.2	<i>Quality Control</i> y <i>Quality Assurance</i>	99
5.5.3	<i>Plan-Do-Check-Act (PDCA) Cycle</i>	100
5.6	Resumen de responsabilidades.....	101
5.7	Scrum vs gestión de proyectos tradicional.....	102
6.	Cambio.....	104
6.1	Introducción	104
6.2	Guía de los roles.....	105
6.3	Descripción	105
6.3.1	<i>Unapproved and Approved Change Requests</i>	106
6.4	Cambio en Scrum	108
6.4.1	Equilibrio entre flexibilidad y estabilidad.....	108
6.4.2	El logro de la flexibilidad.....	108
6.5	Integración del cambio.....	114
6.5.1	Los cambios a un <i>Sprint</i>	114
6.6	Cambio en <i>Portfolios</i> y <i>Programs</i>	119
6.6.1	En <i>Portfolio</i>	120
6.6.2	En <i>Program</i>	120
6.7	Resumen de responsabilidades.....	122
6.8	Scrum vs gestión de proyectos tradicional.....	123
7.	RIESGO	125
7.1	Introducción	125
7.2	Guía de roles	126
7.3	¿Qué es <i>Risk</i> ?	126
7.3.1	Diferencia entre <i>Risks</i> e <i>Issues</i>	126
7.3.2	<i>Risk Attitude</i>	127
7.4	Procedimiento de gestión de <i>risks</i>	128
7.4.1	<i>Risk Identification</i>	128
7.4.2	<i>Risk Assessment</i>	129

7.4.3	<i>Risk Prioritization</i>	132
7.4.4	Risk Mitigation.....	135
7.4.5	<i>Risk Communication</i>	135
7.5	Reducción al mínimo de <i>Risks</i> a través de Scrum.....	137
7.6	<i>Risks en Portfolios y Programs</i>	138
7.6.1	<i>En Portfolio</i>	138
7.6.2	<i>En Program</i>	138
7.7	Resumen de Responsabilidades	140
7.8	Scrum vs Gestión de proyectos tradicional.....	141
8.	INICIAR	142
8.1	<i>Crear la visión del proyecto</i>	146
8.1.1	Entradas.....	148
8.1.2	Herramientas.....	152
8.1.3	Salidas.....	153
8.2	<i>Identificar al Scrum Master y al socio(s)</i>	155
8.2.1	Entradas.....	157
8.2.2	Herramientas.....	159
8.2.3	Salidas.....	161
8.3	<i>Formación de un equipo Scrum</i>	162
8.3.1	Entradas.....	164
8.3.2	Herramientas.....	165
8.3.3	Salidas.....	166
8.4	<i>Desarrollo de épica(s)</i>	168
8.4.1	Entradas.....	170
8.4.2	Herramientas.....	173
8.4.3	Salidas.....	175
8.5	<i>Creación de la lista priorizada de pendientes del producto</i>	177
8.5.1	Entradas.....	179
8.5.2	Herramientas.....	180
8.5.3	Salidas	182
8.6	<i>Realizar el plan de lanzamiento</i>	184

8.6.1	Entradas.....	186
8.6.2	Herramientas.....	187
8.6.3	Salidas.....	188
8.7	Fase <i>diagrama de flujo de datos</i>	190
9.	PLANEAR Y ESTIMAR	191
9.1	<i>Elaborar historias de usuario</i>	195
9.1.1	Entradas.....	197
9.1.2	Herramientas.....	198
9.1.3	Salidas.....	200
9.2	<i>Aprobar, estimar y asignar historias de usuarios</i>	202
9.2.1	Entradas.....	203
9.2.2	Herramientas.....	203
9.2.3	Salidas.....	206
9.3	<i>Elaboración de tareas</i>	207
9.3.1	Entradas.....	208
9.3.2	Herramientas.....	208
9.3.3	Salidas.....	210
9.4	<i>Estimar tareas</i>	212
9.4.1	Entradas.....	213
9.4.2	Herramientas.....	214
9.4.3	Salidas.....	215
9.5	<i>Elaboración de la lista de pendientes del Sprint</i>	216
9.5.1	Entradas.....	218
9.5.2	Herramientas.....	219
9.5.3	Salidas.....	220
9.6	Fase Diagrama de flujo de datos	221
10.	IMPLEMENTAR	222
10.1	<i>Crear entregables</i>	226
10.1.1	Entradas.....	228
10.1.2	Herramientas.....	230
10.1.3	Salidas.....	231

10.2 <i>Llevar a cabo el Standup diario</i>	233
10.2.1 Entradas.....	234
10.2.2 Herramientas.....	235
10.2.3 Salidas.....	236
10.3 <i>Mantenimiento de la lista priorizada de pendientes del producto</i>	238
10.3.1 Entradas.....	240
10.3.2 Herramientas.....	242
10.3.3 Salidas.....	243
10.4 Fase Diagrama de flujo de Datos.....	244
11. REVISIÓN Y RETROSPECTIVA	246
11.1 <i>Convocar Scrum de Scrums</i>	250
11.1.1 Entradas.....	251
11.1.2 Herramientas.....	252
11.1.3 Salidas.....	254
11.2 <i>Demostración y validación del Sprint</i>	255
11.2.1 Entradas.....	257
11.2.2 Herramientas.....	258
11.2.3 Salidas.....	259
11.3 <i>Retrospectiva de Sprint</i>	261
11.3.1 Entradas.....	263
11.3.2 Herramientas.....	263
11.3.3 Salidas.....	265
11.4 Fase Diagrama de flujo de Datos.....	267
12. LANZAMIENTO	268
12.1 <i>Envío de entregables</i>	272
12.1.1 Entradas.....	273
12.1.2 Herramientas.....	274
12.1.3 Salidas.....	275
12.2 <i>Retrospectiva del proyecto</i>	276
12.2.1 Entradas.....	277
12.2.2 Herramientas.....	278

TABLE OF CONTENTS

12.2.3 Salidas.....	279
12.3 Fase Diagrama de flujo de datos	280
APÉNDICE A. RESUMEN DE AGILE.....	282
APÉNDICE B. AUTORES Y REVISORES DE LA GUÍA SBOK™	292
REFERENCIAS	294
GLOSARIO.....	296
INDEX.....	334

LISTA DE FIGURAS

Figura 1-1: Flujo de Scrum para un <i>Sprint</i>	2
Figura 1-2: <i>SBOK™ Guía del Marco</i>	7
Figura 1-3: <i>Scrum Principles</i>	9
Figura 1-4: Organización en Scrum.....	13
Figura 2-1: <i>Transparencia</i> en Scrum	23
Figura 2-2: <i>Inspection</i> en Scrum.....	24
Figura 2-3: <i>Adaptation</i> en Scrum.....	25
Figura 2-4: Retos en la Gestión de Proyectos Tradicional	26
Figura 2-5: Objetivos de un equipo de auto-organización	29
Figura 2-6: Beneficios de <i>Colaboración</i> en proyectos Scrum.....	32
Figura 2-7: <i>Value-based Prioritization</i>	34
Figura 2-8: Duración de <i>Time-Box</i> para las reuniones de Scrum.....	37
Figura 2-9: Scrum vs Cascada tradicional.....	39
Figura 3-1: Roles de Scrum - Descripción General	44
Figura 3-2: Las preguntas formuladas durante un <i>Scrum of Scrums Meeting</i>	49
Figura 3-3: Características deseadas de los papeles principales de Scrum	51
Figura 3-4: Reunión de <i>Scrum of Scrums (SoS)</i>	54
Figura 3-5: Scrum en toda la organización para <i>projects, programs, y portfolios</i>	56
Figura 3-6: Etapas de Tuckman de Desarrollo de Grupos	60
Figura 3-7: Teoría de Jerarquía de Necesidades de Maslow	66
Figura 4-1: Entrega de valor en Scrum vs Proyectos Tradicionales	71
Figura 4-2: Jerarquía de responsabilidades de <i>Justificación del negocio</i>	71
Figura 4-3: <i>Justificación del negocio</i> y el Ciclo de Vida del Proyecto	75
Figura 4-4: <i>Kano Analysis</i>	79
Figura 4-5: Ejemplo de <i>Cumulative Flow Diagram (CFD)</i>	84
Figura 5-1: Diagrama de Flujo del Incremento del Proyecto.....	92
Figura 5-2: Acceptance Criteria en Cascada (<i>Waterfall</i>)	93
Figura 5-3: PDCA Cycle en Scrum	100

Figura 6-1: Ejemplo de Proceso de Cambio de Aprobación	107
Figura 6-2: Actualización de <i>Prioritized Product Backlog</i> con los cambios aprobados.....	107
Figura 6-3: Características de Scrum para lograr flexibilidad	109
Figura 6-4: Motivación de los <i>socios</i> para la solicitud de cambios	110
Figura 6-5: La motivación del Equipo Central de Scrum para solicitar cambios	111
Figura 6-6: Integración del cambio en Scrum	115
Figura 6-7: Impacto del cambio esperado en <i>Length of Sprint</i>	117
Figura 6-8: La incorporación de cambios en el <i>portfolio y program</i>	121
Figura 7-1: Muestra de <i>Probability Tree</i>	130
Figura 7-2: Ejemplo del diagrama de Pareto	131
Figura 7-3: Muestra de Matriz de Probabilidad e Impacto	132
Figura 7-4: Ilustra el proceso de <i>Risk Prioritization</i>	134
Figura 7-5: Ejemplo de <i>Risk Burndown Chart</i>	136
Figura 7-6: Manejo de <i>risks</i> en <i>portfolios y programs</i>	139
Figura 8-1: Información general sobre <i>Initiate</i>	144
Figura 8-2: Información general de <i>Initiate</i> (Esenciales)	145
Figura 8-3: <i>Crear la visión del proyecto</i> – Entradas, Herramientas y Salidas.....	146
Figura 8-4: El diagrama de flujos de datos del <i>Crear la visión del proyecto</i>	147
Figura 8-5: El proceso de <i>Gap Analysis</i>	153
Figura 8-6: <i>Identificar al Scrum Master y al socio(s)</i> —Entradas, Herramientas y Salidas.....	155
Figura 8-7: <i>Identificar al Scrum Master y al socio(s)</i> —Diagrama de flujo de datos	156
Figura 8-8: <i>Formación de un equipo Scrum</i> —Entradas, Herramientas, y Salidas.....	162
Figura 8-9: <i>Formación de un equipo Scrum</i> —Diagrama de flujo de datos	163
Figura 8-10: <i>Desarrollo de épica(s)</i> —Entradas, Herramientas y Salidas	168
Figura 8-11: <i>Desarrollo de épica(s)</i> —Diagrama de flujo de datos	169
Figura 8-12: <i>Creación de la lista priorizada de pendientes del producto</i> —Entradas, Herramientas y Salidas	177
Figura 8-13: <i>Creación de la lista priorizada de pendientes del producto</i> — Diagrama de flujo de datos.....	178
Figura 8-14: <i>Realizar el plan de lanzamiento</i> —Entrada, Herramientas y Salidas	184
Figura 8-15: <i>Realizar el plan de lanzamiento</i> —Diagrama de flujo de datos	185
Figura 8-16: <i>Initiate Phase</i> —Diagrama de flujo de datos	190
Figura 9-1: Resumen de <i>Plan and Estimate</i>	193

Figura 9-2: Resumen de <i>Plan and Estimate</i> (Esenciales)	194
Figura 9-3: <i>Elaborar historias de usuario</i> —Entradas, Herramientas, y Salidas	195
Figura 9-4: <i>Elaborar historias de usuario</i> —Diagrama de flujo de datos	196
Figura 9-5: <i>Aprobar, estimar y asignar historias de usuarios</i> —Entrada, Herramientas y Salidas.....	202
Figura 9-6: <i>Aprobar, estimar y asignar historias de usuarios</i> — Diagrama de flujo de datos	202
Figura 9-7: <i>Elaboración de tareas</i> —Entradas, Herramientas y Salidas.....	207
Figura 9-8: <i>Elaboración de tareas</i> —Diagrama de flujo de dato	208
Figura 9-9: <i>Task Planning Meetings</i>	209
Figura 9-10: <i>Estimar tareas</i> —Entradas, Herramientas y Salidas	212
Figura 9-11: <i>Estimar tareas</i> —Diagrama de flujo de datos	212
Figura 9-12: <i>Elaboración de la lista de pendientes del Sprint</i> —Entradas, Herramientas y Salidas	216
Figura 9-13: <i>Elaboración de la lista de pendientes del Sprint</i> —Diagrama de flujo de dato	217
Figura 9-14: <i>Plan and Estimate</i> Fase—Diagrama de Ffijo de datos.....	221
Figura 10-1: <i>Implement</i> - Resumen	224
Figura 10-2: Repaso de <i>Implement</i> (Esenciales).....	225
Figura 10-3: <i>Crear entregables</i> —Entradas, Herramientas y Salidas	226
Figura 10-4: <i>Crear entregables</i> —Diagrama de flujo de datos	227
Figura 10-5: <i>Scrumboard</i>	228
Figura 10-6: <i>Llevar a cabo el Standup diario</i> —Entradas, Herramientas y Salidas	233
Figura 10-7: <i>Llevar a cabo el Standup diario</i> —Diagrama de flujo de datos.....	233
Figura 10-8: <i>Mantenimiento de la lista priorizada de pendientes del producto</i> —Entradas, Herramientas y Salidas	238
Figura 10-9: <i>Mantenimiento de la lista priorizada de pendientes del producto</i> —Diagrama de flujo de datos	239
Figura 10-10: Implementar la Fase—Diagrama de flujo de datos	244
Figura 11-1: Resumen de Revisión y Retrospectiva.....	248
Figura 11-2: Resumen de Revisión y Retrospectiva (Esenciales)	249
Figura 11-3: <i>Convocar Scrum de Scrums</i> —Entradas, Herramientas y Salidas.....	250
Figura 11-4: <i>Convocar Scrum de Scrums</i> —Diagrama de flujo de datos	250
Figura 11-5: <i>Demostración y validación del Sprint</i> —Entradas, Herramientas y Salidas.....	255
Figura 11-6: <i>Demostración y validación del Sprint</i> —Diagrama de flujo de datos	256
Figura 11-7: <i>Retrospectiva de Sprint</i> —Entradas, Herramientas y Salidas	261

Figura 11-8: <i>Retrospectiva de Sprint</i> —Diagrama de flujo de datos	262
Figura 11-9: <i>Fase Review and Retrospect</i> —Diagrama de flujo de datos	267
Figura 12-1: Resumen de <i>Release</i> (Lanzamiento)	270
Figura 12-2: Resumen de <i>Release</i> (Esenciales)	271
Figura 12-3: <i>Envío de entregables</i> —Entradas, Herramientas y Salidas.....	272
Figura 12-4: <i>Envío de entregables</i> —Diagrama de flujo de datos	272
Figura 12-5: <i>Retrospectiva del proyecto</i> —Entrada, Herramientas y Salidas	276
Figura 12-6: <i>Retrospectiva del proyecto</i> —Diagrama de flujo de datos.....	276
Figura 12-7: <i>Release Phase</i> (Fase de lanzamiento)—Diagrama de flujo de datos	280

LISTA DE TABLAS

Tabla 1-1: Resumen de los procesos de Scrum.....	16
Tabla 1-2: Scrum vs. Gestión de proyectos tradicional	20
Tabla 3-1: Responsabilidades del <i>Propietario del producto</i> en los procesos de Scrum	47
Tabla 3-2: Responsabilidades del <i>Scrum Master</i> en los procesos de Scrum.....	49
Tabla 3-3: Responsabilidades del <i>Equipo Scrum</i> en los procesos de Scrum.....	51
Tabla 3-4: Resumen de las responsabilidades pertinentes a la organización	58
Tabla 4-1: Fórmulas de Valor Ganado	82
Tabla 4-2: Resumen de las responsabilidades pertinentes a <i>Justificación del negocio</i>	86
Tabla 5-1: Resumen de las responsabilidades pertinentes a la Calidad	101
Tabla 6-1: Resumen de las responsabilidades relacionadas con el Cambio.....	122
Tabla 7-1: Resumen de las responsabilidades pertinentes a <i>Risks</i>	140

1. INTRODUCCIÓN

Una guía para el conocimiento de Scrum (*Guía SBOK™*) proporciona las pautas para la implementación exitosa de la metodología Scrum-la gestión de projects *agile* y la metodología de desarrollo de productos más popular. Este libro proporciona un marco integral que incluya los principios, aspectos y procesos de Scrum.

Scrum, tal como se define en la *Guía SBOK™*, es aplicable a los siguientes:

- *Portfolios, programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se les entregará a los *Shareholders*
- *Projects* de cualquier tamaño y complejidad

El término "producto" (*product*) en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria-desde proyectos pequeños o equipos con tan sólo seis miembros, hasta proyectos grandes y complejos con cientos de miembros por equipo.

Este primer capítulo describe la finalidad y el marco de la *Guía SBOK™* y proporciona una introducción a los conceptos claves de Scrum. Contiene un resumen de los principios de Scrum, al igual que aspectos y procesos del mismo. El capítulo 2 expande en los seis principios de Scrum, que son la base de Scrum. Los capítulos 3 al 7 elaboran sobre los cinco aspectos de Scrum que deben ser abordados a través de cualquier proyecto: la organización, *Justificación del negocio*, la calidad, el cambio y el riesgo. Los capítulos 8 al 12 cubren los 19 procesos de Scrum que forman parte en la creación de un proyecto Scrum. Estos procesos forman parte de las cinco fases de Scrum: Iniciar; Planificar y Estimar; Implementar, Revisión y Retrospectiva; y Lanzamiento. Estas fases describen en detalle las entradas y salidas asociadas de cada proceso, así como las diferentes herramientas que se pueden utilizar en cada proceso. Algunas entradas, herramientas y salidas son obligatorias y se indican como tales; otras son opcionales dependiendo del proyecto específico, los requisitos de organización y/o lineamientos establecidos por *Cuerpo de asesoramiento de Scrum* de la organización (Scrum Guidance Body - SGB). Por último, el Apéndice A contiene una descripción general de *Agile Manifesto* (Fowler y Highsmith, 2001) y un análisis de los diversos métodos ágiles para los que quieren más información acerca de Agile.

Este capítulo está dividido en las siguientes secciones:

1.1 Información general de Scrum

1.2 ¿Por qué utilizar Scrum?

1.3 Propósito de la *Guía SBOK™*

1.4 Marco de la *Guía SBOK™*

1.5 Scrum vs Gestión de Proyectos Tradicional

1.1 Información general de Scrum

Un proyecto (*project*) Scrum implica un esfuerzo de colaboración para crear un nuevo producto (*product*), servicio, o cualquier otro resultado como se define en el *Declaración de la Visión del Proyecto*. Los *projects* se ven afectados por las limitaciones de tiempo, costo, alcance, calidad, recursos, capacidades organizativas, y otras limitaciones que los hacen difíciles de planificar, ejecutar, administrar y finalmente tener éxito. Sin embargo, la implementación exitosa de los resultados de un *project* acabado le proporciona ventajas económicas significativas a una organización. Por lo tanto, es importante que las organizaciones seleccionen y practiquen una metodología adecuada de gestión de *projects*.

Scrum es una de las metodologías ágiles más populares. Es una metodología de adaptación, iterativa, rápida, flexible y eficaz, diseñada para ofrecer un valor significativo de forma rápida en todo el proyecto. Scrum garantiza transparencia en la comunicación y crea un ambiente de responsabilidad colectiva y de progreso continuo. El marco de Scrum, tal como se define en la *Guía SBOK™*, está estructurado de tal manera que es compatible con los productos y el desarrollo de servicio en todo tipo de industrias y en cualquier tipo de proyecto, independientemente de su complejidad.

Una fortaleza clave de Scrum radica en el uso de equipos multi-funcionales, auto-organizados, y con poder que dividen su trabajo en ciclos de trabajo cortos y concentrados llamados *Sprints*. Figura 1-1 proporciona una visión general de flujo de un *project* Scrum.

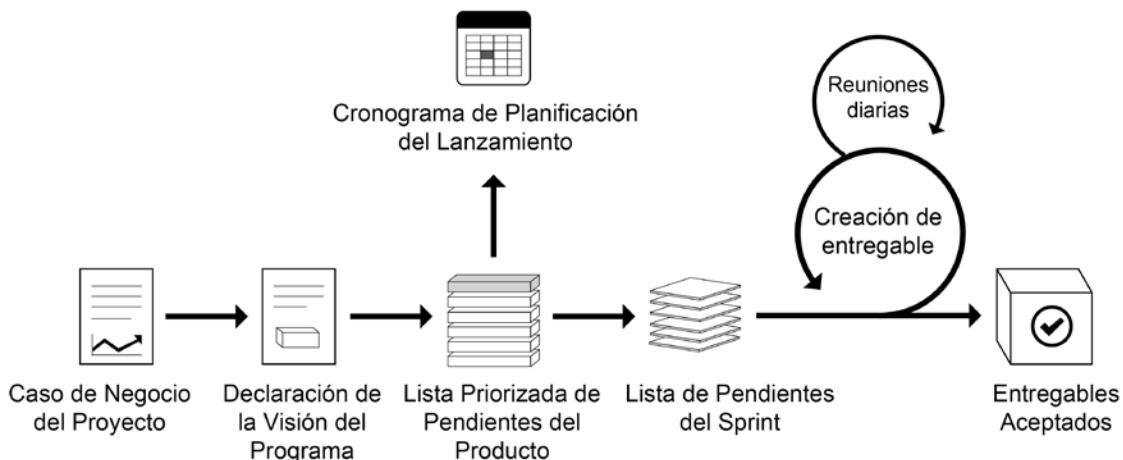


Figura 1-1: Del Flujo de Trabajo en Scrum

El ciclo de Scrum comienza con un *Stakeholder Meeting*, durante el cual se crea la visión del *project*. El *Propietario del producto* (*product owner*), entonces desarrolla un *Prioritized Product Backlog* que contiene una lista priorizada de los requerimientos del negocio en forma de *User Story*. Cada *Sprint* comienza con un *Sprint Planning Meeting* durante el cual los *User Stories* de alta prioridad son considerados para su inclusión en el *Sprint*.

Un *Sprint* suele durar entre una y seis semanas en el cual el *Equipo Scrum* trabaja en la creación Entregables (Deliverables) potencialmente listos en incrementos del *product*. Durante el *Sprint*, se llevan a cabo *Daily Standup Meetings* cortos y muy concretos donde los miembros del equipo discuten progresos diarios. A medida que concluye el *Sprint*, un *Sprint Planning Meeting* se lleva a cabo en el cual al *Propietario del producto* y a los *Socios* relevantes se les proporciona una demostración de los bienes y servicios. El *Propietario del producto* acepta las entregas sólo si cumplen con los *Acceptance Criteria* predefinidos. El ciclo de *Sprint* termina con un *Retrospectiva de Sprint Meeting*, donde el equipo presenta modos para mejorar los procesos y el rendimiento a medida que avanzan al siguiente *Sprint*.

1.1.1 Breve historia de Scrum

A mediados de la década de los 80, Hirotaka Takeuchi y Ikujiro Nonaka definieron una estrategia de desarrollo de producto flexible e incluyente en la que el equipo de desarrollo trabaja como una unidad para alcanzar un objetivo común. Hirotaka Takeuchi y Ikujiro Nonaka describieron un enfoque innovador para el desarrollo de productos que ellos llaman un enfoque holístico o "rugby", "donde un equipo intenta llegar hasta el final como una unidad, pasando el balón hacia atrás y adelante." (*where a team tries to go the distance as a unit, passing the ball back and forth.*) Ellos basan su enfoque en los estudios de casos de diversas industrias de fabricación. Takeuchi y Nonaka propusieron que el desarrollo de productos no debe ser como una carrera de relevos secuencial, sino que debería ser análogo al del juego de rugby en el que el equipo trabaja en conjunto, pasando el balón hacia atrás y hacia adelante a medida que se mueve como una unidad por el campo. El concepto de rugby de un "Scrum" (donde un grupo de jugadores se junta para reiniciar el juego) se introdujo en este artículo para describir la propuesta de los autores de que el desarrollo de productos debe implicar "mover al Scrum campo abajo" (*moving the Scrum downfield*).

Ken Schwaber y Jeff Sutherland elaboraron sobre el concepto de Scrum y su aplicabilidad al desarrollo de software durante una presentación en la conferencia *Object-Oriented Programming, Systems, Languages & Applications (OOPSLA)* en 1995 en Austin, Texas. Desde entonces, varios practicantes, expertos y autores de Scrum han seguido perfeccionando la conceptualización y metodología de Scrum. En los últimos años, Scrum ha aumentado en popularidad y ahora es la metodología de desarrollo de proyectos preferida por muchas organizaciones a nivel mundial.

1.2 ¿Por qué utilizar Scrum?

Algunas de las ventajas principales de la utilización de Scrum en cualquier proyecto son:

1. **Adaptabilidad**—*Control del proceso empírico e Desarrollo iterativo* hacen que los *projects* sean adaptables y abiertos a la incorporación del cambio.
2. **Transparencia**—Todos los radiadores de información tal como un *Scrumboard* y *Sprint Burndown Chart* son compartidos, lo que lleva a un ambiente de trabajo abierto.
3. **Retroalimentación Continua**—Retroalimentación continua se proporciona a través de los procesos llamados *Llevar a cabo el Standup diario* y *Demostración y validación del Sprint*.
4. **Continuous Improvement**—Los entregables se mejoran progresivamente *Sprint* por *Sprint* a través del proceso *Mantenimiento de la lista priorizada de pendientes del producto*.
5. **Entrega Continúa de Valor**—los procesos iterativos permiten la entrega continua de valor tan frecuentemente como el *customer* lo requiere a través del proceso *Ship Deliverable*.
6. **Sustainable Pace**— Los procesos Scrum están diseñados de tal manera que las personas involucradas pueden trabajar a un paso cómodo (*sustainable pace*) que, en teoría, se puede continuar indefinidamente.
7. **Entrega Anticipada de Alto Valor**—El proceso de Creación de la lista priorizada de pendientes del producto asegura que los requisitos de mayor valor del *Customer* sean los primeros en cubrirse.
8. **Proceso de Desarrollo Eficiente**—*Tiempo asignado* y la reducción al mínimo de trabajo que no es esencial conducen a mayores niveles de eficiencia.
9. **Motivación**—Los procesos de *Llevar a cabo el Standup diario* y *Retrospectiva de Sprint* conducen a mayores niveles de motivación entre los empleados.
10. **Resolución de Problemas de Forma más Rápida**—*Colaboración* y *Colocation* de equipos multi-funcionales conducen a la resolución de problemas con mayor rapidez.
11. **Entregables Efectivos**—El proceso de *Creación de la lista priorizada de pendientes del producto* y revisiones periódicas después de la creación de entregables asegura entregas efectivas para el *Customer*.
12. **Centrado en el Customer (cliente)**— El poner énfasis en el valor del negocio y tener un enfoque de colaboración con los socios asegura un marco orientado al *customer*.
13. **Entorno de Alta Confianza**—Los procesos de *Llevar a cabo el Standup diario* and *Retrospectiva de Sprint* promueven transparencia y *colaboration*, dando lugar a un ambiente de trabajo de alta confianza, asegurando así una baja fricción entre los empleados.

14. **Responsabilidad Colectiva**—El proceso de *Approve, Estimate and Commit User Stories* permite que los miembros del equipo se sientan responsables del proyecto y su trabajo resultando en una mejor calidad.
15. **Alta Velocidad**—Un marco de colaboración que le permite a los equipos multi-funcionales altamente cualificados alcanzar su potencial y alta velocidad.
16. **Medio Ambiente Innovador**—Los procesos *Retrospectiva de Sprint* y *Retrospectiva del proyecto* crean un ambiente de introspección, aprendizaje y capacidad de adaptación que lleva a un entorno de trabajo innovador y creativo.

1.2.1 Escalabilidad de Scrum

Para ser eficaz, los *Equipos Scrum* deben tener idealmente de seis a diez miembros. Esta práctica resulta en la idea errónea de que el marco de Scrum sólo se puede utilizar para proyectos pequeños. Sin embargo, se puede escalar fácilmente para el uso eficaz en grandes *projects*. En situaciones donde el tamaño del *Equipo Scrum* excede diez personas, múltiples *Equipos Scrum* se pueden formar para trabajar en el *project*. El proceso *Scrum of Scrum* facilita la coordinación entre los *Equipos Scrum* lo que permite una aplicación eficaz en los proyectos grandes.

Los proyectos grandes o complejos a menudo se implementan como parte de un *program* o *portfolio*. El marco de Scrum también se puede aplicar para gestionar *programs* y *portfolios*. El enfoque lógico de las directrices y los principios de este marco pueden utilizarse para gestionar proyectos de cualquier tamaño, que abarcan grandes geografías y organizaciones. Los *projects* grandes pueden tener múltiples *Equipos Scrum* trabajando de forma paralela por lo que es necesario sincronizarse y facilitar el flujo de información y mejorar la comunicación. El proceso *Convene Scrum of Scrum* asegura esta sincronización. Los diversos *Equipos Scrum* están representados en esta reunión y los objetivos son proporcionar actualizaciones sobre el progreso, discutir los retos que enfrentan durante el proyecto, y coordinar las actividades. No hay reglas fijas en cuanto a la frecuencia de estas reuniones. Los factores que determinan la frecuencia son la cantidad de dependencia entre los equipos, el tamaño del *project*, el nivel de complejidad y las recomendaciones del *Cuerpo de asesoramiento de Scrum*.

1.3 Propósito de la Guía SBOK™

En los últimos años, se ha hecho evidente que las organizaciones que utilizan Scrum como el marco de ejecución de *projects* preferido, consistentemente obtienen altos rendimientos de inversión. El enfoque de Scrum en la entrega impulsada por el valor ayuda a que los *Equipos Scrum* ofrezcan resultados durante el proyecto tan pronto como les sea posible.

La *Guía SBOK™* ha sido desarrollada como un medio para crear una guía necesaria para las organizaciones y profesionales de la gestión de *projects* que deseen implementar Scrum, así como para los que ya lo están haciendo y desean mejorar sus procesos. Se basa en la experiencia adquirida de miles de *projects* a través de una variedad de organizaciones e industrias. Las contribuciones de muchos expertos en Scrum y profesionales de gestión de *projects* se han considerado en su desarrollo.

La *Guía SBOK™* es especialmente valiosa:

- para los miembros principales del equipo Scrum incluyendo los siguientes:
 - *Propietario del producto* que deseen comprender plenamente el marco de Scrum y particularmente al *customer* o *stakeholder* relacionado con las preocupaciones que involucran *Justificación del negocio*, la calidad, el cambio, y los aspectos de riesgo asociados con los *projects* Scrum.
 - *Scrum Masters* que quieran aprender sobre su papel en la supervisión de la aplicación del marco de Scrum en *projects* Scrum.
 - Miembros del *Equipo Scrum* que deseen comprender mejor los procesos de Scrum y las herramientas asociadas que pueden ser utilizadas para crear el *product* o servicio del *project*.
- como una guía completa para todos los profesionales que ya trabajan en *projects* Scrum en una organización o industria.
- como fuente de referencia para cualquier persona que interactúe con el equipo central de Scrum, incluyendo pero no limitado al *Portfolio Propietario del producto*, *Scrum Master*, *Portfolio Scrum Master*, *Program Propietario del producto*, *Program Scrum Master*, *Cuerpo de asesoramiento de Scrum*, y *Stakeholder* (tal como un patrocinador, *customer* y usuarios).
- como un manual para cualquier persona que no tenga experiencia previa o conocimiento del marco de Scrum, pero quiera aprender más sobre el tema.

El contenido de la *Guía SBOK™* también es útil para las personas que se preparan para tomar los siguientes exámenes de certificación SCRUMstudy™:

- *Scrum Developer Certified (SDC™)*
- *Scrum Master Certified (SMC™)*
- *Agile Expert Certified (AEC™)*
- *Scrum Propietario del producto Certified (SPOC™)*
- *Expert Scrum Master (ESM™)*

1.4 Marco de la Guía SBOK™

La Guía SBOK™ se divide en las siguientes tres áreas:

1. **Los principios** (*principles*) contemplados en el capítulo 2 amplian sobre los seis principios que constituyen la fundación sobre la que se basa Scrum.
2. **Los aspectos** (*aspects*) cubiertos en los capítulos 3 al 7 describen los cinco aspectos que son considerados importantes para todos los proyectos Scrum.
3. **Los procesos** (*processes*) cubiertos en los capítulos 8 al 12 incluyen los diecinueve procesos de Scrum y sus entradas (*inputs*), herramientas (*tools*) y salidas (*outputs*) asociadas.

La figura 1-2 ilustra el marco de la Guía SBOK™, lo que demuestra que los principios, aspectos y procesos interactúan entre sí y son de igual importancia al tratar de obtener una mejor comprensión del marco de Scrum.

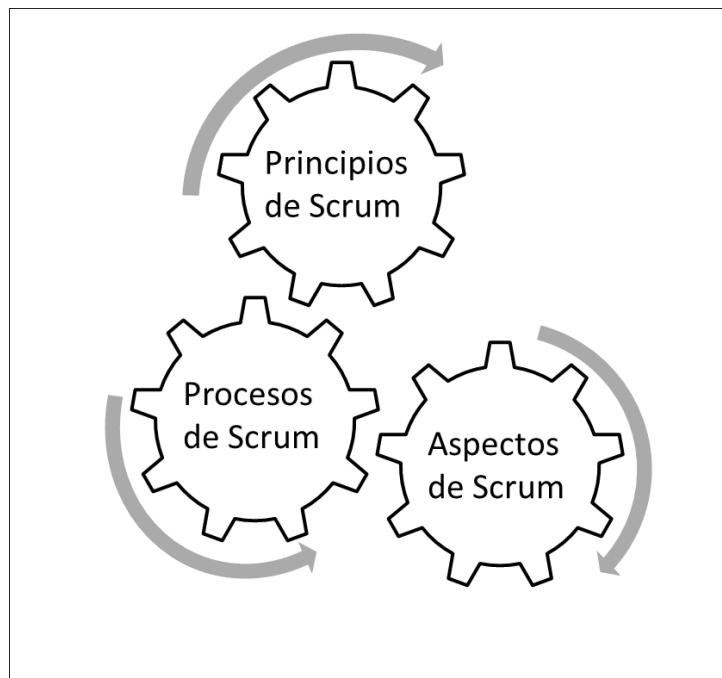


Figura 1-2: SBOK™ Guía del Marco

1.4.1 ¿Cómo usar la *Guía SBOK™*?

La *Guía SBOK™* puede ser usada como referencia y guía de conocimiento tanto por practicantes de Scrum con experiencia y otros profesionales de desarrollo de *products* y de servicios, así como por personas que no tienen experiencia previa o conocimiento de Scrum o metodologías de gestión de *projects*. Los contenidos se organizan para facilitar la consulta de los tres roles centrales del equipo Scrum: *Scrum Master*, *Propietario del producto* y *Equipo Scrum*.

Los capítulos que abarcan los seis principios de Scrum (capítulo 2) y cinco aspectos de Scrum (capítulo 3 al 7) incluyen una *Guía de roles*. Esta guía ofrece orientación sobre los roles del Equipo Principal/Central de Scrum.

Con el fin de facilitar la mejor aplicación del marco de Scrum, la *Guía SBOK™* ha diferenciado claramente entre las entradas, las herramientas y las salidas obligatorias de las opcionales. Las entradas, herramientas y salidas indicadas por asteriscos (*) son obligatorias mientras que las otras sin asteriscos son opcionales. Se recomienda que las personas que recién aprenden sobre Scrum, se centren principalmente en las entradas, las herramientas y las salidas obligatorias, mientras que los profesionales con más experiencia deben leer todos los capítulos del proceso.

1.4.2 Scrum Principles

Los principios de Scrum son las pautas básicas para aplicar el marco de Scrum y obligatoriamente deben usarse en todos los *projects* Scrum. Los seis principios de Scrum presentados en el capítulo 2 son:

1. *Control del proceso empírico*
2. *Auto-organización*
3. *Colaboración*
4. *Value-based Prioritization*
5. *Tiempo asignado*
6. *Desarrollo iterativo*

La figura 1-3 ilustra los seis principios de Scrum.

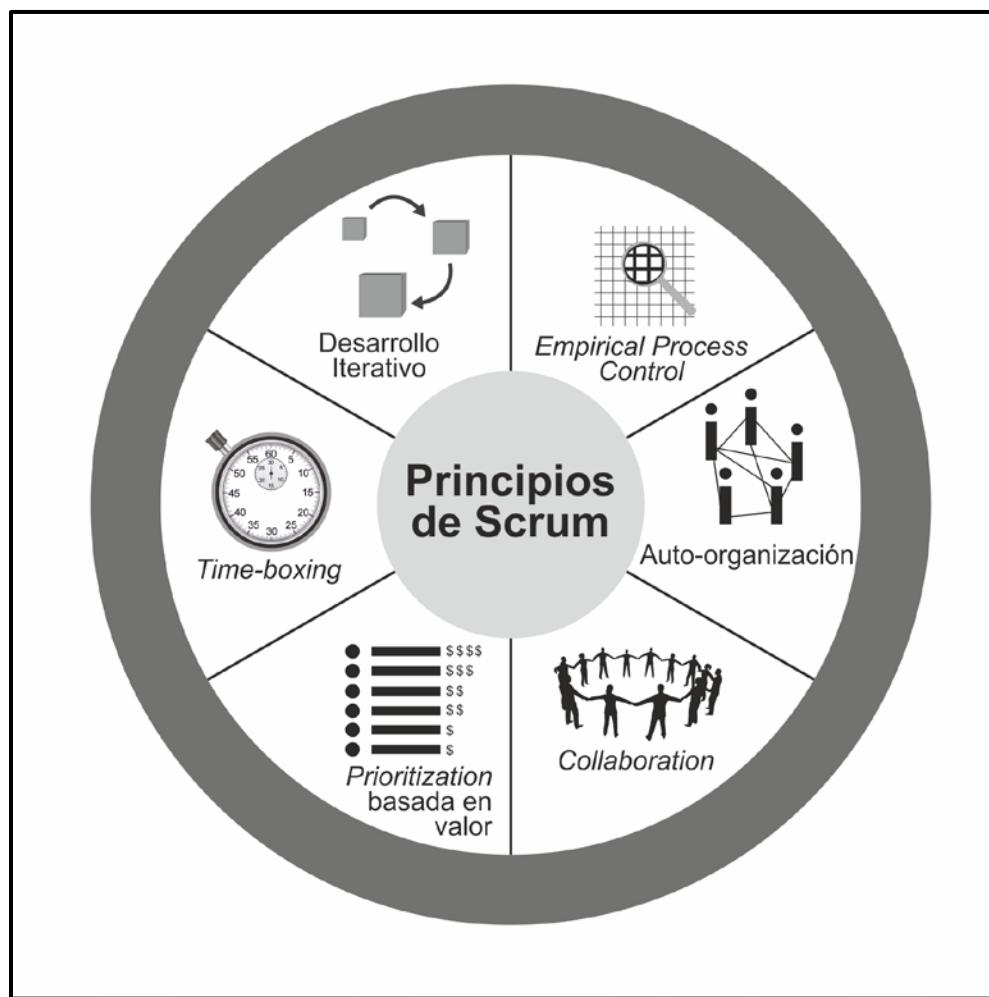


Figura 1-3: Scrum Principles (Los principios de Scrum)

Los principios de Scrum se pueden aplicar a cualquier tipo de *project* en cualquier organización y se deben mantener con el fin de garantizar la aplicación efectiva del marco de Scrum. Los principios Scrum no son negociables y deben aplicarse como se especifica en la *Guía SBOK™*. El mantener los principios intactos y usarlos apropiadamente infunde confianza en el marco de Scrum con respecto a la consecución de los objetivos del proyecto. Los aspectos y procesos de Scrum, sin embargo, pueden ser modificados para cumplir con los requisitos del *project* o la organización.

1. ***Control del proceso empírico***—Este principio pone de relieve la filosofía central de Scrum en base a las tres ideas principales de *transparencia, inspection y adaptation*.
2. ***Auto-organización***—Este principio se centra en los trabajadores de hoy, que entregan un valor significativamente mayor cuando son auto-organizados lo cual resulta en equipos con un gran sentimiento de compromiso y responsabilidad; a su vez, esto produce un entorno innovador y creativo que es más propicio para el crecimiento.
3. ***Colaboración***—Este principio se centra en las tres dimensiones básicas relacionadas con el trabajo colaborativo: conciencia, articulación y apropiación. También aboga por la gestión de proyectos como un proceso de creación de valor compartido con los equipos de trabajo e interactuar conjuntamente para ofrecer el mayor valor.
4. ***Prioritization basado en valor***—Este principio pone de relieve el enfoque de Scrum para ofrecer el máximo valor de negocio, desde el principio del *project* hasta su conclusión.
5. ***Tiempo asignado***—Este principio describe cómo el tiempo se considera una restricción limitante en Scrum, y cómo se utiliza para ayudar a manejar eficazmente la planificación y ejecución del *project*. Los elementos de *time-box* en Scrum son *Sprints, Daily Standup Meetings, Sprint Planning Meetings, y Sprint Review Meetings*.
6. ***Desarrollo Iterativo***—Este principio define el desarrollo iterativo y enfatiza cómo manejar mejor los cambios y crear productos que satisfagan las necesidades del *customer*. También delinea las responsabilidades del *Propietario del producto* y las de la organización relacionadas con el desarrollo iterativo.

1.4.3 Aspectos de Scrum

Los aspectos de Scrum se deben abordar y gestionar a lo largo de un *project* Scrum. Los cinco aspectos de Scrum presentados en los capítulos 3 al 7 son:

1.4.3.1 Organización (Organization)

Entender los roles y responsabilidades definidos en un *project* Scrum es muy importante para asegurar la exitosa implementación de Scrum.

Los roles de Scrum se dividen en dos grandes categorías:

1. **Core Roles**—*Core Roles* son aquellos papeles que obligatoriamente se requieren para producir el producto o servicio del proyecto. Las personas a quienes se les asignan *Core Roles* están plenamente comprometidas con el proyecto y son las responsables del éxito de cada iteración del *project* y del *project* en su totalidad.

Estas funciones incluyen:

- El *Propietario del producto* es la persona responsable de lograr el máximo valor empresarial para el proyecto. Él/ella también es responsable de la articulación de requisitos del *customer* y de mantener el *Justificación del negocio* para el *project*. El *Propietario del producto* representa *la voz del cliente* (*voice of the customer-VOC*).
- El *Scrum Master* es un facilitador que asegura que el *Equipo Scrum* esté dotado de un ambiente propicio para completar el *project* con éxito. El *Scrum Master* guía, facilita y les enseña las prácticas de Scrum a todos los involucrados en el proyecto; elimina los *impediments* que encuentra el equipo; y asegura que se estén siguiendo los procesos de Scrum.
- El *Equipo Scrum* es el grupo o equipo de personas responsables de la comprensión de los requisitos especificados por el *Propietario del producto* y de la creación de los Entregables (*Deliverables*) del *project*.

2. ***Non-core Roles***—*Non-core Roles* son los papeles que no son obligatoriamente necesarios para el proyecto Scrum y pueden incluir miembros de los equipos que están interesados en el *project*. No tienen ningún papel formal en el equipo del *project* y pueden interactuar con el equipo, pero no pueden ser responsables del éxito del *project*. Las *Non-core Roles* deben tenerse en cuenta en cualquier proyecto de Scrum.

Non-core roles incluyen los siguientes:

- **Socio(s)**, que es un término colectivo que incluye a los *customers*, usuarios y patrocinadores, que con frecuencia interactúan con el Equipo Principal de Scrum (Scrum Core Team), e influyen en el *project* a lo largo de su desarrollo. Lo más importante es que el *project* produzca beneficios de colaboración para los *socios*.
- **Cuerpo de asesoramiento de Scrum** es una función opcional, que generalmente consiste en un conjunto de documentos y/o un grupo de expertos que normalmente están involucrados en la definición de objetivos relacionados con la calidad, las regulaciones gubernamentales, la seguridad y otros parámetros claves de la organización. El SGB guía el trabajo llevado a cabo por el Propietario del producto, Scrum Master y Equipo Scrum.
- **Los vendedores (vendors)**, incluyendo a individuos u organizaciones externas, ofrecen productos y/o servicios que no están dentro de las competencias básicas de la organización del *project*.
- **Jefe Propietario del producto** es un papel en los proyectos más grandes con *Equipos Scrums* múltiples. Esta función se encarga de facilitar el trabajo de los *Propietario del producto* y del mantenimiento de *Justificación del negocio* para el *project* más grande.
- **El Jefe Scrum Master** es el responsable de coordinar las actividades relacionadas con Scrum en grandes *projects*, las cuales pueden requerir que varios *Equipos Scrum* trabajen en paralelo.

La figura 1-4 ilustra la estructura de la organización Scrum.

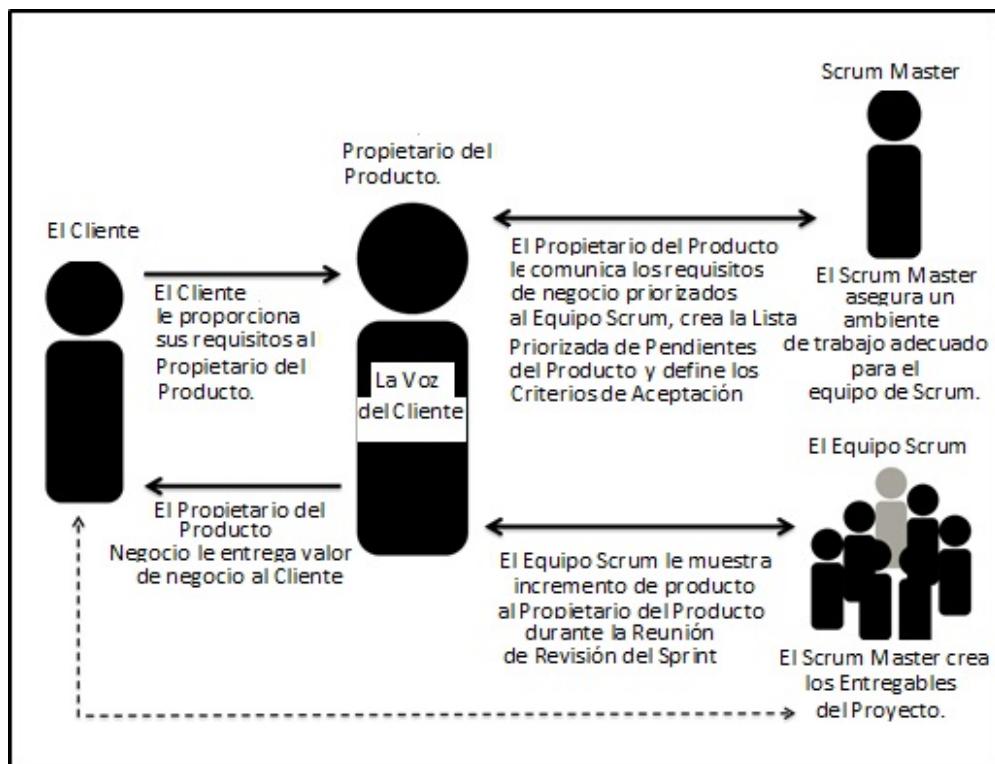


Figura 1-4: Organización en Scrum

El aspecto Organización de Scrum también se ocupa de los requisitos de estructura del equipo para implementar Scrum en *programs* y *portfolios*.

1.4.3.2 Justificación del negocio (*Business Justification*)

Es importante para una organización llevar a cabo una evaluación apropiada del negocio antes de comenzar un proyecto. Esto ayuda a quienes toman decisiones a entender la necesidad de cambio en la empresa, o de un nuevo *product* o servicio, al igual que a comprender la justificación para seguir adelante con un *project* y su viabilidad.

Justificación del negocio en Scrum se basa en el concepto de entrega impulsada por el valor. Una de las características claves de cualquier proyecto es la incertidumbre de los resultados. Es imposible garantizar el éxito del proyecto, independientemente del tamaño o la complejidad de un *project*. Teniendo en cuenta esta incertidumbre de alcanzar el éxito, Scrum intenta iniciar la entrega de los resultados lo antes posible en el *project*. Esta entrega temprana de los resultados, y por lo tanto el valor, proporciona una oportunidad para la reinversión y les demuestra el valor del *project* a los socios.

La adaptabilidad de Scrum permite que los objetivos y procesos del *project* cambien si cambia su *Justificación del negocio*. Es importante señalar que si bien el *Propietario del producto* es el responsable principal del *Justificación del negocio*, otros miembros del equipo también contribuyen considerablemente.

1.4.3.3 Calidad (*Quality*)

En Scrum, la calidad se define como la capacidad del *product* o *products* entregables de cumplir con los *Acceptance Criteria* y de alcanzar el valor de negocio que espera el *customer*.

Para asegurar que un proyecto cumpla con los requisitos de calidad, Scrum adopta un enfoque de *Continuous Improvement* del cual el equipo aprende de sus experiencias y del compromiso de los *socios*, y así actualiza constantemente al *Prioritized Product Backlog* con cualquier cambio de requisito. El *Prioritized Product Backlog* sólo se completa al cierre o a la terminación del *project*. Cualquier cambio en los requisitos debe reflejar los cambios en el entorno empresarial ya sea interno o externo, y el equipo continuamente se debe adaptar a alcanzar esos requisitos.

Dado a que Scrum requiere que el trabajo se realice en incrementos durante los *Sprints*, esto hace que los errores o defectos sean notados con más facilidad a través de pruebas de calidad (*quality*) repetitivas, y no simplemente cuando el *product* final o servicio esté casi terminado. Por otra parte, las tareas relacionadas con la calidad (por ejemplo, desarrollo, prueba y documentación) se completan por el mismo equipo como parte del mismo *Sprint*. Esto asegura que la calidad sea inherente a cualquier entregable creado como parte de un *Sprint*. Dichas entregas de *projects* Scrum, que son potencialmente entregables, se les conoce como "Hecho" (*Done*).

Por lo tanto, las pruebas repetitivas de *Continuous Improvement* optimizan la probabilidad de alcanzar los niveles esperados de calidad en un *project* Scrum. Las discusiones constantes entre el equipo principal de Scrum y los *socios* (incluyendo los clientes y los usuarios), junto con incrementos reales del *product* que se entregan al final de cada *Sprint*, aseguran que la diferencia entre las expectativas de los *customers* del proyecto y los verdaderos entregables se reduzca constantemente.

El *Cuerpo de asesoramiento de Scrum* también puede proporcionar directrices sobre la calidad que pueden ser de interés para todos los *projects* de Scrum en la organización.

1.4.3.4 Cambio (*Change*)

Cada proyecto, independientemente de su método o marco utilizado, se expone a cambios. Es imperativo que los miembros del equipo del *project* entiendan que los procesos de desarrollo de Scrum están diseñados para aceptar el cambio. Las organizaciones deben tratar de maximizar los beneficios que se derivan de los cambios y minimizar los impactos negativos a través de los procesos de gestión de cambio diligente, según los principios de Scrum.

Un principio fundamental de Scrum es su reconocimiento de que a) los *socios* (por ejemplo, los *customers*, los usuarios y patrocinadores) cambian de opinión acerca de lo que quieren y necesitan durante el curso del proyecto (a veces referido como "requisitos churn") y b) que es muy difícil, si no imposible, que los *socios* definan todos los requisitos durante el inicio del *project*.

Los proyectos Scrum le dan la bienvenida a los cambios mediante el uso de los *Sprints* cortos y repetitivos que incorporan la retroalimentación del *customer* en cada entrega del *Sprint*. Esto permite que el *customer* interactue regularmente con los miembros del *Equipo Scrum*, vea entregables a medida que estén listos, y cambie los requisitos si es necesario antes del siguiente *Sprint*.

Asimismo, los equipos de gestión de *programs* o de *portfolio* pueden responder a los *Change Requests* pertenecientes a los *projects* Scrum aplicables a su nivel.

1.4.3.5 Riesgo (*Risk*)

El riesgo se define como un evento incierto o conjunto de eventos que pueden afectar a los objetivos de un *project* y pueden contribuir a su éxito o fracaso. Los *risks* que pueden tener un impacto positivo en el *project* se les conoce como *Opportunities*, mientras que las amenazas son *risks* que podrían afectar al proyecto de una manera negativa. La gestión del riesgo debe hacerse de forma preventiva, y es un proceso iterativo que debe comenzar al inicio del *project* y continuar a lo largo del ciclo de vida del *projecto*. El proceso de gestión de *risks* debe seguir algunos pasos estandarizados para asegurar que los *risks* sean identificados, evaluados y un curso de acción esté determinado y que se actúe en consecuencia.

Los *risks* deben ser identificados, evaluados, y se les debe responder basado en dos factores: la probabilidad de ocurrencia de cada riesgo y el posible impacto en el caso de tal ocurrencia. Los *risks* con una alta probabilidad y valor de impacto (que se calculará multiplicando los dos factores), deberán ser atendidos antes que los que tengan un valor relativamente bajo. En general, una vez que se detecte un riesgo, es importante entender el riesgo en relación con las causas probables y los posibles efectos.

1.4.4 Procesos de Scrum

Los procesos de Scrum abordan las actividades y el flujo específico de un proyecto Scrum. En total hay diecinueve procesos que se agrupan en cinco fases. Estas fases se presentan en los capítulos 8 al 12 de la *Guía SBOK™*, como se muestra en la Tabla 1-1.

Capítulo	Fase	Procesos
8	Iniciar (<i>Initiate</i>)	<ol style="list-style-type: none"> 1. <i>Crear la visión del proyecto (Create Project Vision)</i> 2. <i>Identificar al Scrum Master y al socio(s) (Identify Scrum Master and Stakeholder(s))</i> 3. <i>Formación de un equipo Scrum (Form Equipo Scrum)</i> 4. <i>Desarrollo de épica(s) (Develop Epic(s))</i> 5. <i>Creación de la lista priorizada de pendientes del producto (Create Prioritized Product Backlog)</i> 6. <i>Realizar el plan de lanzamiento (Conduct Release Planning)</i>
9	Planear y Estimar (<i>Plan and Estimate</i>)	<ol style="list-style-type: none"> 7. <i>Elaborar historias de usuario (Create User Stories)</i> 8. <i>Aprobar, estimar y asignar historias de usuarios (Approve, Estimate, and Commit User Stories)</i> 9. <i>Elaboración de tareas (Create Tasks)</i> 10. <i>Estimar tareas (Estimate Tasks)</i> 11. <i>Elaboración de la lista de pendientes del Sprint (Create Sprint Backlog)</i>
10	Implementar (<i>Implement</i>)	<ol style="list-style-type: none"> 12. <i>Crear entregables (Create Deliverables)</i> 13. <i>Llevar a cabo el Standup diario (Conduct Daily Standup)</i> 14. <i>Mantenimiento de la lista priorizada de pendientes del producto (Groom Prioritized Product Backlog)</i>
11	Revisión y Retrospectiva (<i>Review and Retrospect</i>)	<ol style="list-style-type: none"> 15. <i>Convocar Scrum de Scrums (Convene Scrum of Scrums)</i> 16. <i>Demostración y validación del Sprint (Demonstrate and Validate Sprint)</i> 17. <i>Retrospectiva de Sprint (Retrospect Sprint)</i>
12	Lanzamiento (<i>Release</i>)	<ol style="list-style-type: none"> 18. <i>Envío de entregables (Ship Deliverables)</i> 19. <i>Retrospectiva del proyecto (Retrospect Project)</i>

Tabla 1-1: Resumen de los procesos de Scrum

Las fases describen cada proceso en detalle, incluyendo sus entradas, herramientas y salidas asociadas. En cada proceso, algunas entradas, herramientas y salidas son obligatorias (las que tienen un asterisco [*] después de sus nombres), mientras que otras son opcionales. La inclusión de las entradas, herramientas y/o salidas opcionales dependerá del proyecto en particular, organización o industria. Las entradas, herramientas y salidas indicadas como obligatorias son importantes para la implementación exitosa de Scrum en cualquier organización.

1.4.4.1 *Initiate (Iniciar)*

1. *Crear la visión del proyecto*—En este proceso, el *Project Vision Case* del negocio es revisado para crear una *Declaración de la Visión del Proyecto* que servirá de inspiración y proporcionará un enfoque de todo el proyecto. El *Propietario del producto* se identifica en este proceso.
2. *Identificar al Scrum Master y al socio(s)*—En este proceso, el *Scrum Master* y el *Stakeholder* se identifican utilizando criterios de selección específicos.
3. *Formación de un equipo Scrum*—En este proceso, se identifican a los miembros del *Equipo Scrum*. Normalmente, el *Propietario del producto* es el responsable principal de la selección de los miembros del equipo, pero a menudo lo hace en *Colaboración* con el *Scrum Master*.
4. *Desarrollo de épica(s)*—En este proceso, la *Declaración de la Visión del Proyecto* sirve como la base para el desarrollo de *Epics*. *User Group Meetings* se pueden llevar a cabo para discutir *Epics* que sean apropiados.
5. *Creación de la lista priorizada de pendientes del producto*—En este proceso, *Epic(s)* son refinados, elaborados, y luego priorizados para crear un *Prioritized Product Backlog*. Los *Done Criteria* también se establecen en este punto.
6. *Realizar el plan de lanzamiento*—En este proceso, el Equipo Principal/Central de Scrum revisa los *User Stories* en el *Prioritized Product Backlog* para desarrollar un *Release Planning Schedule*, que es esencialmente un *program* (programa) de implementación por fases que se puede compartir con los *socios del proyecto*. El *Length of Sprint* también se determina en este proceso.

1.4.4.2 *Plan and Estimate (Planear y Estimar)*

7. *Elaborar historias de usuario*—En este proceso, se crean los *User Stories* y los *User Story Acceptance Criteria*. Los *User Stories* son generalmente escritos por el *Propietario del producto* y están diseñados para asegurar que los requisitos del *customer* estén claramente representados y puedan ser plenamente comprendidos por todos los *Socios*.

Los ejercicios de escritura de user stories se podrán llevar a cabo lo cual involucra a los miembros *Equipo Scrum*, resultando en la creación de user stories. Los *User Stories* se incorporan en el *Prioritized Product Backlog*.

8. *Aprobar, estimar y asignar historias de usuarios*—En este proceso, el Propietario del producto aprueba los *User Stories* para un *Sprint*. Luego, el *Scrum Master* y el *Equipo Scrum* estiman el esfuerzo necesario para desarrollar la funcionalidad descrita en cada historia de usuario, y el *Equipo Scrum* se compromete a entregar los requisitos del *customer* en forma de *Approved, Estimated, and Committed User Stories*.
9. *Elaboración de tareas*—En este proceso, los *Approved, Estimated, and Committed User Stories* se dividen en tareas específicas y se compilan en un *Task List*. A menudo, un *Task Planning Meeting* se convocará al efecto.
10. *Estimar tareas*—En este proceso, el Equipo Principal de Scrum durante *Task Estimation Meetings* (Reuniones de Estimacion del Labor) estima el esfuerzo necesario para realizar cada tarea del *Task List*. El resultado de este proceso es un *Effort Estimated Task List*.
11. *Elaboración de la lista de pendientes del Sprint*—En este proceso, el Equipo Principal de Scrum lleva a cabo un *Sprint Planning Meeting* donde el grupo crea un *Sprint Backlog* que contiene todas las tareas que deben completarse en el *Sprint*.

1.4.4.3 *Implement (Implementar)*

12. *Crear entregables*—En este proceso, el *Equipo Scrum* trabaja en las tareas del *Sprint Backlog* para crear *Sprint Deliverables*. Se utiliza a menudo un *Scrumboard* para realizar el seguimiento del trabajo y de actividades que se llevan a cabo. Las cuestiones o problemas que enfrenta el *Equipo Scrum* podrían ser actualizadas en un *Impediment Log*.
13. *Llevar a cabo el Standup diario*—En este proceso, todos los días se lleva a cabo una reunión que es *Time-box* altamente concentrada que se refiere como *Daily Standup Meeting*. Es aquí donde los miembros del *Equipo Scrum* se actualizan el uno al otro referente a sus progresos y sobre los *Impediments* que puedan enfrentar.
14. *Mantenimiento de la lista priorizada de pendientes del producto*—En este proceso, el *Prioritized Product Backlog* se actualiza y se mantiene continuamente. Un *Prioritized Product Backlog Review Meeting* puede ser considerado, en el que se discute y se incorpora el *Prioritized Product Backlog* de forma apropiada.

1.4.4.4 Review and Retrospect (Revisión y retrospectiva)

15. *Convocar Scrum de Scrums*—En este proceso, Equipo Scrum Representatives (*representantes del Equipo Scrum*) convocan *Scrum of Scrums* (*SoS*) en intervalos predeterminados o cuando sea necesario para colaborar y realizar un seguimiento de su respectivo progreso, *impediments*, y de las dependencias entre los equipos. Esto es relevante sólo para grandes proyectos en los que múltiples *Equipos Scrum* están involucrados.
16. *Demostración y validación del Sprint*—En este proceso, el *Equipo Scrum* le demuestra el *Sprint Deliverable* al *Propietario del producto* y a los *Socios* relevantes en un *Sprint Review Meeting*. El propósito de esta reunión es asegurar la aprobación y aceptación del *Propietario del producto* de los Entregables creados en el *Sprint*.
17. *Retrospectiva de Sprint*—En este proceso, el *Scrum Master* y el *Equipo Scrum* se reúnen para discutir las lecciones aprendidas a lo largo del *Sprint*. Esta información se documenta como las lecciones aprendidas que pueden aplicarse a los futuros *Sprints*. A menudo, como resultado de esta discusión, puede haber *Agreed Actionable Improvements* o recomendaciones actualizadas por parte del *Cuerpo de asesoramiento de Scrum*.

1.4.4.5 Release (Lanzamiento)

18. *Envío de entregables*—En este proceso, los *Accepted Deliverables* se les entregan a los *Socios* relevantes. Un acuerdo formal llamdo *Working Deliverables Agreement* documenta la finalización con éxito del *Sprint*.
19. *Retrospectiva del proyecto*—En este proceso, que completa el proyecto, los socios y miembros principales del del Equipo Principal de Scrum se reúnen para hacer una retrospectiva del proyecto e identificar, documentar e internalizar las lecciones aprendidas. A menudo, estas lecciones llevan a la documentación de *Agreed Actionable Improvement*, que se aplicará en futuros proyectos.

1.5 Scrum vs gestión de proyectos tradicional

En la tabla 1-2 se resumen muchas de las diferencias entre los modelos tradicionales de gestión de *projects* y Scrum.

	Scrum	Gestión de Proyectos Tradicional
El énfasis está en	Personas	Procesos
Documentación	Sólo mínima según se requiera	Exhaustivo
Estilo de Procesos	Iterativo	Lineal
Planificación por Adelantada	Baja	Alta
<i>Prioritization</i> de los Requisitos	Según el valor del negocio y regularmente actualizada	Fijo en el plan de proyecto
Quality Assurance	Centrada en el <i>customer</i>	Centrada en el Proceso
Organización	Auto-organizada	Gestionada
El Estilo de Gestión	Descentralizado	Centralizado
Cambio	Las actualizaciones de <i>Prioritized Product Backlog</i>	Sistema formal de Gestión del Cambio
Liderazgo	<i>Collaborative, Servant Leadership</i>	Mando y control
La Medición del Rendimiento	El valor del negocio	Plan de la Conformidad
<i>Return on Investment (ROI)</i>	Al comienzo y a lo largo del proyecto	Al final del proyecto
Participación del <i>Customer</i>	Alta durante todo el proyecto	Varía en función del ciclo de vida del proyecto

Tabla 1-2: Scrum vs. Gestión de proyectos tradicional

2. PRINCIPIOS

2.1 Introducción

2

Los principios de Scrum son el fundamento sobre lo que se basa el marco de Scrum. Estos principios se pueden aplicar a cualquier tipo de *project* u organización, y deben ser respetados con el fin de garantizar la aplicación apropiada de Scrum. Los aspectos y procesos de Scrum pueden ser modificados para cumplir con los requerimientos del *project*, o la organización de usuario, pero los principios de Scrum no son negociables y deben aplicarse como se describe en el marco presentado en *Una guía para los conocimientos de Scrum (Guía SBOK™)*. El mantener los principios intactos y usarlos apropiadamente infunde confianza en el usuario del marco de Scrum con respecto a la consecución de los objetivos del *project*. Los principios se consideran los lineamientos básicos para la aplicación del marco de Scrum.

Los principios, tales como se definen en la *Guía SBOK™*, son aplicables a los siguientes:

- *Portfolios, programs, y/o projects* de cualquier sector
- *Products*, servicios, o cualquier otro resultado que se les entregará a los socios
- *Projects* de cualquier tamaño y complejidad

El término "producto" (*product*) en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria-desde proyectos pequeños o equipos con tan sólo seis miembros, hasta *projects* grandes y complejos con hasta varios cientos de miembros por equipo.

Este capítulo está dividido en las siguientes secciones:

2.2 Guía de los roles—Esta sección describe qué sección o subsección es más relevante para cada uno de los núcleos de los roles de Scrum tales como el *Propietario del producto*, *Scrum Master*, y *Equipo Scrum*.

2.3 Control del proceso empírico—Esta sección describe el primer principio de Scrum y las tres ideas principales: *Transparencia, inspection y adaptation*.

2.4 Auto-organización—Esta sección destaca el segundo principio de Scrum que se centra en los trabajadores de hoy, quienes entregan un valor significativamente mayor cuando se auto-organizan. Esta auto-organización se traduce en un mejor equipo de *buy-in* (creen en lo hacen) donde se sienten responsables; esto a su vez proporciona un entorno innovador y creativo que es más propicio para el crecimiento.

2.5 Colaboración—Esta sección hace hincapié en el tercer principio de Scrum donde el desarrollo de los *products* es un proceso de creación de valor compartido que necesita que todos los *Socios* trabajen e

interactúen en conjunto para ofrecer el mayor valor. También se centra en las dimensiones básicas de trabajo colaborativo: la sensibilización, articulación y apropiación.

2.6 Prioritization basada en el Valor—Esta sección presenta el cuarto principio de Scrum, que pone de relieve la unidad del marco de Scrum para entregar el máximo valor empresarial en un período de tiempo mínimo.

2.7 Tiempo asignado—Esta sección explica el quinto principio de Scrum que trata al tiempo como un factor limitante. También cubre el *Sprint*, *Daily Standup Meeting* y otros *Sprints* relacionados con las reuniones, tales como el *Daily Sprint Planning Meeting*, *Sprint Review Meeting*, los cuales están *Time-boxed*.

2.8 Desarrollo iterativo—En esta sección se aborda el sexto principio de Scrum que hace hincapié en que el desarrollo iterativo ayuda a gestionar mejores cambios y crear productos que satisfagan las necesidades del *customer*.

2.9 Scrum vs Gestión de Proyecto Tradicional—Esta sección destaca las principales diferencias entre los principios de Scrum y los principios de gestión de *projects* tradicional (Modelo de Cascada/Waterfall Model) y explica cómo Scrum funciona mejor en el mundo tan cambiante de hoy.

2.2 Guía de Roles

Todas las secciones de este capítulo son importantes para todos los papeles del Equipo Principal de Scrum - *Propietario del producto*, *Scrum Master*, y *Equipo Scrum*. Una comprensión clara de los principios de Scrum por parte de todos *Socios* es esencial para que el marco de Scrum sea un éxito en cualquier organización.

2.3 Control del proceso empírico

En Scrum, las decisiones se toman sobre la base de la observación y la experimentación, más que en la planificación inicial detallada. *Control del proceso empírico* se basa en las tres ideas principales de la transparencia, *Inspection* y *Adaptation*.

2.3.1 Transparencia

Transparencia permite que todas las facetas de cualquier proceso de Scrum sean observadas por cualquier persona. Esto promueve un flujo fácil y transparente de información en toda la organización y crea una cultura de trabajo abierta. En Scrum, la transparencia es representada a través de las siguientes acciones:

- Un *Declaración de la Visión del Proyecto* que puede ser visto por todos los *socios* y el *Equipo Scrum*

- Un proceso abierto de *Prioritized Product Backlog* con *User Stories* priorizados que se pueden ver por todos, tanto dentro como fuera del *Equipo Scrum*
- Un *Release Planning Schedule* que puede ser coordinado a través de múltiples *Equipos Scrum*
- Clara visibilidad sobre el progreso del equipo a través del uso de un *Scrumboard*, *Burndown Chart* y otros radiadores de información
- *Daily Standup Meetings* realizados durante el proceso de *Llevar a cabo el Standup diario* en el que todos los miembros del equipo informan lo que han hecho el día anterior, lo que van a hacer hoy, y cualquier problema que les impida completar sus tareas en el *Sprint* actual
- *Sprint Review Meetings* se llevan a cabo durante el proceso de *Sprint* llamado *Demonstrate and Validate*, en el que el *Equipo Scrum* les muestra los *Sprint Deliverables* potencialmente entregables a los *Propietario del producto* y a los *Socios*

La figura 2-1 resume el concepto de *Transparencia* en Scrum.

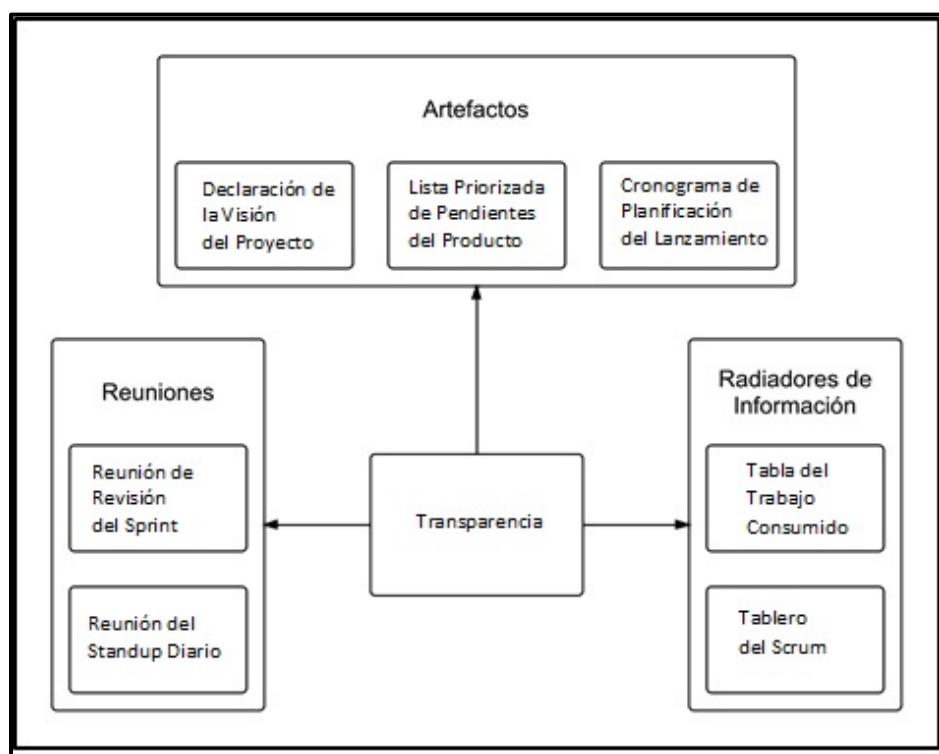


Figura 2-1: *Transparencia* en Scrum

2.3.2 *Inspection*

Inspection en Scrum es representado a través de las siguientes acciones:

- El uso de un *Scrumboard* común y otros radiadores de información que muestran el progreso del *Equipo Scrum* en completar las tareas del *Sprint* actual.
- La colección de retroalimentación del *customer* y otros *stakeholders* durante los procesos de *Desarrollo de épica(s)*, *Creación de la lista priorizada de pendientes del producto*, y *Realizar el plan de lanzamiento*.
- *Inspection* y la aprobación de los entregables por el *Propietario del producto* y el *customer* en el proceso de *Demostración y validación del Sprint*.

La figura 2-2 resume el concepto de *Inspection* en Scrum.

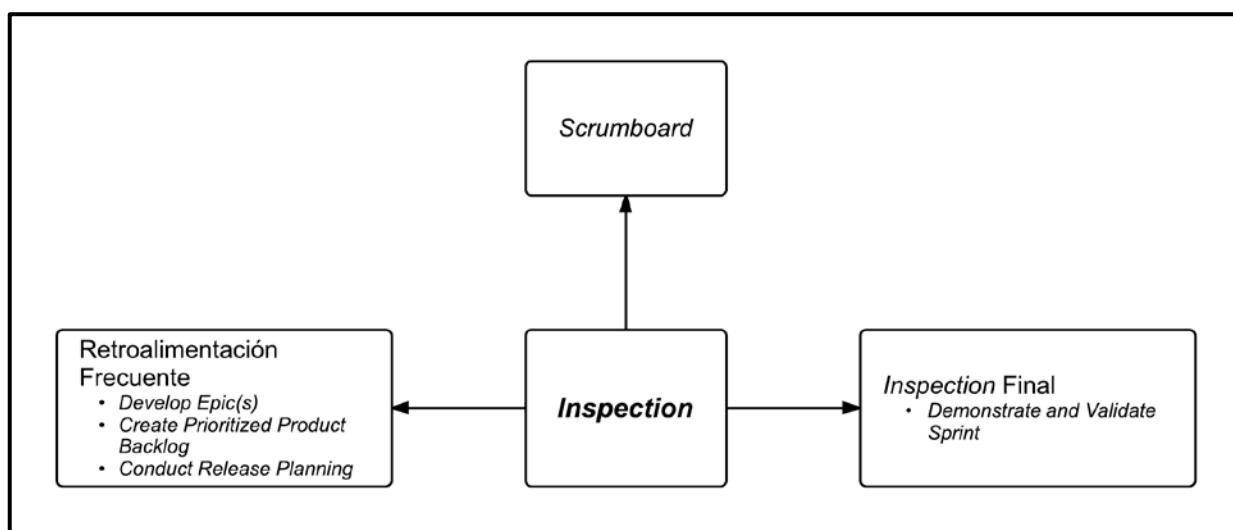


Figura 2-2: *Inspection* en Scrum

2.3.3 *Adaptation*

Adaptation sucede cuando el Equipo Principal de Scrum y los *socios* aprenden a través de la transparencia y la *inspection* y luego se adaptan al hacer mejoras en el trabajo ya en progreso. Algunos ejemplos de *adaptation* incluyen:

- En *Daily Standup Meetings*, los miembros del *Equipo Scrum* discuten abiertamente los *Impediments* para completar sus tareas y buscar la ayuda de otros miembros del equipo. Los miembros con más experiencia en el *Equipo Scrum* sirven como mentores a aquellos quienes tienen relativamente menos experiencia y conocimiento del proyecto o de tecnología.
- *Risk identification* se lleva a cabo y se reitera a lo largo del proyecto. Los *risks* identificados se convierten en entradas para varios procesos de Scrum incluyendo *Creación de la lista priorizada de*

pendientes del producto, Mantenimiento de la lista priorizada de pendientes del producto, y Demostración y validación del Sprint.

- Las mejoras pueden resultar en *change requests* que son discutidas y aprobadas durante los procesos de Desarrollo de épica(s), Creación de la lista priorizada de pendientes del producto, y Mantenimiento de la lista priorizada de pendientes del producto.
- El Cuerpo de asesoramiento de Scrum interactúa con los miembros del Equipo Scrum durante los procesos de User Stories, Estimar Tareas, Crear entregables y Mantenimiento de la lista priorizada de pendientes del producto para ofrecer orientación y también proporcionar conocimientos según sea necesario.
- En el proceso *Retrospectiva de Sprint* se determinan los *Agreed Actionable Improvements* en base a las salidas del proceso de Demostración y validación del Sprint.
- En *Retrospectiva del proyecto Meeting*, los participantes documentan las lecciones aprendidas y realizan revisiones en busca de *Opportunities* para mejorar los procesos y abordar las ineficiencias.

La figura 2-3 resume el concepto de *Adaptation* en Scrum.

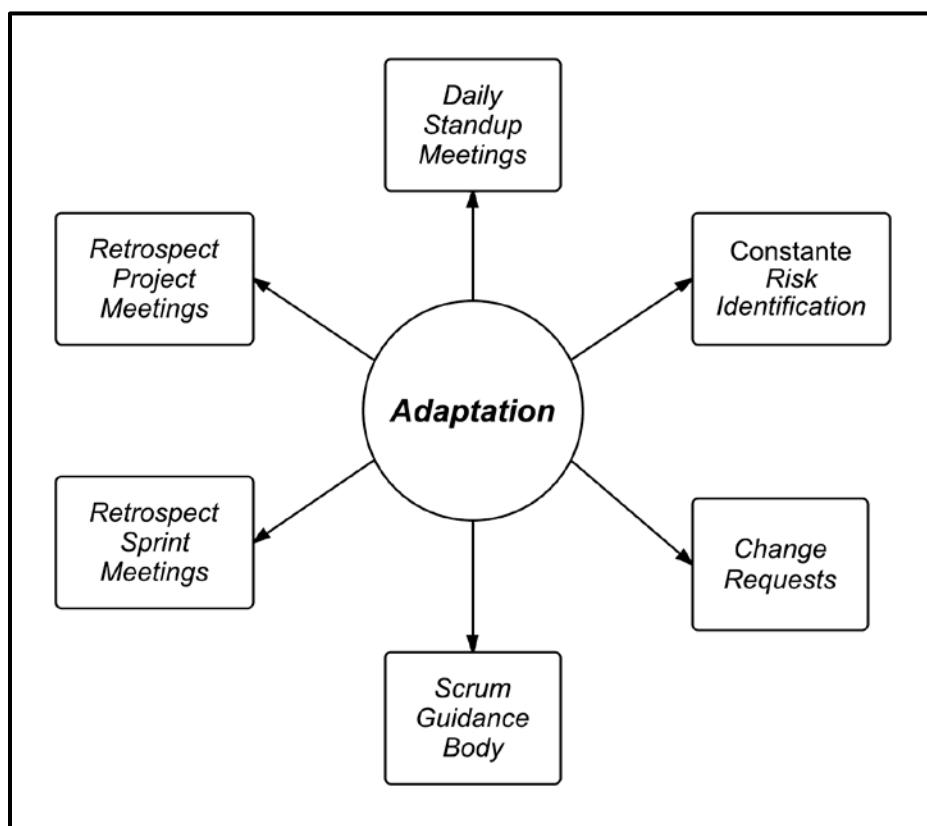


Figura 2-3: *Adaptation* en Scrum

Con otros métodos, como el modelo tradicional de Waterfall, se requiere de una planificación considerable que hay que hacer por adelantada y el *customer* generalmente no revisa los componentes del producto

hasta casi el final de una fase, o al final del *project*. Este método a menudo presenta enormes *risks* al éxito del proyecto, ya que tiene más potencial para impactar la ejecución de proyectos y la aceptación del *customer* de forma significativa. La interpretación y comprensión del *customer* sobre el producto final puede ser muy diferente de lo que realmente se entendió originalmente y fue producido por el equipo, algo que no se sabría hasta muy tarde en el desarrollo del proyecto.

La figura 2-4 muestra un ejemplo de estos desafíos.

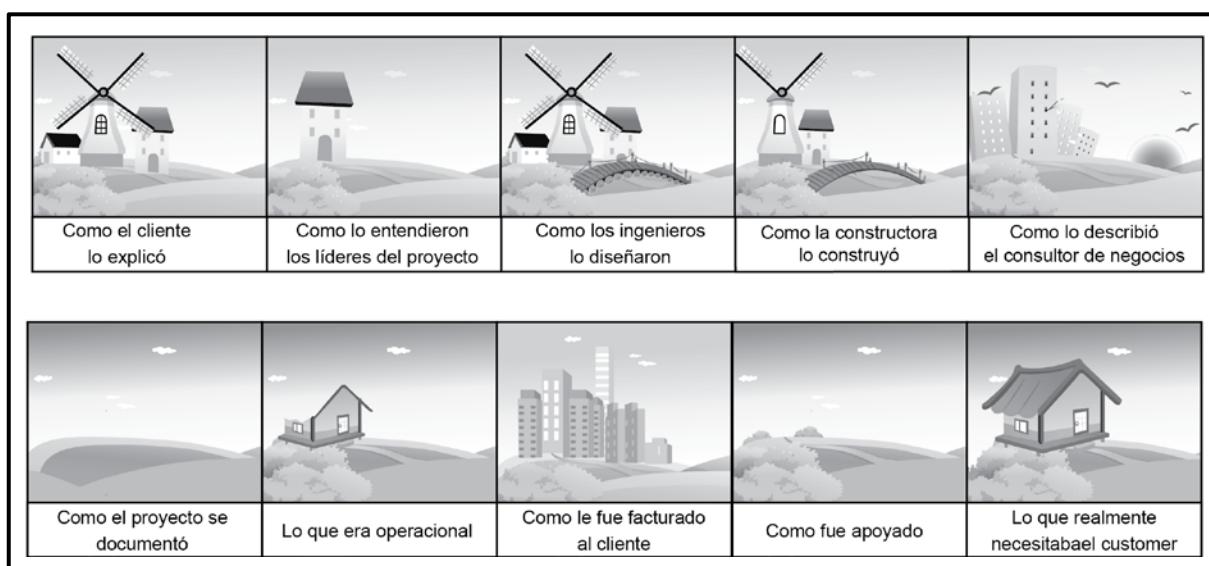


Figura 2-4: Retos en la Gestión de Proyectos Tradicional

2.4 Auto-organización

Scrum cree que los empleados son auto-motivados y buscan aceptar una mayor responsabilidad. Dado a eso, ofrecen mucho más valor cuando se organizan por cuenta propia.

El estilo de liderazgo preferido en Scrum es *Servant Leadership*, lo que hace hincapié en la obtención de resultados, centrándose en las necesidades del *Equipo Scrum*. Consulte la sección 3.10.3 para una discusión de liderazgo y de gestión de diferentes estilos.

2.4.1 Beneficios de Auto-organización

Auto-organización (auto-organización) como un principio esencial en Scrum conduce a lo siguiente:

- *Buy-in* del equipo (creer en lo que se hace) y responsabilidad y apropiación compartida
- Motivación, lo que conduce a un nivel de mejor rendimiento del equipo
- Entorno innovador y creativo que conduce al crecimiento

Auto-organización no significa que a los miembros del equipo se les permite actuar de la manera que deseen. Sólo significa que una vez que la visión del producto se define en el proceso *Crear la visión del proyecto*, el *Propietario del producto*, *Scrum Master* y *Equipo Scrum* son identificados. También, el equipo principal de Scrum trabaja muy de cerca con los *stakeholder* para perfeccionar los requisitos a medida que avanzan a través de los procesos de *Desarrollo de épica(s)* y *Elaborar historias de usuario*. *Team Expertise* se utiliza para evaluar las entradas necesarias para ejecutar la obra prevista del proyecto. Este juicio y la experiencia se aplican a todos los aspectos técnicos y de gestión del *project* durante el proceso de *Crear entregables*.

Aunque la *prioritization* se hace sobre todo por el *Propietario del producto*, quien representa la voz del *customer*, el *Equipo Scrum* auto-organizado está involucrado con la distribución de tareas y la estimación durante los procesos de *Elaboración de tareas* y *Estimar tareas*. Durante estos procesos, cada miembro del equipo es responsable de determinar qué tipo de trabajo él o ella va a hacer. Durante la ejecución de un *Sprint*, los miembros del equipo pueden necesitar alguna ayuda para completar sus tareas. Scrum se ocupa de esto a través de la interacción regular obligatoria en los *Daily Standup Meeting*. El *Equipo Scrum* sí interactúa con otros equipos a través de *Scrum of Scrums Meetings* y si es necesario puede buscar orientación adicional del *Cuerpo de asesoramiento de Scrum*.

Por último, el *Equipo Scrum* y el *Scrum Master* trabajan juntos para demostrar el incremento del *product* creado durante el *Sprint* en el proceso *Demostración y validación del Sprint*, donde los entregables hechos apropiadamente son aceptados. Dado a que los entregables son potencialmente enviables, (y el *Prioritized Product Backlog* se prioriza por los *User Stories* en el orden del valor creado por ellos), el *Propietario del producto* y el *customer* pueden visualizar y articular claramente el valor creado después de cada *Sprint*; y el *Equipo Scrum* a su vez tiene la satisfacción de ver su trabajo aceptado por el *customer* y los otros *Socios*.

Los principales objetivos de los equipos auto-organizados son los siguientes:

- Entender la visión del *project* y por qué el proyecto aporta valor a la organización
- *Estimate User Stories* durante el proceso de *Aprobar, estimar y asignar historias de usuarios* y asignar tareas a sí mismos durante el proceso de *Elaboración de la lista de pendientes del Sprint*
- Crear tareas de forma independiente durante el proceso de *Elaboración de tareas*
- Aplicar y aprovechar la experiencia de ser un equipo multi-funcional al trabajar en las tareas durante el proceso de *Crear entregables*
- Lograr resultados tangibles, que son aceptados por el *customer* y otros *socios* durante el proceso de *Demostración y validación del Sprint*
- Resolver problemas individuales al discutirlos durante los *Daily Standup Meetings*
- Aclarar cualquier discrepancia o duda y estar abierto a aprender cosas nuevas
- Actualizar los conocimientos y habilidades de manera continua a través de interacciones regulares dentro del equipo
- Mantener la estabilidad de los miembros del equipo durante toda la duración del proyecto al no cambiar los miembros, a menos que sea inevitable

La figura 2-5 ilustra los objetivos de un equipo auto-organizado.

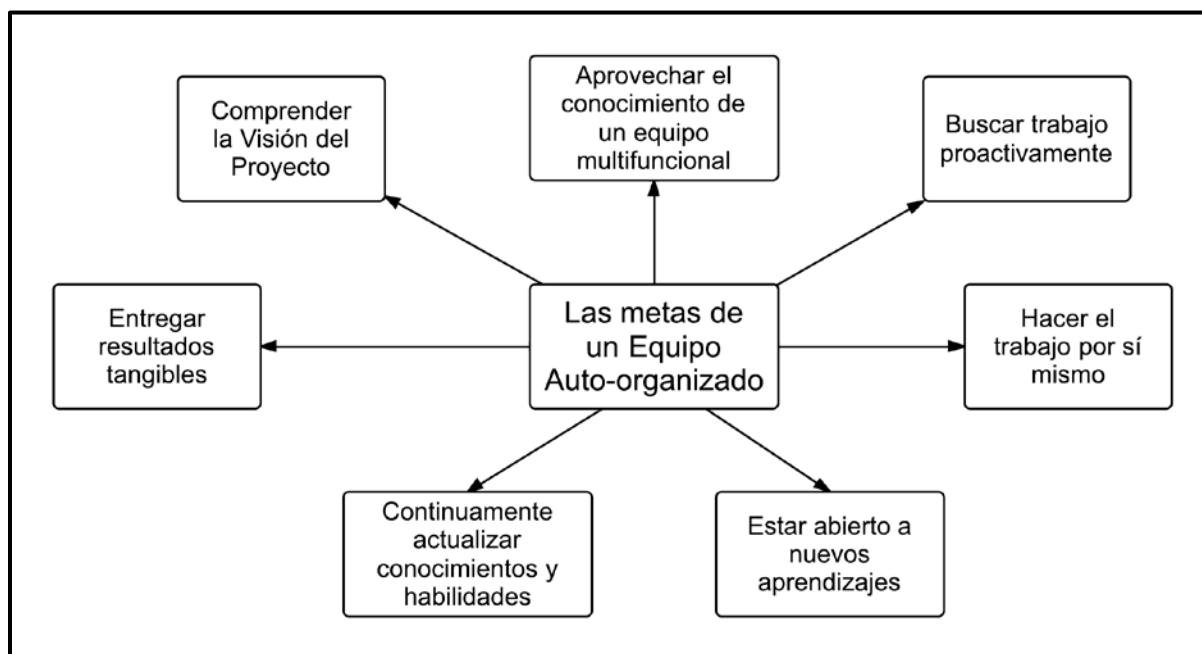


Figura 2-5: Objetivos de un equipo de auto-organización

2.5 Colaboración

Colaboración en Scrum se refiere a que el Equipo Principal de *Scrum* trabaja e interactua junto con los *Socios* para crear y validar los resultados del *project* así cumplir con los objetivos planteados en la Visión del Proyecto (Project Vision). Es importante tener en cuenta la diferencia entre la cooperación y *Colaboración*. La cooperación se produce cuando el producto del trabajo consiste en la suma de los esfuerzos de trabajo de varias personas en un equipo. *Colaboración* se produce cuando un equipo trabaja en conjunto para trabajar con los aportes del otro para producir algo más grande.

Las dimensiones básicas de trabajo en colaboración (*Colaboración*) son los siguientes:

- *Concientización* (*ser consciente del otro*)—Las personas que trabajan juntas deben estar al tanto del trabajo de los demás.
- *Articulación*— Los colaboradores deben dividir el trabajo en unidades, dividir las unidades entre los miembros del equipo, y luego, después que el trabajo esté hecho, reintegrarlo.
- *Apropiación*— La adaptación de tecnología a la propia situación; la tecnología se puede utilizar de una manera completamente diferente de lo esperado por los diseñadores.

2.5.1 Beneficios de *Colaboración* en Proyectos Scrum

The Agile Manifesto (Fowler y Highsmith, 2001) hace hincapié en *customer Colaboración* sobre la negociación del contrato. Por lo tanto, el marco de Scrum adopta un enfoque en el que los miembros del Equipo Principal de Scrum (*Propietario del producto*, *Scrum Master* y *Equipo Scrum*) colaboran entre sí y con los *Socios* para crear los entregables que proporcionan mayor valor posible para el *customer*. Esta *Colaboración* se produce durante todo el *project*.

Colaboración asegura que los siguientes *Project Benefits* se realicen:

1. La necesidad de cambios debido a requisitos poco clarificados se reduce al mínimo. Por ejemplo, durante los procesos de *Crear la visión del proyecto*, *Desarrollo de épica(s)*, y *Creación de la lista priorizada de pendientes del producto* el *Propietario del producto* colabora con los *socios* para crear la visión del proyecto, *Epic(s)* y *Prioritized Product Backlog* respectivamente. Esto asegurará que haya claridad entre los miembros principales del *Equipo Scrum* sobre el trabajo que se requiere para completar el *project*. El *Equipo Scrum* colabora continuamente con el *Propietario del producto* y los *socios* a través de un *Prioritized Product Backlog* transparente para crear los entregables del proyecto. Los procesos de *Llevar a cabo el Standup diario*, *Mantenimiento de la lista priorizada de pendientes del producto*, y *Retrospectiva de Sprint* dan margen a los miembros del equipo principal de Scrum para discutir lo que se ha hecho y colaborar en lo que hay que hacer. De esta manera se minimiza el número de *Change Requests* pedidos por el *customer*.
2. Los *Risks* se identifican y se tratan de manera eficiente. Por ejemplo, los *Risks* del proyecto se identifican y evalúan en los procesos de *Desarrollo de épica(s)*, *Crear entregables*, and *Llevar a cabo el Standup diario*.

cabo el *Standup diario* por parte de los miembros del equipo principal de Scrum. Las herramientas de reunión de revisión de Scrum como *Daily Standup Meeting*, *Sprint Planning Meeting*, *Prioritized Product Backlog Review Meeting*, proporcionan *opportunities* para el equipo, no sólo para identificar y evaluar los *risks*, sino también para implementar respuestas a los riesgos identificados como *high-priority risks*.

3. Se realiza el verdadero potencial del equipo. Por ejemplo, el proceso de *Llevar a cabo el Standup diario* le ofrece un margen al *Equipo Scrum* para colaborar y comprender las fortalezas y debilidades de sus miembros. Si un miembro del equipo se pasó del plazo de una tarea, los miembros del *Equipo Scrum* se alinean en colaboración para completar la tarea y cumplir con los objetivos acordados así llevar a cabo el *Sprint*.
4. Se garantiza *Continuous Improvement* a través de las lecciones aprendidas. Por ejemplo, el *Equipo Scrum* utiliza el proceso *Retrospectiva de Sprint* para identificar lo que no salió bien y lo que salió bien en el *Sprint* anterior. Esto proporciona una oportunidad para que el *Scrum Master* trabaje con el equipo y así estar más preparado para el próximo *Sprint*. Esto también asegurará que la *Colaboración* sea aún más eficaz en el próximo *Sprint*.

La figura 2-6 ilustra los beneficios de *Colaboración* en proyectos Scrum.

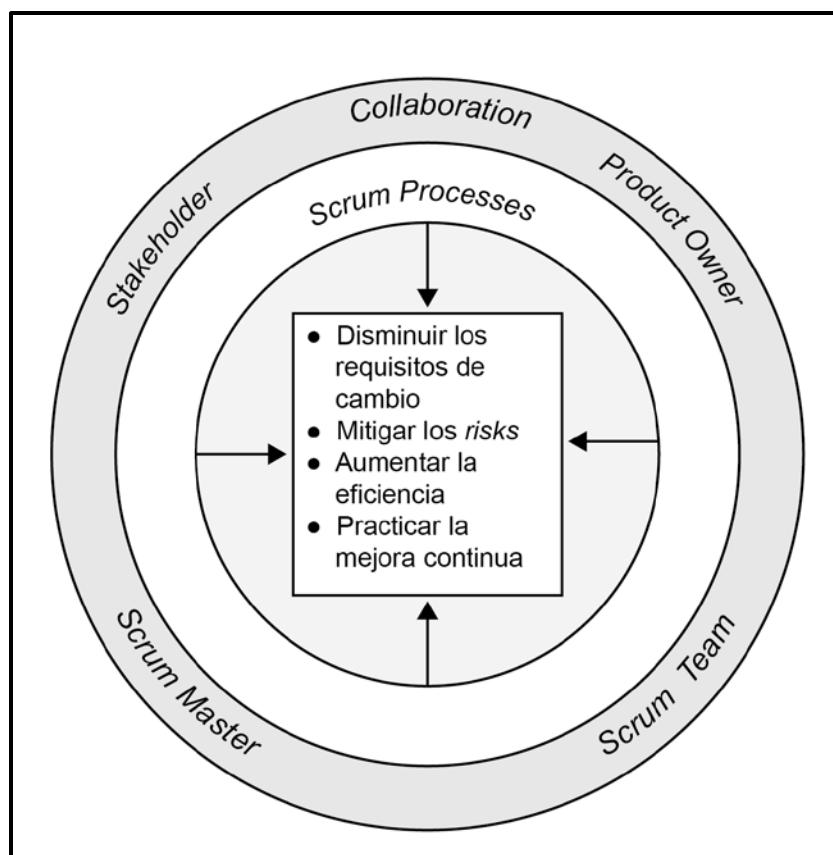


Figura 2-6: Beneficios de *Colaboración* en proyectos Scrum

2.5.2 La importancia de *Colocation* en *Colaboración*

Para muchas de las prácticas de Scrum, se requiere la comunicación de banda ancha. Por eso, se prefiere que los miembros del equipo sean colocados. *Colocation* permite la interacción formal e informal entre los miembros del equipo. Esto proporciona la ventaja de contar con los miembros del equipo siempre a mano para la coordinación, resolución de problemas y el aprendizaje. Algunos de los beneficios de *Colocation* son los siguientes:

- Las preguntas se contestan rápidamente.
- Los problemas se solucionan en ese momento.
- Se produce menor fricción entre las interacciones.
- La confianza se gana con mucha más rapidez.

Las herramientas de *Colaboración* que se pueden utilizar para los equipos colocados o distribuidos son los siguientes:

1. *Colocated Teams* (es decir, los equipos que trabajan en la misma oficina)—En Scrum, es preferible tener equipos colocados. Si los equipos están colocados, los modos de comunicación preferidos incluyen las interacciones, salas de decisión, *War Rooms*, *Scrumboards*, demostraciones en la pared, mesas compartidas, etc.

2. **Distributed Teams** (es decir, los equipos que trabajan en diferentes ubicaciones físicas)—Aunque se prefieren los equipos en un mismo lugar, a veces el *Equipo Scrum* se puede distribuir debido a la subcontratación, la deslocalización, las diferentes ubicaciones físicas, las opciones de trabajo desde casa, etc. Algunas herramientas que podrían utilizarse para tener *Colaboración* eficaz entre los equipos distribuidos incluyen la videoconferencia, mensajes instantáneos, chats, redes sociales, pantallas compartidas y herramientas de software que simulan la funcionalidad de *Scrumboards*, pantallas de pared, y así sucesivamente.

2.6 Prioritization basado en valor

El marco de Scrum es impulsado por el objetivo de ofrecer el máximo valor empresarial en un período de tiempo mínimo. Una de las herramientas más eficaces para realizar el mayor valor en el menor tiempo posible es la *Prioritization*.

Prioritization se puede definir como la determinación del orden y de la separación de lo que debe hacerse ahora, de lo que hay que hacer después. El concepto de *Prioritization* no es nuevo para la gestión de proyectos. El modelo tradicional de gestión del proyecto llamdo Cascada (*Waterfall*) propone el uso de múltiples herramientas de *Prioritization*. Desde el punto de vista del gestor del proyecto, *Prioritization* es integral porque ciertas tareas deben llevarse a cabo primero para acelerar el proceso de desarrollo y la consecución de los objetivos del proyecto. Algunas de las técnicas tradicionales de la *Prioritization* de tarea incluyen el establecimiento de plazos para las tareas delegadas y la utilización de *Prioritization* de matrices.

Scrum, sin embargo, utiliza la *Prioritization* basada en valor como uno de los principios básicos que impulsa la estructura y funcionalidad de todo el marco Scrum - ayuda a que los proyectos se beneficien a través de la capacidad de adaptación y desarrollo iterativo del producto o servicio. Más significativamente, Scrum tiene como finalidad entregar un producto o servicio valioso para el *customer* de forma oportuna y continua.

La *Prioritization* es hecha por el *Propietario del producto* cuando él/ella prioriza los *User Stories* en el *Prioritized Product Backlog*. *Prioritized Product Backlog* contiene una lista de todos los requisitos necesarios para llevar el proyecto a buen término.

Una vez que el *Propietario del producto* ha recibido los *Business Requirements* del *customer* y los ha escrito en forma de *User Stories* viables, él/ella trabaja con el *Customer* y patrocinador (*sponsor*) para entender los *Business Requirements* que prorcionan el máximo valor empresarial. El *Propietario del producto* debe entender lo que el *client* quiere y valora con el fin de organizar los elementos (*User Stories*) del *Prioritized Product Backlog*, según su importancia. A veces, un *customer* puede ordenar que todos los *User Stories* sean de alta prioridad. En ese caso, la propia lista de alta prioridad de *User Stories* también debe ser priorizada. El otorgarle prioridad a una acumulación (*backlog*) implica determinar la importancia de cada *User Story*. Los requisitos de alto valor se identifican y se trasladan a la parte superior del *Prioritized Product Backlog*. Los procesos por los cuales el principio de *Prioritization* basada en valor se pone en

práctica son *Creación de la lista priorizada de pendientes del producto* y *Mantenimiento de la lista priorizada de pendientes del producto*.

Al mismo tiempo, el *Propietario del producto* debe trabajar con el *Equipo Scrum* para entender los *risks* del proyecto y la incertidumbre, ya que estos pueden tener consecuencias negativas. Estos riesgos se deben de tener en cuenta al priorizar *User Stories* con enfoque basado en valores (consulte el capítulo de *Risk* para más información). El *Equipo Scrum* también alerta al *Propietario del producto* de las dependencias que surgen de la aplicación. Estas dependencias se deben tener en cuenta durante la *Prioritization*. La *Prioritization* se puede basar en una estimación subjetiva del valor del negocio proyectado o rentabilidad, o puede estar basada en los resultados y análisis del mercado utilizando herramientas, incluyendo pero no limitado a, las entrevistas del *customer*, encuestas y modelos financieros y técnicas analíticas.

El *Propietario del producto* tiene que traducir las entradas y las necesidades de los proyectos de los *socios* para crear el *Prioritized Product Backlog*. Por lo tanto, mientras se priorizan los user stories en *Prioritized Product Backlog*, se consideran los siguientes tres factores (véase la Figura 2-7):

1. Valor
2. Riesgo o incertidumbre
3. Dependencias

Así, *Prioritization* resulta en entregables que satisfacen los requisitos del *customer* con el objetivo de ofrecer el máximo valor de negocio en el menor tiempo posible.

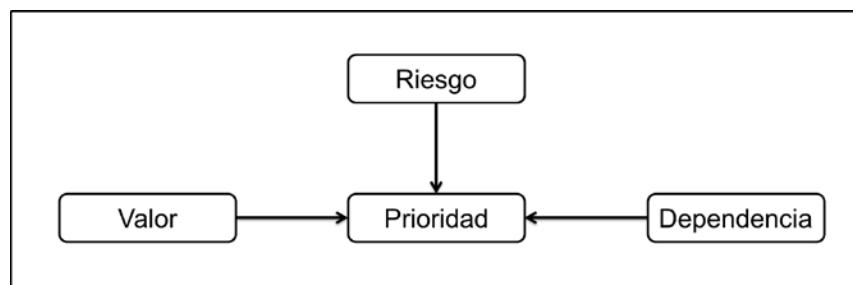


Figura 2-7: *Value-based Prioritization (Prioritization basado en valor)*

2

2.7 Tiempo asignado

Scrum trata al tiempo como uno de los obstáculos más importantes en la gestión de un proyecto. Para hacer frente a la restricción del tiempo, Scrum introduce un concepto llamado *Tiempo asignado* que propone la fijación de una cierta cantidad de tiempo para cada proceso y actividad en un proyecto Scrum. Esto garantiza que los miembros del *Equipo Scrum* no ocupen demasiado o muy poco tiempo por un trabajo determinado, y que no desperdicien su tiempo y energía en un trabajo para el cual tienen poca claridad.

Algunas de las ventajas de *Tiempo asignado* son los siguientes:

- Proceso de desarrollo eficiente
- Menos gastos generales
- Alta velocidad para los equipos

Tiempo asignado puede ser utilizado en muchos procesos de *Scrum*, por ejemplo, en el proceso de *Llevar a cabo el Standup diario*, la duración del *Daily Standup Meeting* es *time-boxed*. A veces, *Tiempo asignado* se puede utilizar para evitar la mejora excesiva de un elemento (es decir, *gold-plating*).

Tiempo asignado es una práctica crítica en *Scrum* y debe aplicarse con cuidado. Un *Tiempo asignado* arbitrario puede llevar a la desmotivación del equipo y puede tener como consecuencia la creación de un entorno aprensivo, por lo que *Tiempo asignado* debe ser utilizado de manera apropiada.

2.7.1 Scrum Time-boxes

- **Sprint**—Un *Sprint* es una iteración *Time-boxed* de una a seis semanas de duración durante el cual el *Scrum Master* guía, facilita y protege al *Equipo Scrum* de *Impediments* tanto internos como externos durante el proceso de *Crear entregables*. Esto ayuda a evitar el arrastramiento, o lentitud, de la visión que podría afectar la meta del *Sprint*. Durante este tiempo, el equipo trabaja para convertir las necesidades del *Prioritized Product Backlog* en funcionalidades de productos fáciles de enviar. Para obtener los máximos beneficios de un proyecto *Scrum*, siempre se recomienda mantener el *Sprint Time-boxed* a 4 semanas, a menos que existan *projects* con requisitos muy estables, donde los *Sprints* se pueden extender hasta 6 semanas.
- **Daily Standup Meeting**—Esto es una reunión diaria de corta duración, *Time-boxed* a 15 minutos. Los miembros del equipo se reúnen para informar sobre como marcha el *project*, respondiendo a las siguientes tres preguntas:
 1. ¿Qué terminé ayer?
 2. ¿Qué voy a terminar hoy?
 3. ¿Qué obstáculos, *Impediments* (si los hay), estoy enfrentando en la actualidad?

Esta reunión se lleva a cabo por el equipo como parte del proceso de *Llevar a cabo el Standup diario*.

- **Sprint Planning Meeting**— Esta reunión se lleva a cabo antes del *Sprint*, como parte del proceso de *Sprint Backlog*. Es *time-boxed* a ocho horas durante un *Sprint* de un mes. El *Sprint Planning Meeting* se divide en dos partes:
 1. Definición del objetivo—Durante la primera mitad de la reunión, el *Propietario del producto* explica la máxima prioridad de los *User Stories* o requisitos del *Prioritized Product Backlog*

para el *Equipo Scrum*. El *Equipo Scrum* en Colaboración con el *Propietario del producto* luego define el objetivo del *Sprint*.

2. Estimación del trabajo—Durante la segunda mitad de la reunión, el *Equipo Scrum* decide como completar los *Prioritized Product Backlog* seleccionados para cumplir con la meta del *Sprint*.

A veces, a los *Task Planning Meetings* (realizados durante el proceso de *Elaboración de tareas*) y los *Task Estimation Meetings* (llevados a cabo durante el *Estimar tareas Process*) también se les conoce como *Sprint Planning Meetings*.

- ***Sprint Review Meeting***—El *Sprint Planning Meeting* es *Time-boxed* a cuatro horas en un *Sprint* de un mes. Durante el *Sprint Review Meeting* que se lleva a cabo en el proceso de *Demostración y validación del Sprint*, el *Equipo Scrum* le presenta los entregables del *Sprint* actual al *Propietario del producto*. El *Propietario del producto* revisa el *product* (o incremento del producto) para compararlos con los *Acceptance Criteria* acordados y él/ella acepta o rechaza los *User Stories* concluidos.
- ***Retrospectiva de Sprint Meeting***—es *Time-boxed* a 4 horas para un *Sprint* de un mes y se lleva a cabo como parte del proceso *Sprint Retrospect*. Durante esta reunión, el *Equipo Scrum* revisa y reflexiona sobre el *Sprint* anterior en relación a los procesos seguidos, las herramientas empleadas, *Colaboración*, y los mecanismos de comunicación, al igual que otros aspectos de interés para el *proyecto*. El equipo discute lo que salió bien durante el *Sprint* anterior y lo que no salió bien, con el objetivo de aprender y mejorar los *Sprints* futuros. Algunas oportunidades de mejora o las mejores prácticas de esta reunión también podrían actualizarse como parte de los documentos de *Cuerpo de asesoramiento de Scrum*.

La figura 2-8 ilustra las duraciones de *Time-boxed* para las reuniones relacionadas con Scrum.

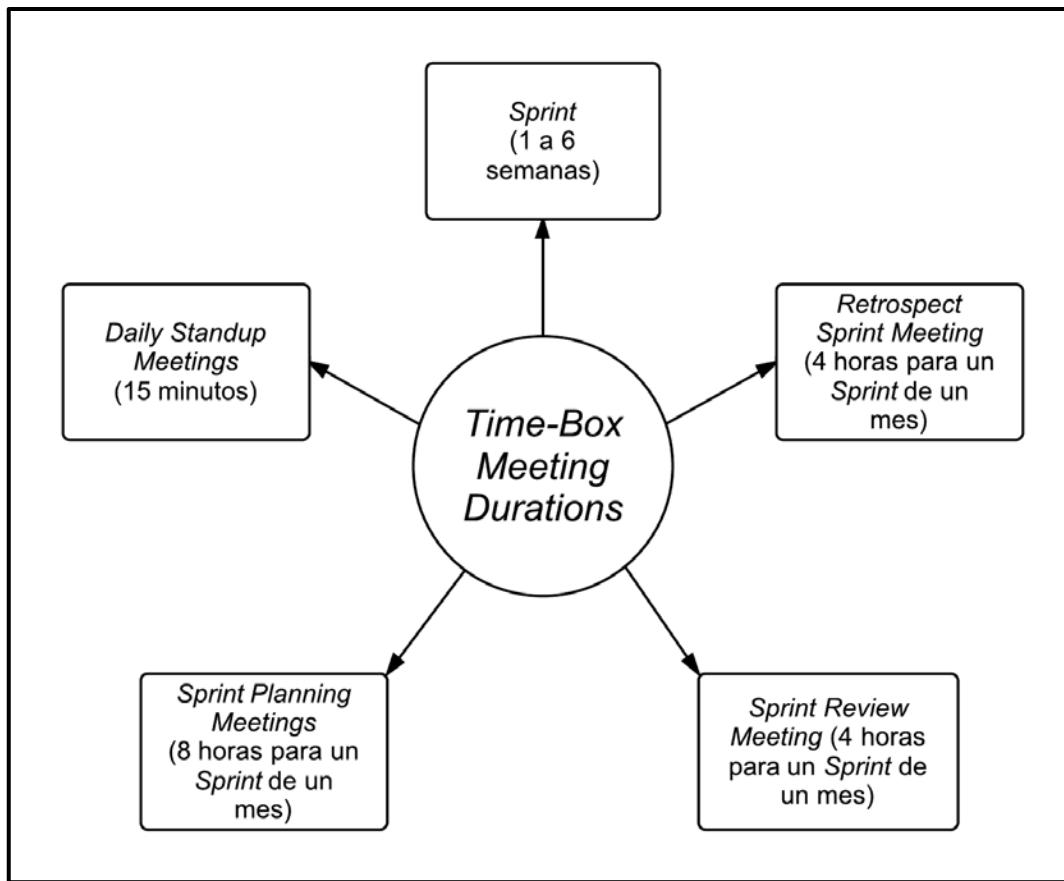


Figura 2-8: Duración de *Time-Box* para las reuniones de Scrum

2.8 Desarrollo iterativo

El marco de Scrum es impulsado por el objetivo de ofrecer el máximo valor empresarial en un período de tiempo mínimo. Para lograr esto de forma práctica, Scrum cree en Desarrollo iterativo *of Deliverables* (*entregas de desarrollo iterativas*).

En la mayoría de los proyectos complejos, el *Customer* puede que no sea capaz de definir unos requisitos muy concretos o puede no estar seguro de cómo debería ser el *product* final. El modelo iterativo es más flexible para asegurar que cualquier cambio solicitado por el *customer* pueda ser incluido como parte del *project*. Es posible que los *User Stories* tengan que ser escritos constantemente a lo largo de la duración del *project*. En las etapas iniciales de la escritura, la mayoría de los *User Stories* son las funcionalidades de alto nivel. Estos *User Stories* son conocidos como *Epic(s)*. *Epics*, por lo general son muy grandes para que los equipos los completen en un sólo *Sprint* y por lo tanto se dividen en pequeños *User Stories*.

Cada aspecto complejo del proyecto se divide mediante la elaboración progresiva durante el proceso de *Mantenimiento de la lista priorizada de pendientes del producto*. Los procesos de *Elaborar historias de usuario* y *Estimate, Approve, and Commit User Stories* se utilizan para agregar nuevos requisitos para el *Prioritized Product Backlog*. La tarea del *Propietario del producto* es asegurar un mayor retorno de la inversión (ROI), centrándose en el valor y la entrega continua con cada *Sprint*. El *Propietario del producto* debería tener una buena comprensión del *Justificación del negocio* y el valor que el proyecto se supone debe entregar cuando redacta el *Prioritized Product Backlog*, y por lo tanto decidir qué entregables contractuales y valores se han de entregar en cada *Sprint*. Luego, los procesos de *Elaboración de tareas*, *Estimar tareas*, y *Elaboración de la lista de pendientes del Sprint* producen el *Sprint Backlog* lo cual el equipo utiliza para crear los entregables.

En cada *Sprint*, el proceso de *Crear entregables* se utiliza para desarrollar las salidas del *Sprint*. El *Scrum Master* tiene que garantizar que se siguen los procesos de Scrum y facilitar al equipo el trabajo de la manera más productiva. El *Equipo Scrum* se auto-organiza y tiene como objetivo crear *Sprint Deliverables* de los *User Stories* en el *Sprint Backlog*. En grandes *projects*, varios equipos multifuncionales funcionan en paralelo a través de *Sprints*, entregando soluciones potencialmente entregables al final de cada *Sprint*. Después de que cada *Sprint* se ha completado, el *Propietario del producto* acepta o rechaza los entregables basados en los *Acceptance Criteria* del proceso *Demostración y validación del Sprint*.

Como se ilustra en la Figura 2-9, los proyectos Scrum se completan de manera iterativa entregando valor a lo largo del ciclo de vida del proyecto.

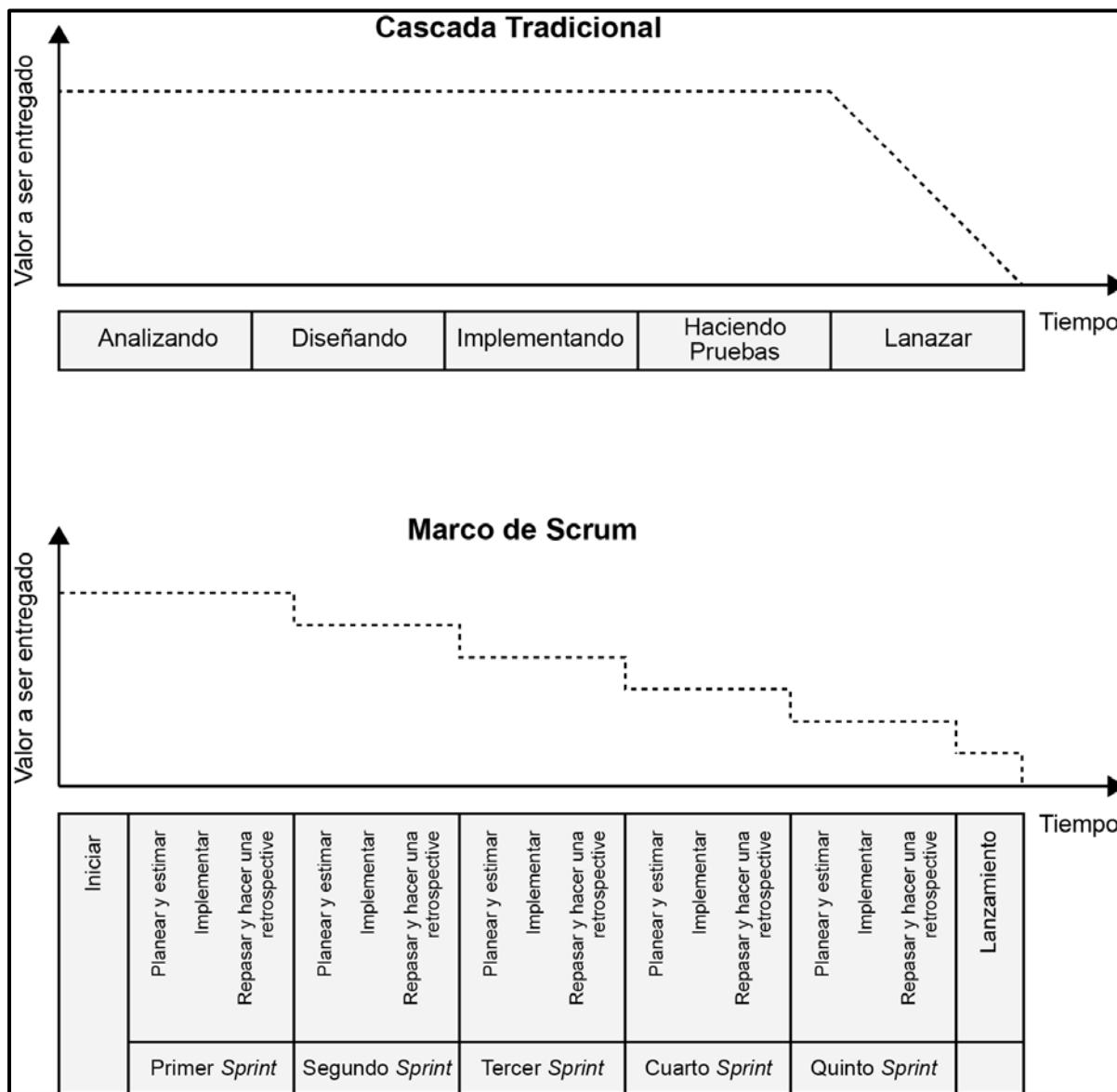


Figura 2-9: Scrum vs Cascada tradicional

El beneficio del desarrollo iterativo es que permite la corrección a medida que todas las personas involucradas obtengan una mejor comprensión de lo que debe ser entregado como parte del *project*, e incorporen lo aprendido de manera iterativa. Así, el tiempo y el esfuerzo requerido para alcanzar el punto final definitivo, se reduce considerablemente y el equipo produce entregables que se adaptan mejor al entorno empresarial.

2.9 Scrum vs Gestión tradicional de proyectos

El énfasis en la gestión de proyectos tradicional es llevar a cabo la planificación detallada por adelantada del proyecto con énfasis en gestionar y solucionar el alcance, costo, horarios y gestión de esos parámetros. La gestión tradicional de *projects* a veces puede llevar a una situación en la que, aunque el plan se ha logrado, el *Customer* no está satisfecho.

El Marco de Scrum se basa en la creencia de que los trabajadores del día de hoy pueden ofrecer mucho más que sus conocimientos técnicos, y que tratar de asignar y plenear en un entorno de constante cambio no es eficiente. Por lo tanto, Scrum anima la toma de decisiones iterativa basada en datos. En Scrum, el enfoque principal es la entrega de productos que satisfagan los requisitos del *customer* en pequeños incrementos iterativos que sean despachables.

Para entregar la mayor cantidad de valor en el menor tiempo posible, Scrum promueve *Prioritization* y *Tiempo asignado* sobre la fijación del alcance, costo y cronograma de un proyecto. Una característica importante de Scrum es *Auto-organización* lo cual le permite a los empleados que estimen y tomen posesión de las tareas.

3. ORGANIZACIÓN

3.1 Introducción

En esta sección, vamos a discutir las diversas facetas de una organización de proyecto Scrum al igual que los roles principales y *Non-core roles* y cómo formar *Equipos Scrum* de alto rendimiento.

Organización, tal como se define en *Una guía para el conocimiento de Scrum (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolios, programs* y/o proyectos de *cualquier sector*
- *Products*, servicios o cualquier otro resultado que se entregará a los *socios*
- *Projects* de cualquier tamaño y complejidad

El término "producto" (product) en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria-desde pequeños proyectos o equipos con tan sólo seis miembros por equipo, hasta proyectos grandes y complejos con cientos de miembros por equipo.

Este capítulo está dividido en las siguientes secciones:

3.2 Guía de los Roles—Esta sección identifica qué sección o subsección es importante para un *Propietario del producto*, *Scrum Master* y *Equipo Scrum*.

3.3 Roles de un proyecto Scrum—Esta sección cubre el núcleo fundamental y *Non-Core Roles* asociados con un *project Scrum*.

3.4 Propietario del producto—En esta sección se destacan las principales responsabilidades del *Propietario del producto* en relación con un *project Scrum*.

3.5 Scrum Master—Esta sección se centra en las principales responsabilidades del *Scrum Master* en el contexto de un *project Scrum*.

3.6 Equipo Scrum—Esta sección hace hincapié en las responsabilidades claves del *Equipo Scrum* en el contexto de un *project Scrum*.

3.7 Scrum en Projects, Programs, y Portfolios—Esta sección se centra en cómo un marco de Scrum puede adaptarse y utilizarse en los diferentes contextos de los *programs* y *portfolios*. También se destacan las responsabilidades específicas de los miembros del *Equipo Scrum* en relación con la comunicación, la integración y el trabajo con los equipos de las empresas y de gestión de *programs*.

3.8 Responsabilidades—En esta sección se describen las responsabilidades pertinentes al tema de la organización, para todos los que trabajan en un *project*, en función de sus roles.

3.9 Scrum vs Gestión de Proyecto Tradicional—Esta sección explica las principales diferencias y ventajas del modelo Scrum en relación con el modelo de Cascada (Waterfall) tradicional de gestión de *projects*.

3.10 Las teorías de recursos humanos populares y su relevancia para Scrum— Esta sección contiene algunas de las teorías de recursos humanos más populares de utilidad para todos los miembros del Equipo Principal/Central de Scrum.

3.2 Guía de los roles

1. *Propietario del producto*—Es imperativo que el *Propietario del producto* lea todo el capítulo.
2. *Scrum Master*—El *Scrum Master* también debe estar familiarizado con todo este capítulo con enfoque principal en las secciones 3.3, 3.5, 3.6, 3.8 y 3.10.4.
3. *Equipo Scrum*—El *Equipo Scrum* debe centrarse principalmente en las secciones 3.3, 3.6, y 3.8.

3.3 Roles de un proyecto Scrum

Entender los roles y las responsabilidades definidas es muy importante para asegurar la implementación exitosa de los proyectos Scrum.

Los roles de Scrum se dividen en dos categorías:

1. ***Core Roles***—Los *Core Roles* son los papeles que obligatoriamente se requieren para producir el producto del proyecto, estos papeles están comprometidos con el proyecto, y por último son los responsables del éxito de cada *Sprint* del proyecto y del proyecto en sí.
2. ***Non-core Roles***—*Non-core Roles* son las funciones que no son obligatoriamente necesarias para el proyecto Scrum, y pueden incluir miembros de los equipos que están interesados en el *project*, pero no tienen ningún papel formal en el equipo del *project*. Ellos pueden interactuar con el equipo, pero no son responsables del éxito del *project*. Los *non-core roles* también deben tenerse en cuenta en cualquier *project* de Scrum.

3.3.1 Core Roles

Hay tres *Core Roles* (roles/papeles principales) en Scrum que son en última instancia responsables de cumplir con los objetivos del *project*. Los *core roles* son el *Propietario del producto*, *Scrum Master*, y el *Equipo Scrum*. Juntos se les conoce como el Equipo Central/Principal de Scrum (*Scrum Core Team*). Es importante tener en cuenta que, de estos tres papeles, ningún papel tiene autoridad sobre los otros.

1. Propietario del producto

El *Propietario del producto* es la persona responsable de maximizar el valor del negocio para el proyecto. Él/ella es responsable de articular los requisitos del *Customer* y de mantener el *Justificación del negocio* del proyecto. El *Propietario del producto* representa la *voz del cliente* (*Voice of the Customer*).

De la misma forma que está el papel de *Propietario del producto* en un proyecto, podría haber un *Program Propietario del producto* en un *program*, o un *Portfolio Propietario del producto* en un *Portfolio*.

2. Scrum Master

El *Scrum Master* es un facilitador que asegura que el *Equipo Scrum* esté dotado de un ambiente propicio para completar con éxito el desarrollo del producto. El *Scrum Master* guía, facilita y les enseña prácticas de Scrum a todos los involucrados en el proyecto, elimina los *impediments* que enfrenta el equipo; y asegura que se estén siguiendo los procesos de Scrum.

Tenga en cuenta que el papel del *Scrum Master* es muy diferente a la función desempeñada por el director de proyecto en un modelo de Cascada tradicional de gestión de proyectos, en la que el director de proyecto trabaja como gerente o líder del proyecto. El *Scrum Master* sólo funciona como un facilitador y él/ella está en el mismo nivel jerárquico que cualquier otra persona en el *Equipo Scrum* – cualquier personal del *Equipo Scrum* que aprenda a facilitar proyectos Scrum puede convertirse en el *Scrum Master* de un proyecto o *Sprint*.

Correspondiente al papel de *Scrum Master* en un proyecto, también podría haber un *Program Scrum Master* para un *program*, o un *Portfolio Scrum Master* para un *portfolio*.

3. Equipo Scrum

El *Equipo Scrum* es un grupo o equipo de personas que son responsables de la comprensión de los *Business requirements* especificados por el *Propietario del producto*, la estimación de *User Stories* y la creación final de los Entregables (*Deliverables*) del *project*.

La figura 3-1 presenta un resumen de los roles principales del *Core Equipo Scrum*.

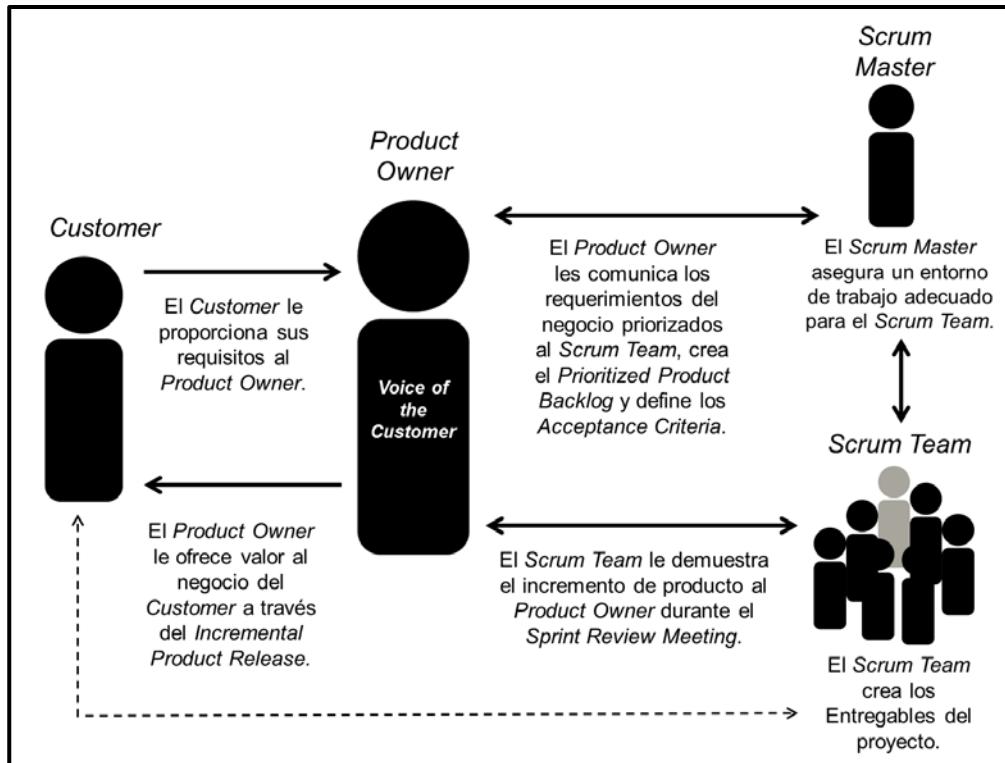


Figura 3-1: Roles de Scrum - Descripción General

3.3.2 Non-core Roles

Non-core roles son aquellos papeles que no son obligatoriamente necesarios para el proyecto Scrum y pueden no estar involucrados en el proceso de Scrum. Sin embargo, es de importancia saber sobre estos *Non-core roles* ya que podrían desempeñar un papel importante en algunos *projects* Scrum.

Los *Non-core roles* pueden incluir los siguientes:

1. Stakeholder(s)

Stakeholder(s) es un término colectivo que incluye a los *customers*, los usuarios y patrocinadores, que a menudo interactúan con el *Propietario del producto*, *Scrum Master* y *Equipo Scrum* para proporcionarles las entradas (*inputs*) y facilitar la creación del producto del *project*, servicio, o cualquier otro resultado. El/los *stakeholder(s)* influyen en el *project* a lo largo del desarrollo del proyecto. Los *socios* también pueden desempeñar un papel en los procesos importantes de Scrum tales como: *Desarrollo de épica(s)*, *Creación de la lista priorizada de pendientes del producto*, *Realizar el plan de lanzamiento* y *Retrospectiva de Sprint*

- **Customer**

El *customer* es la persona o la organización que adquiere el producto del proyecto, servicio, o cualquier otro resultado. Para cualquier organización, dependiendo del *project*, no puede haber

dos *customers* internos (es decir, dentro de la misma organización) o *customers* externos (es decir, fuera de la organización).

- **Usuarios**

El usuario es el individuo o la organización que utiliza directamente el producto del proyecto, servicio, o cualquier otro resultado. Al igual que los *customers*, para cualquier organización, no puede haber dos usuarios internos ni externos. También, en algunas industrias los *customers* y los *usuarios* pueden ser los mismos.

- **Patrocinador**

El patrocinador es la persona o la organización que provee recursos y apoyo para el proyecto. El patrocinador es también el *stakeholder*, a quien todos le deben rendir cuentas al final.

A veces, la misma persona u organización puede desempeñar múltiples funciones – el *stakeholder*, por ejemplo, el promotor y el *customer* puede ser el mismo.

2. Vendedores

Los vendedores incluyen a individuos u organizaciones externas que ofrecen productos y servicios que no están dentro de las competencias básicas de la organización del *project*.

3. Cuerpo de asesoramiento de Scrum

El *Cuerpo de asesoramiento de Scrum* (SGB) es una función opcional. Por lo general, se compone de un grupo de documentos y/o un grupo de expertos que normalmente están involucrados en la definición de los objetivos relacionados con la calidad, las regulaciones gubernamentales, la seguridad y otros parámetros clave de la organización. Estos objetivos guían la labor llevada a cabo por el *Propietario del producto*, *Scrum Master*, y *Equipo Scrum*. El *Cuerpo de asesoramiento de Scrum* también ayuda a capturar las mejores prácticas que se deben utilizar en todos los *projects* Scrum en la organización.

El *Cuerpo de asesoramiento de Scrum* no toma decisiones relacionadas con el *project*. En cambio, actúa como una consultoría o una estructura de orientación para todos los niveles de jerarquía en el *project* de organización del *portfolio*, *program* y *proyect*. Los *Equipos Scrum* tienen la opción de pedirle ayuda al *Scrum Guidance of Body* sobre cualquier asesoramiento requerido.

3.4 *Propietario del producto*

El *Propietario del producto* representa los intereses de la comunidad de *Socios* al *Equipo Scrum*. El *Propietario del producto* es responsable de asegurar una comunicación clara sobre el *product* y los requisitos de funcionalidad de servicios con el *Equipo Scrum*, al igual que definir el *Acceptance Criteria*, y de asegurar que se cumplan esos criterios. En otras palabras, el *Propietario del producto* es responsable de

asegurar que el *Equipo Scrum* ofrezca valor. El *Propietario del producto* siempre debe mantener una visión dual. Él/ella debe entender y apoyar las necesidades e intereses de todos los *socios*, mientras que comprenden las necesidades y el funcionamiento del *Equipo Scrum*. Debido a que el *Propietario del producto* debe entender las necesidades y prioridades de los *socios*, incluyendo los *customers* y los usuarios, este papel se conoce comúnmente como la Voz del *Customer*.

La Tabla 3-1 resume las responsabilidades del *Propietario del producto* en los diferentes procesos de Scrum.

Proceso	Responsabilidades del <i>Propietario del producto</i>
8.1 Crear la visión del proyecto	<ul style="list-style-type: none"> • Define la Visión del Proyecto • Ayuda a crear el <i>Project Charter</i> y el <i>Project Budget</i>
8.2 Identificar al <i>Scrum Master</i> y a el/los <i>Stakeholder(s)</i>	<ul style="list-style-type: none"> • Ayuda a finalizar la elección del <i>Scrum Master</i> para el proyecto • Identifica al/ a los <i>Stakeholder(s)</i>
8.3 Formar el <i>Equipo Scrum</i>	<ul style="list-style-type: none"> • Ayuda a determinar los miembros del <i>Equipo Scrum</i> • Ayuda a desarrollar un <i>Colaboración Plan</i> • Ayuda a desarrollar el <i>Team Building Plan</i> con el/los <i>Scrum Master(s)</i>
8.4 Desarrollo de épica(s)	<ul style="list-style-type: none"> • Crea <i>Epic(s)</i> y <i>Personas</i>
8.5 Crear el <i>Prioritized Product Backlog</i>	<ul style="list-style-type: none"> • Prioriza los elementos de <i>Prioritized Product Backlog</i> • Define el <i>Done Criteria</i>
8.6 Realizar el plan de lanzamiento	<ul style="list-style-type: none"> • Crea <i>Release Planning Schedule</i> • Ayuda a determinar el <i>Length of Sprint</i>
9.1 Elaborar historias de usuario	<ul style="list-style-type: none"> • Ayuda a <i>Elaborar historias de usuario</i> • Define el <i>Acceptance Criteria</i> para cada <i>User Story</i>
9.2 Aprobar, Estimar y Comprometerse a los <i>User Stories</i>	<ul style="list-style-type: none"> • Aprueba los <i>User Stories</i> • Facilita al <i>Equipo Scrum</i> y se compromete a los <i>User Stories</i>
9.3 Elaboración de tareas	<ul style="list-style-type: none"> • Le explica los <i>User Stories</i> al <i>Equipo Scrum</i>, mientras crea el <i>Task List</i>
9.4 Estimar tareas	<ul style="list-style-type: none"> • Le proporciona orientación y aclaración al <i>Equipo Scrum</i> sobre la estimación de esfuerzo para las tareas
9.5 Elaboración de la lista de pendientes del <i>Sprint</i>	<ul style="list-style-type: none"> • Le aclara los requisitos al <i>Equipo Scrum</i> mientras crea el <i>Sprint Backlog</i>
10.1 Crear entregables	<ul style="list-style-type: none"> • Le aclara el <i>Business Requirements</i> al <i>Equipo Scrum</i>
10.3 Mantenimiento de la lista priorizada de pendientes del producto	<ul style="list-style-type: none"> • Mantiene el <i>Prioritized Product Backlog</i>
11.2 Demostración y validación del <i>Sprints</i>	<ul style="list-style-type: none"> • Acepta / Rechaza los Entregables • Proporciona retroalimentación necesaria para el <i>Scrum Master</i> y los <i>Equipo Scrum</i> • Actualiza el Plan de Lanzamiento y el <i>Prioritized Product Backlog</i>

12.1 Envío de entregables	• Ayuda con el lanzamiento de los productos y coordina esto con el <i>Customer</i>
12.2 Retrospectiva del proyecto	• Participa en <i>Retrospective Sprint Meetings</i>

Tabla 3-1: Responsabilidades del *Propietario del producto* en los procesos de Scrum

Las demás responsabilidades de un *Propietario del producto* son:

- Determinar los requisitos generales iniciales del *project* y dar inicio a las actividades del *project*; esto puede implicar la interacción con el *Program Propietario del producto* y *Portfolio Propietario del producto* para asegurar que el *project* se alinea con la dirección proporcionada por la alta dirección.
- Representar a los usuarios del *product* o servicio con un profundo conocimiento de la comunidad de usuarios.
- Asegurar los recursos financieros del *project*.
- Centrarse en la creación de valor y en general de *Return on Investment (ROI)*.
- Evaluar la viabilidad y garantizar la entrega del producto o servicio.

3

3.4.1 Voice of the Customer (VOC)

Como representante del *customer*, se dice que el *Propietario del producto* es la Voz del Cliente (*Voice of the Customer*), ya que es quien asegura que las necesidades explícitas e implícitas del *Customer* sean reflejadas en *User Stories* en el *Prioritized Product Backlog* y más tarde se utilicen para crear los Entregables del *project* para el *Customer*.

3.4.2 Jefe Propietario del producto

En el caso de grandes *projects* con numerosos *Equipos Scrum*, el tener un *Jefe Propietario del producto* puede ser algo necesario. Esta función se encarga de coordinar el trabajo de múltiples *Propietario del producto*. Es el *Jefe Propietario del producto* quien prepara y mantiene el *Prioritized Product Backlog* en su totalidad para los *projects* grandes, usándolo para así coordinar el trabajo a través de los *Propietario del producto* de los *Equipos Scrum*. Los *Propietario del producto*, a su vez, gestionan sus respectivas partes del *Prioritized Product Backlog*.

El *Jefe Propietario del producto* también se comunica con el *Program Propietario del producto* para asegurar la alineación del proyecto con las metas y objetivos del programa.

3.5 Scrum Master

El *Scrum Master* es el *Servant Leader* del *Equipo Scrum* quién modera y facilita las interacciones del equipo como entrenador del equipo y motivador. El *Scrum Master* es responsable de asegurarse que el equipo tenga un ambiente de trabajo productivo mediante la protección del equipo de las influencias externas, la eliminación de todos los obstáculos, y de confirmar que se cumplan los principios, aspectos y procesos de Scrum.

La Tabla 3-2 resume las responsabilidades del *Scrum Master* en los diferentes procesos de Scrum.

Procesos	Responsabilidades del <i>Scrum Master</i>
8.2 Identificar al <i>Scrum Master</i> y al/a los <i>Stakeholder(s)</i>	<ul style="list-style-type: none"> Ayuda a identificar al/a los <i>Stakeholder(s)</i> para el proyecto
8.3 Formar el <i>Equipo Scrum</i>	<ul style="list-style-type: none"> Facilita la selección del <i>Equipo Scrum</i> Facilita la creación del <i>Colaboración Plan</i> y el <i>Team Building Plan</i> Asegura que los recursos de respaldo están disponibles para el funcionamiento del proyecto sin problemas
8.4 Desarrollo de épica(s)	<ul style="list-style-type: none"> Facilita la creación de <i>Epic(s)</i> y <i>Personas</i>
8.5 Creación de la lista priorizada de pendientes del producto	<ul style="list-style-type: none"> Ayuda al <i>Propietario del producto</i> en la creación del <i>Prioritized Product Backlog</i> y en la definición de los <i>Done Criteria</i>
8.6 Realizar el plan de lanzamiento	<ul style="list-style-type: none"> Coordina la creación del <i>Release Planning Schedule</i> Determina el <i>Length of Sprint</i>
9.1 Elaborar historias de usuario	<ul style="list-style-type: none"> Asiste al <i>Equipo Scrum</i> en la creación de <i>User Stories</i> y sus <i>Acceptance Criteria</i>
9.2 Approve, Estimate and Commit User Stories	<ul style="list-style-type: none"> Facilita reuniones del <i>Equipo Scrum</i> para estimar y <i>Elaborar historias de usuario</i>
9.3 Elaboración de tareas	<ul style="list-style-type: none"> Facilita al <i>Equipo Scrum</i> en la creación del <i>Task List</i> para el próximo <i>Sprint</i>
9.4 Estimar tareas	<ul style="list-style-type: none"> Asiste al <i>Equipo Scrum</i> en estimar el esfuerzo necesario para completar las tareas acordadas para el <i>Sprint</i>
9.5 Elaboración de la lista de pendientes del <i>Sprint</i>	<ul style="list-style-type: none"> Asiste al <i>Equipo Scrum</i> en el desarrollo del <i>Sprint Backlog</i> y el <i>Sprint Burndown Chart</i>
10.1 Crear entregables	<ul style="list-style-type: none"> Apoya al <i>Equipo Scrum</i> en la creación de los <i>Entregables</i> (<i>Deliverables</i>) acordados para el <i>Sprint</i> Ayuda a actualizar el <i>Scrumboard</i> y el <i>Impediment Log</i>
10.2 Llevar a cabo el Standup diario	<ul style="list-style-type: none"> Asegura que el <i>Scrumboard</i> y el <i>Impediment Log</i> permanezcan actualizados
10.3 Mantenimiento de la lista priorizada de pendientes del producto	<ul style="list-style-type: none"> Facilita la reuniones de revisión de <i>Prioritized Product Backlog</i>

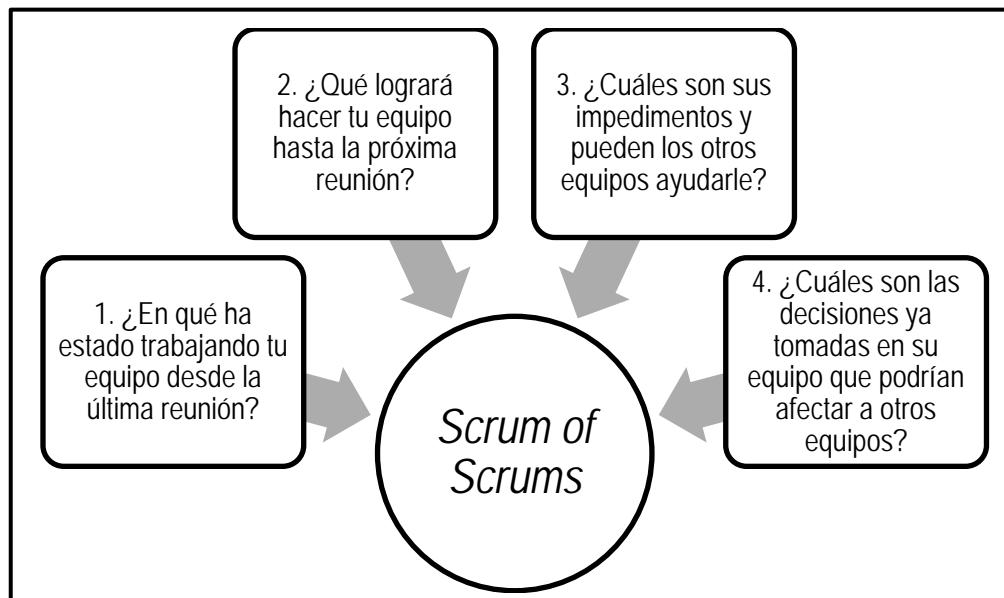
11.1 Convocar Scrum de Scrums	<ul style="list-style-type: none"> • Se asegura que los <i>Issues</i> que afectan al <i>Equipo Scrum</i> se discutan y resuelvan
11.2 Demostración y validación del Sprints	<ul style="list-style-type: none"> • Facilita la presentación de los Entregables ya completados por el <i>Equipo Scrum</i> para la aprobación del <i>Propietario del producto</i>
11.3 Retrospectiva de Sprint	<ul style="list-style-type: none"> • Se asegura que exista un ambiente ideal para el <i>Equipo Scrum</i> del proyecto en los sucesivos <i>Sprints</i>
12.2 Retrospectiva del proyecto	<ul style="list-style-type: none"> • Representa al Equipo Principal de Scrum (<i>Scrum Core Team</i>) para proporcionar lecciones del proyecto actual, si es necesario

Tabla 3-2: Responsabilidades del *Scrum Master* en los procesos de Scrum

3.5.1 Jefe Scrum Master

Los grandes proyectos requieren que varios *Equipos Scrum* trabajen en paralelo. Es muy posible que la información obtenida por un equipo les tenga que ser comunicada adecuadamente a otros equipos. El *Jefe Scrum Master* es el responsable de esta actividad.

La coordinación entre distintos *Equipos Scrum* que trabajan en un proyecto *Scrum* se realiza normalmente a través de *Scrum of Scrums (SoS) Meeting* (referirse a la sección 3.7.2.1). Esto es análogo al *Daily Standup Meeting* y se ve facilitado por el *Jefe Scrum Master*. El *Jefe Scrum Master* suele ser el individuo responsable de resolver los *impediments* que afectan a más de un *Equipo Scrum*.

Figura 3-2: proporciona preguntas que se hacen durante un *Scrum of Scrums (SoS) Meeting*.Figura 3-2: Las preguntas formuladas durante un *Scrum of Scrums Meeting*

Por lo general, cualquier *issue* que haya mismo en el equipo (inter-equipo) son discutidos y resueltos por las propias partes interesadas. Esto se hace en una sesión, inmediatamente después de la reunión de *Scrum of Scrums*, la cual es facilitada por el *Jefe Scrum Master*.

3.6 Equipo Scrum

El *Equipo Scrum* es referido a veces como el equipo de desarrollo, ya que son responsables del desarrollo del producto, servicio, o cualquier otro resultado. Consiste en un grupo de personas que trabajan en los *User Stories* en el *Sprint Backlog* para crear los entregables del *project*.

La Tabla 3-3 resume las responsabilidades del *Equipo Scrum* en los diversos procesos de Scrum.

Procesos	Responsabilidades del <i>Equipo Scrum</i>
8.3 Formar el <i>Equipo Scrum</i>	<ul style="list-style-type: none"> Proporciona entradas (<i>inputs</i>) para la creación de <i>Colaboración Plan</i> y del <i>Team Building Plan</i>
8.4 Desarrollo de épica(s)	<ul style="list-style-type: none"> Asegura una comprensión clara de <i>Epic(s)</i> y de <i>Personas</i>
8.5 Prioritized Product Backlog	<ul style="list-style-type: none"> Entiende los <i>User Stories</i> en el <i>Prioritized Product Backlog</i>
8.6 Realizar el plan de lanzamiento	<ul style="list-style-type: none"> Está de acuerdo con los demás miembros del Equipo Principal de Scrum en el <i>Length of Sprint</i> Busca clarificación sobre los nuevos productos o cambios, si los hay, en los productos existentes en el <i>Prioritized Product Backlog</i>.
9.1 Elaborar historias de usuario	<ul style="list-style-type: none"> Le proporciona entradas al <i>Propietario del producto</i> en la creación de <i>User Stories</i>
9.2 Aprobar, Estimar y Comprometerse a los <i>User Stories</i>	<ul style="list-style-type: none"> Estima los <i>User Stories</i> aprobados por el <i>Propietario del producto</i> Se compromete a las <i>User Stories</i> que hay que hacer en un <i>Sprint</i>
9.3 Elaboración de tareas	<ul style="list-style-type: none"> Desarrolla <i>Task List</i> basada en <i>User Stories</i> ya convenidos y dependencias
9.4 Estimar las Tareas	<ul style="list-style-type: none"> Calcula el esfuerzo para las tareas identificadas y si es necesario, actualiza el <i>Tasks Lists</i>
9.5 Elaboración de la lista de pendientes del <i>Sprint</i>	<ul style="list-style-type: none"> Desarrolla el <i>Sprint Backlog</i> y el <i>Sprint Burndown Chart</i>
10.1 Crear entregables	<ul style="list-style-type: none"> Crea Entregables Identifica <i>Risks</i> y ejecuta acciones de <i>Risk Mitigation</i>, si lo hay Actualiza el <i>Impediment Log</i> y las dependencias
10.2 Llevar a cabo el Standup diario	<ul style="list-style-type: none"> Actualiza el <i>Burndown Chart</i>, <i>Scrumboard</i>, y el <i>Impediment Log</i> Discute <i>issues</i> que enfrenta cada miembro y busca soluciones para motivar al equipo

	<ul style="list-style-type: none"> • Identifica <i>Risks</i>, si lo hay • Presenta <i>Change Requests</i>, si se requiere
10.3 Mantenimiento de la lista priorizada de pendientes del producto	<ul style="list-style-type: none"> • Participa en las reuniones de revisión de <i>Prioritized Product Backlog</i>
11.1 Convocar Scrum de Scrums	<ul style="list-style-type: none"> • Proporciona entradas al <i>Scrum Master</i> para los <i>Scrum of Scrum (SoS) Meetings</i>
11.2 Demostración y validación del Sprints	<ul style="list-style-type: none"> • Le demuestra entregables completados al <i>Propietario del producto</i> para su aprobación
11.3 Retrospectiva de Sprint	<ul style="list-style-type: none"> • Identifica <i>opportunities</i> de mejora, si las hay, del <i>Sprint</i> corriente y dice si está de acuerdo sobre las posibles mejoras viables para el próximo <i>Sprint</i>
12.2 Retrospectiva del proyecto	<ul style="list-style-type: none"> • Participa en el <i>Retrospectiva del proyecto Meeting</i>

Tabla 3-3: Responsabilidades del *Equipo Scrum* en los procesos de Scrum

3.6.1 Selección de Personal

La figura 3-3 enumera las características deseables para las funciones básicas de Scrum.

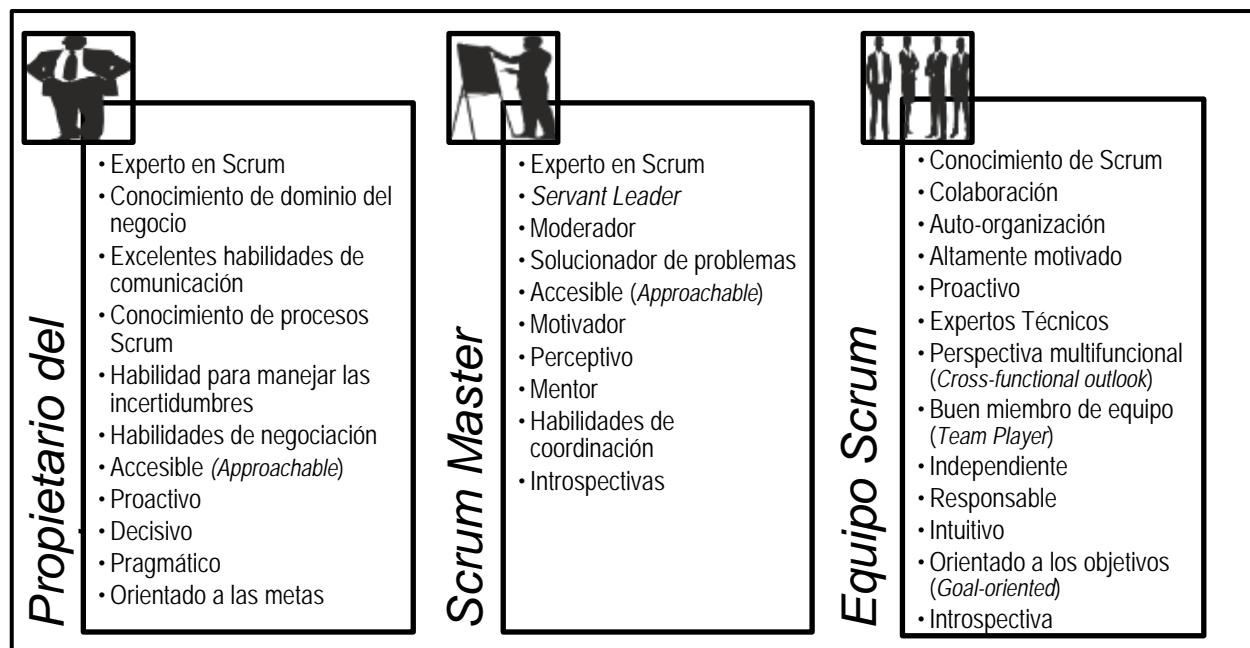


Figura 3-3: Características deseadas de los papeles principales de Scrum

3.6.2 Tamaño del *Equipo Scrum*

Es importante que el *Equipo Scrum* posea todas las habilidades esenciales necesarias para llevar a cabo el trabajo del *project*. También es necesario contar con un alto nivel de *Colaboración* para maximizar la productividad, por lo que se requiere una mínima coordinación para hacer las cosas.

El tamaño óptimo de un *Equipo Scrum* es de seis a diez miembros, lo suficientemente grande para asegurar habilidades adecuadas, pero lo suficientemente pequeño como para colaborar fácilmente. Un beneficio clave de un equipo de seis a diez miembros es que la comunicación y la gestión suelen ser simples y requieren un mínimo esfuerzo. Sin embargo, también puede haber inconvenientes. Una desventaja importante es que los equipos más pequeños se ven afectados más significativamente por la pérdida de un miembro del equipo en comparación a los equipos más grandes, así esta pérdida sea por un corto tiempo. Este problema se puede solucionar si los miembros del equipo tienen conocimientos especializados y habilidades fuera de su papel específico. Sin embargo, esto puede ser difícil y depende del tipo de *project*, la industria, y el tamaño de la organización. También se recomienda tener suplentes para reemplazar a cualquier persona que pueda tener que dejar el *Equipo Scrum*.

3.7 Scrum en Proyectos, Programas, y Portfolios

3.7.1 Definición de Proyecto, Programa, y Portfolio

- **Proyecto (*Project*)**—Un *project* es una empresa de colaboración para crear nuevos productos o servicios, o para obtener resultados como los definidos en el *Declaración de la Visión del Proyecto*. Los proyectos son por lo general afectados por limitaciones de tiempo, costo, alcance, la calidad, la gente y la capacidad de la organización. El objetivo del equipo de proyecto es *Crear entregables*, como se define en el *Prioritized Product Backlog*.
- **Programa (*Program*)**—Un *program* es un grupo de proyectos relacionados con el objetivo de entregar resultados de negocio definidos en el *Program Vision Statement*. El *Prioritized Program Backlog* incorpora el *Prioritized Product Backlog* de todos los proyectos del programa.
- **Portafolio (*Portfolio*)**—Un *portfolio* es un grupo de programas relacionados, con el objetivo de entregar resultados de negocio como se define en el *Portfolio Vision Statement* (Declaración de la Visión del Programa). El *Prioritized Portfolio Backlog* incorpora el *Prioritized Program Backlog* de todos los programas en el *Portfolio*.

Los siguientes son ejemplos de proyectos, programas y portafolios de diferentes industrias y sectores:

Ejemplo 1: Empresa Constructora

- Proyecto—Construcción de una casa
- Programa—Construcción de un complejo de viviendas
- Portafolio—Todos los proyectos de vivienda de la empresa

Ejemplo 2: Organización Aeroespacial

- Proyecto—Construcción del vehículo de lanzamiento
- Programa—Lanzamiento exitoso de un satélite
- Portafolio—Todos los programas de satélites activos

Ejemplo 3: Empresa de TI (Tecnología de la información)

- Proyecto—Desarrollo del módulo de carrito de compras
- Programa—Desarrollo de un sitio web de comercio electrónico (e-commerce) completamente funcional
- Portafolio—Todos los sitios web desarrollados por la empresa hasta ahora

3.7.2 Scrum en Proyectos

Debido a que Scrum favorece a equipos pequeños, uno puede pensar que este método sólo se puede utilizar en proyectos pequeños, pero ese no es el caso. Scrum también se puede utilizar con eficacia en proyectos de escala grande. Cuando se requieren más de diez personas para llevar a cabo el trabajo, se pueden formar múltiples *Equipos Scrum*. El equipo del *project* está formado por múltiples *Equipos Scrum* que trabajan juntos para *Crear entregables* y Entregas del Producto (*Product Releases*), con el fin de lograr los resultados deseados para el proyecto en general.

Dado que un proyecto puede tener múltiples *Equipos Scrum* trabajando en paralelo, la coordinación entre los diferentes equipos se convierte en algo sumamente importante. Los *Equipos Scrum* por lo general se comunican y coordinan entre sí en una variedad de maneras, pero el enfoque más común se conoce como un *Scrum of Scrums (SoS) Meetings*. Los miembros que representan a cada *Equipo Scrum* se reúnen para discutir el progreso, *issues* y para coordinar las actividades entre los equipos. Estas reuniones son similares en formato a los *Daily Standup Meeting*; sin embargo, la frecuencia del *Scrum of Scrums* podría ser en intervalos predeterminados o coordinado tal como es requerido por los diferentes *Equipos Scrum*.

3.7.2.1 Reuniones de *Scrum of Scrums (SoS)*

Un *Scrum of Scrums (SoS) Meeting* es un elemento importante al escalar o ajustar Scrum a proyectos grandes. Por lo general, hay un representante en la reunión de cada uno de los *Equipos Scrum*. Típicamente el representante es el *Scrum Master*, pero también es común para cualquier persona del *Equipo Scrum* asistir a la reunión si es necesario. Esta reunión es usualmente facilitada por el *Jefe Scrum Master* y su objetivo es centrarse en las áreas de coordinación e integración entre los diferentes *Equipos Scrum*. Tal reunión se lleva a cabo en intervalos predeterminados o cuando lo requieran los *Equipos Scrum*.

En organizaciones donde hay varios *Equipos Scrum* trabajando en varias partes de un proyecto a la misma vez, *SoS* se puede escalar a otro nivel de lo que se conoce como *Scrum of Scrums of Scrums Meeting*. En esta situación, un *SoS Meeting* mantiene la coordinación de cada grupo de los *Equipos Scrum* y luego un *Scrum of Scrums of Scrums Meeting* se puede llevar a cabo para coordinar e integrar los *projects* a un nivel mayor. Los equipos tienen que evaluar cuidadosamente los beneficios de contar con *Scrum of Scrums of Scrums Meeting*, ya que la tercera capa añade una cantidad significativa de complejidad logística.

La Figura 3-4 ilustra el concepto de *Scrum of Scrums (SoS)* y el *Scrum of Scrums of Scrums Meetings*

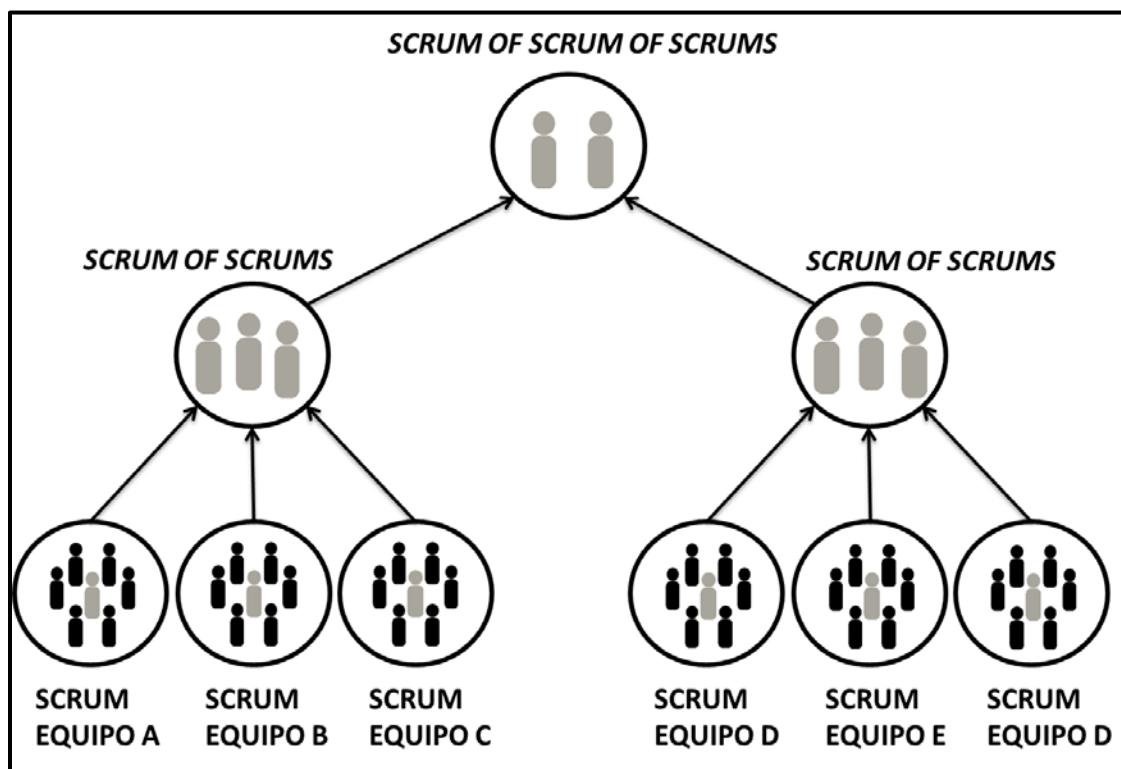


Figura 3-4: Reunión de *Scrum of Scrums (SoS)*

En este ejemplo, hay seis *Equipos Scrum* que trabajan simultáneamente. Los *Equipos Scrum A, B y C* están trabajando en las partes de un *project* relacionado, mientras que los *Equipos Scrum D, E y F* están trabajando en porciones de otro *project* relacionado. Un *Scrum of Scrum Meeting* se lleva a cabo para

coordinar las interdependencias entre los *projects* relacionados. Un *Scrum of Scrums Meeting* se llevará a cabo para coordinar y gestionar las dependencias en todos los *projects*.

3.7.3 Scrum en *Portfolios* y *Programs*

3.7.3.1 Portfolios

En *portfolios*, unos papeles importantes para la gestión de Scrum *Portfolio* son:

1. **Portfolio Propietario del producto**—Define los objetivos estratégicos y las prioridades del *Portfolio*.
2. **Portfolio Scrum Master**—Resuelve problemas, elimina *Impediments*, facilita, y lleva a cabo reuniones para el *Portfolio*.

3

Estas funciones son similares a las del *Propietario del producto* y *Scrum Master* con la diferencia que satisfacen las necesidades de su *Portfolio* o de la empresa en lugar de las de simplemente un *Equipo Scrum*.

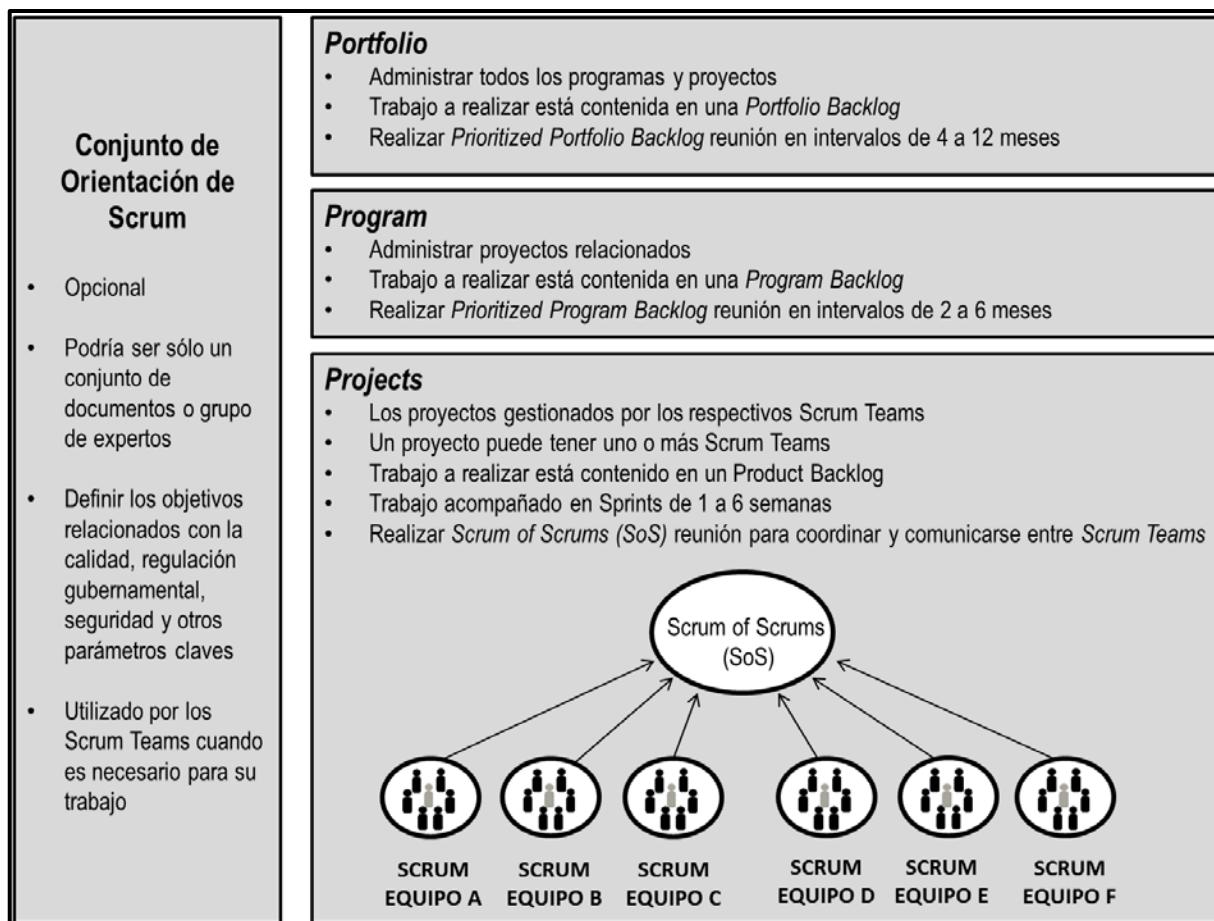
3.7.3.2 Programs

En los *programs*, las funciones importantes para la gestión de programas de Scrum son:

1. **Program Propietario del producto**—define los objetivos y las prioridades estratégicas para el programa.
2. **Program Scrum Master**—Resuelve problemas, remueve *Impediments*, facilita, y lleva a cabo reuniones para el programa.

Estas funciones son similares a las del *Propietario del producto* y *Scrum Master* excepto que satisfacen las necesidades de su programa o unidad de negocio en lugar de los de un sólo *Equipo Scrum*.

Figura 3-5 ilustra cómo Scrum se puede utilizar en toda la organización para los *Portfolios*, *Programs* o *Projects*.

Figura 3-5: Scrum en toda la organización para *projects*, *programs* y *portfolios*

3.7.3.3 Trabajar con *Portfolio* y Equipos de Programas

Al aplicar Scrum para gestionar proyectos en el marco de un programa o un portfolio se recomienda que los principios generales de Scrum que se presentan en esta publicación se cumplan. Se entiende, sin embargo, que con el fin de acomodar el programa en su totalidad o actividades relacionadas con el *Portfolio* e interdependencias, pueden ser necesarios pequeños ajustes en el conjunto de herramientas, así como la estructura organizativa. Si existe un *Cuerpo de asesoramiento de Scrum*, éste puede ser responsable de examinar la organización a diferentes niveles para entender y definir la aplicación adecuada de Scrum, y actuar como facilitador de consulta para todos los que trabajan en un *Project*, *Program* o *Portfolio*.

Los *Portfolios* y *Programs* cuentan con equipos separados y con diferentes conjuntos de objetivos. Los equipos de gestión de *Program* tienen por objetivo ofrecer capacidades y llevar a cabo ciertas metas que contribuyan a objetivos específicos del programa. Por el contrario, el equipo de *Portfolio* tiene que equilibrar los objetivos de los distintos programas para alcanzar los objetivos estratégicos de la organización en su totalidad.

3.7.3.4 La gestión de comunicación con los equipos de *portfolio* y *programs*

Los problemas y los *issue* que se enfrentan al utilizar Scrum dentro de un *Program* o *Portfolio* implican principalmente la coordinación entre los numerosos equipos. Esto puede conducir al fracaso si no se maneja con cuidado. Las herramientas utilizadas para la comunicación deben ampliarse para que coincida con los requisitos de los varios equipos que participan en un *Program* o un *Portfolio*. Cada *Equipo Scrum* debe atender no sólo la comunicación interna, sino también la comunicación externa con otros equipos y los *Socios* pertinentes al *Program* o *Portfolio*.

3.7.4 El mantenimiento de la participación de los *Socios*

Scrum requiere el apoyo completo de los *Socios* de los proyectos. La responsabilidad de mantener a los *Socios* envueltos depende del *Propietario del producto*. Las siguientes son las acciones recomendadas para el mantenimiento de la participación y el apoyo de los *Socios*.

- Asegurar la *Colaboración* efectiva y la participación de los *Socios* en el proyecto
- Evaluar continuamente el impacto en el negocio
- Mantener una comunicación regular con los *Socios*
- Administrar las expectativas de los *Stakeholders*

Un *Stakeholder* clave es el patrocinador—el individuo que provee los fondos y otros recursos para un proyecto. Los patrocinadores quieren entender los resultados financieros relacionados con un producto o servicio, y están por lo general más preocupados por los resultados finales, que con las tareas individuales.

Es importante que los patrocinadores que financian el proyecto estén claros sobre los siguientes *issues*:

- Beneficios de la implementación de Scrum
- Plazos del objetivo y los costos estimados de los proyectos Scrum
- Los *Risks* en general envueltos en la participación de proyectos Scrum y las medidas para mitigarlos
- Fechas de lanzamiento esperadas y entregables finales

3.8 Resumen de responsabilidades

Función	Las responsabilidades
<i>Cuerpo de asesoramiento de Scrum</i>	<ul style="list-style-type: none"> • Establece las directrices generales y las métricas para el desarrollo de descripciones de roles para los miembros del <i>Equipo Scrum</i> • Actua como consultor de proyectos en toda la organización a diferentes niveles • Entiende y define los niveles apropiados de la agrupación, las funciones y las reuniones de los proyectos Scrum
<i>Portfolio Propietario del producto</i>	<ul style="list-style-type: none"> • Define los objetivos y las prioridades de los <i>portfolios</i> estratégicos
<i>Portfolio Scrum Master</i>	<ul style="list-style-type: none"> • Resuelve problemas y coordina reuniones de <i>portfolios</i>
<i>Program Propietario del producto</i>	<ul style="list-style-type: none"> • Define los objetivos estratégicos y las prioridades de los programas
<i>Program Scrum Master</i>	<ul style="list-style-type: none"> • Resuelve problemas y coordina reuniones para los programas
<i>Stakeholder(s)</i>	<ul style="list-style-type: none"> • Es un término colectivo que incluye <i>customers</i>, los usuarios y patrocinadores • Frecuentemente se relaciona con el <i>Propietario del producto</i>, <i>Scrum Master</i>, y <i>Equipo Scrum</i> para proporcionarles las entradas y facilita la creación de los entregables del proyecto.
<i>Propietario del producto</i>	<ul style="list-style-type: none"> • Crea requisitos globales iniciales del proyecto y pone el proyecto en marcha • Selecciona a las personas adecuadas para los roles de <i>Scrum Master</i> y <i>Equipo Scrum</i> • Proporciona los recursos financieros iniciales y luego en curso para el proyecto • Determina la Visión del producto • Evalúa la viabilidad y garantiza la entrega del producto o servicio • Garantiza la transparencia y la claridad de Items de <i>Prioritized Product Backlog</i> • Decide el contenido mínimo del lanzamiento comercial • Proporciona <i>Acceptance Criteria</i> para los <i>User Stories</i> que se desarrollarán en un <i>Sprint</i> • Inspecciona entregables • Decide la duración del <i>Sprint</i>
<i>Scrum Master</i>	<ul style="list-style-type: none"> • Asegura que los procesos de Scrum se sigan correctamente por todos los miembros del equipo, incluyendo el <i>Propietario del producto</i> • Asegura que el desarrollo del producto o servicio esté progresando sin problemas y que los miembros del <i>Equipo Scrum</i> tengan todas las herramientas necesarias para hacer el trabajo • Supervisa reuniones de planificación de lanzamiento y los horarios de otras reuniones
<i>Equipo Scrum</i>	<ul style="list-style-type: none"> • Asume la responsabilidad colectiva y se asegura que los entregables del proyecto se creen según las necesidades • Les asegura al <i>Propietario del producto</i> y al <i>Scrum Master</i> que se está realizando el trabajo asignado de acuerdo al plan

Tabla 3-4: Resumen de las responsabilidades pertinentes a la organización

3.9 Scrum vs gestión de proyectos tradicional

La estructura de la organización y definición de funciones y responsabilidades asociadas son algunas de las áreas en las que Scrum difiere de manera importante de los métodos tradicionales de gestión de *projects*.

En los métodos tradicionales de gestión de proyectos, la estructura de la organización es jerárquica y la autoridad para todos los aspectos del proyecto se delega desde el nivel superior al inferior (por ejemplo, el patrocinador del *project* delega autoridad al gerente del proyecto, y éste a su vez le delega a los miembros del equipo). Los métodos tradicionales de gestión de proyectos hacen hincapié en la responsabilidad del individual hacia las responsabilidades del *project*, en lugar de la apropiación del grupo hacia el proyecto, o de la rendición de cuentas. Cualquier desviación de la autoridad delegada es vista como un signo de *issues* y puede ser escalado a un nivel más alto en la jerarquía de la organización. Por lo general, es el director del proyecto, quien es responsable de la finalización con éxito del *project* y decisiones sobre diversos aspectos del proyecto, incluyendo la iniciación, planificación, estimación, ejecución, seguimiento y control y cierre.

El énfasis en Scrum es *Auto-organización* y la auto-motivación, donde el equipo asume una mayor responsabilidad en la toma de un proyecto exitoso. Esto también asegura que hay un *buy-in* (se cree en lo que se hace) del equipo y propiedad o responsabilidad compartida. Esto, a su vez, da lugar a la motivación del equipo que lleva a una optimización de la eficiencia del equipo. El *Propietario del producto*, *Scrum Master* y el *Equipo Scrum* trabajan de cerca con el/los *stakeholder(s)* pertinente(s) para ajustar los requisitos a medida que avanzan a través de los procesos de *Desarrollo de épica(s)*, *Creación de la lista priorizada de pendientes del producto*, and *Elaborar historias de usuario*. Esto asegura que no hay margen para la planificación aislada en Scrum. El conocimiento y las experiencias del equipo sobre el desarrollo de productos se utilizan para evaluar las entradas necesarias para planificar, calcular y ejecutar el trabajo del proyecto. *Colaboración* entre los miembros del *Scrum Core Team* asegura que el proyecto se lleve a cabo en un entorno innovador y creativo el cual es propicio para el crecimiento y la armonía del equipo.

3.10 Las teorías de HR populares y su relevancia para Scrum

3.10.1 Modelo de Dinámica de Grupo de Tuckman

El enfoque y método de Scrum inicialmente pueden parecer muy diferente y difícil para un nuevo *Equipo Scrum*. Un nuevo *Equipo Scrum*, al igual que cualquier otro equipo nuevo, por lo general se desarrolla a través de un proceso de cuatro etapas durante su primer proyecto de Scrum. Este proceso se conoce como Modelo de Dinámica de Grupo de Tuckman (Tuckman, 1965). La idea principal es que las cuatro etapas—*Forming*, *Storming*, *Norming* y *Performing*—son imprescindibles para que un equipo se desarrolle y mitigue los problemas y desafíos, encuentre soluciones, planifique el trabajo, y entregue resultados.

Las cuatro etapas del modelo son las siguientes:

1. ***Forming*** (Formación)—Esto a menudo se experimenta como un escenario divertido porque todo es nuevo y el equipo aún no ha encontrado alguna dificultad con el proyecto.
2. ***Storming*** (Enfrentamiento)—Durante esta etapa, el equipo trata de cumplir con el trabajo; Sin embargo, puede encontrar conflictos de poder y con frecuencia hay un caos o confusión entre los miembros del equipo.
3. ***Norming*** (Normalización)—Esto es cuando el equipo comienza a madurar, resolver sus diferencias internas, y encontrar soluciones para así trabajar juntos. Se considera un período de ajuste.
4. ***Performing*** (Desempeño)—Durante esta etapa, el equipo está unido y opera en su nivel más alto en términos de rendimiento. Los miembros se han convertido en un equipo eficiente de profesionales que son consistentemente productivos.

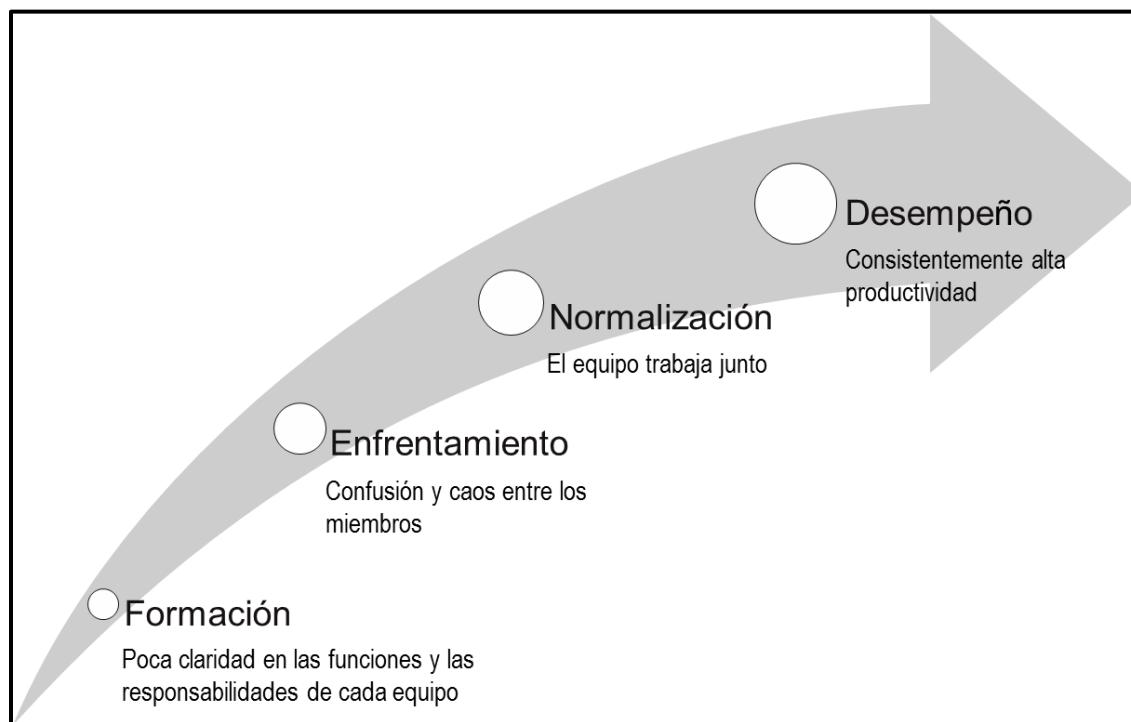


Figura 3-6: Etapas de Tuckman de Desarrollo de Grupos

3.10.2 Manejo de conflictos

Las organizaciones que aplican el marco de Scrum fomentan un ambiente abierto y de diálogo entre los empleados. Los conflictos entre los miembros del *Equipo Scrum* generalmente se resuelven de forma independiente, con poca o ninguna participación de la gerencia o de otros fuera del *Equipo Scrum*.

El conflicto puede ser saludable cuando promueve las discusiones del equipo y alienta debates, ya que por lo general se traduce en beneficios para el *project* y los miembros de los equipos respectivos. Por lo tanto, es importante que la resolución de los conflictos se fomente, promoviendo un entorno abierto donde los miembros del equipo se sientan bienvenidos a expresar con los demás sus inquietudes y opiniones sobre el proyecto, y por último que estén de acuerdo en los entregables y cómo el trabajo se realizará en cada *Sprint*.

Las técnicas de *Conflict Management* son utilizadas por los miembros del equipo para gestionar los conflictos que surgen durante un proyecto Scrum. Las fuentes de conflictos evolucionan principalmente debido a los horarios, prioridades, recursos, informes de jerarquía, cuestiones técnicas, procedimientos, personalidad, y costos.

3.10.3 Técnicas de *Conflict Management*

Por lo general, hay cuatro enfoques para la gestión de conflictos en una organización que aplica procesos de Scrum:

1. Ganar- Ganar
2. Perder-Ganar
3. Perder-Perder
4. Ganar-Perder

3.10.3.1 Ganar-Ganar

Por lo general, es mejor para los miembros del equipo hacerle frente a los problemas directamente con una actitud de cooperación y un diálogo abierto para trabajar a través de cualquier desacuerdo así llegar a un consenso. Este enfoque se denomina Ganar-Ganar (*Win-Win*). Las organizaciones que implementan Scrum deben promover un ambiente donde los empleados se sientan cómodos para hablar y enfrentarse a los problemas o *issues* y trabajar a través de ellos para llegar a los resultados de *Win-Win*.

3.10.3.2 Perder-Ganar

Algunos miembros del equipo a veces pueden sentir que sus contribuciones no son reconocidas ni valoradas por los demás, o que no están siendo tratados por igual. Esto puede conducir a que no contribuyan de manera efectiva al proyecto y estén de acuerdo con todo lo que se les dice que se necesita hacer, incluso si están en desacuerdo. Este enfoque se llama Perder-Ganar (*Lose-Win*). Esta situación puede ocurrir si hay miembros en el equipo (inclusive los administradores) que utilizan un estilo autoritario o directiva de la emisión de órdenes y/o no tratan a todos los miembros del equipo por igual. Este enfoque no es una técnica que se prefiera para *Conflict Management* en los proyectos Scrum, ya que la contribución activa de todos los miembros del equipo es obligatoria para completar con éxito cada uno de los *Sprints*. El *Scrum Master* debe fomentar la participación de los miembros del equipo que parecen estar retirándose de las situaciones de conflicto. Por ejemplo, es importante para todos los miembros del equipo que hablen y contribuyan en cada *Daily Standup Meeting* para que cualquier *issue* o *impediment* se haga saber y sea manejado eficazmente.

3.10.3.3 Perder-Perder

En situaciones de conflicto, los miembros del equipo pueden intentar negociar o buscar soluciones que aporten sólo un grado parcial o medida temporal de satisfacción a las partes en una disputa. Esta situación podría suceder en *Equipos Scrum* si los miembros del equipo tratan de negociar soluciones subóptimas a un problema. Este enfoque suele implicar un poco de "give and take" (dar para recibir) para satisfacer a cada miembro del equipo, en lugar de tratar de realmente resolver el verdadero problema. En general, esto se traduce en un resultado de Perder-Perder (*Lose-Lose*) para los individuos involucrados y en consecuencia el proyecto. El *Equipo Scrum* debe tener cuidado de que los miembros del equipo no lleguen en una mentalidad de perder-perder. Los *Scrum Daily Standups* y otras reuniones de Scrum se llevan a cabo para asegurar que los problemas reales se resuelvan a través de discusiones mutuas.

3.10.3.4 Ganar-Perder

A veces, un *Scrum Master* u otro miembro del equipo influyente pueden creer que él/ella es un líder de facto o gerente y tratar de ejercer su punto de vista, sin tomar en cuenta los puntos de vista de los demás. Esta técnica de *Conflict Management* a menudo se caracteriza por la competitividad y por lo general resulta en Ganar-Perder (*Win-Lose*). Este enfoque no es recomendable cuando se trabaja en proyectos Scrum, porque los *Equipos Scrum* son por naturaleza auto-organizados y con poder, donde ninguna persona tiene verdadera autoridad sobre otro miembro del equipo. Aunque el *Equipo Scrum* puede incluir a las personas con diferentes niveles de experiencia y conocimientos, cada miembro se trata por igual y ninguna persona tiene la autoridad de ser la autoridad máxima.

3.10.4 Estilos de liderazgo

Los estilos de liderazgo varían dependiendo de la organización, la situación, e incluso los individuos y los objetivos del proyecto Scrum específicos. Algunos estilos de liderazgo comunes son las siguientes:

- **Servant Leadership**—*Servant Leadership* implica escuchar cuidadosamente, tener empatía, comprometerse al servicio, tener visión, y compartir el poder y la autoridad con los miembros del equipo. Los *Servant Leaders* logran resultados centrándose en las necesidades del equipo. Este estilo es la realización de la función del *Scrum Master*.
- **Delegating**—*Delegating leaders* están involucrados en la mayor parte de la toma de decisiones, sin embargo delegan parte de la planificación y de la toma de decisiones a los miembros del equipo, sobre todo si son competentes para ejecutar las tareas asignadas. Este estilo de liderazgo es apropiado en situaciones en las que el líder está en sintonía con los detalles de *projects* específicos, y cuando el tiempo es limitado.
- **Autocratic**—*Autocratic leaders* toman decisiones por su cuenta, otorgando poca intervención sobre las decisiones a tomar. Este estilo de liderazgo se debe utilizar solamente en raras ocasiones.
- **Directing**—*Directing leadership* le instruye a los miembros del equipo que tareas se requieren, cuando se deben realizar y la forma en que se deben realizar.
- **Laissez Faire**—Con este estilo de liderazgo el equipo se queda en gran parte sin supervisión, por lo que el líder no interfiere con las actividades laborales diarias. A menudo este estilo lleva a un estado de anarquía.
- **Coaching/Supportive**—Este estilo presenta instrucciones y luego apoya y supervisa a los miembros del equipo al escuchar, ayudar, alentar y presentar una perspectiva positiva en tiempos de incertidumbre.
- **Task-Oriented**—Estos líderes imponen tareas y el cumplimiento de los plazos.
- **Assertive**—*Assertive leaders* confrontan los *issues* y demuestran confianza al establecer autoridad respetuosamente.

3.10.4.1 Servant Leadership

El estilo de liderazgo preferido para los proyectos Scrum es *Servant Leadership*. Este término fue descrito por primera vez por Robert K. Greenleaf, en un ensayo titulado *The Servant as Leader*. A continuación se muestra un extracto en el que se explica el concepto:

El *servant-leader* es primero que nada un servidor... Empieza con el sentimiento natural de que uno quiere servir, servir *primero*. Luego la elección consciente lleva a uno a aspirar a liderar. Esa persona es completamente diferente a aquel que es *líder* primero, tal vez debido a la necesidad de mando inusual o de adquirir posesiones materiales... El líder primero y el *Servant-leader* son dos

tipos extremos. Entre ellos hay matices y mezclas que forman parte de la infinita variedad de la naturaleza humana....

La diferencia se manifiesta en la diligencia del funcionario-primer para asegurarse de que estén siendo atendidas las necesidades más prioritarias de la gente. La mejor prueba, y difícil de administrar, es: ¿A los que se les sirve, crecen como personas? ¿Mientras se les sirve, se vuelven más sanos, más sabios, más libres, más autónomos, más propensos a ser servants (*ser de servicio*) ellos mismos? Y, ¿cuál es el efecto sobre los más desfavorecidos en la sociedad? ¿Van a ser beneficiados o al menos no más desfavorecidos? (Greenleaf 1970, 6)

Elaborando en las escrituras de Greenleaf, Larry Spears identifica diez rasgos que cada *Servant-Leader* eficaz debe poseer:

1. **Escuchar**—Se espera que los *Servant-Leaders* escuchen con atención y sean receptivos a lo que se dice o no se dice. Ellos son capaces de ponerse en contacto con su voz interior para comprender y reflexionar sobre sus propios sentimientos.
2. **Empatía**—Buenos *servant-leaders* aceptan y reconocen a los individuos por sus destrezas únicas y habilidades especiales. Ellos asumen que los trabajadores tienen buenas intenciones y los aceptan como individuos, incluso cuando existen *issues* de comportamiento o rendimiento.
3. **Curación**—La motivación y la capacidad de curarse a sí mismo y la relación con los demás es un fuerte rasgo de *servant-leaders*. Ellos toman la oportunidad de ayudar a sus colegas que están pasando por dolor emocional.
4. **Toma de conciencia**—Ser consciente, especialmente ser auto-conciente es un rasgo de *servant-leaders*. Esto les permite entender mejor e integrar *issues*, tales como los relacionados con la ética, el poder y los valores.
5. **Persuasión**—*Servant-leaders* usan la persuasión, en lugar de su posición de autoridad, para obtener el consenso del grupo y tomar decisiones. En lugar de forzar el cumplimiento y la coerción como es típico en algunos estilos de dirección autoritarios, *servant-leaders* practican la persuasión.
6. **Conceptualización**—Una habilidad especial de los *servant-leaders* es ver y analizar los problemas (en una organización) desde una perspectiva conceptual y visionaria más amplia, en lugar de centrarse en los objetivos inmediatos a corto plazo.
7. **Prospectiva**—Su mente intuitiva le permite al *servant-leader* utilizar y aplicar las lecciones del pasado y la realidad actual para prever el resultado de situaciones y decisiones actuales.
8. **La mayordomía**—Mayordomía exige un compromiso de servir a los demás. El *servant-leader* prefiere la persuasión por el control para obtener la confianza de los demás en la organización.

9. **Compromiso con el crecimiento de los demás**—Los servant-leaders tienen un profundo compromiso con el crecimiento de las personas dentro de su organización. Asumen la responsabilidad de nutrir el crecimiento personal, profesional y espiritual de los demás (por ejemplo, facilitando el acceso a los recursos para el desarrollo personal y profesional, alentando a los trabajadores a participar en la toma de decisiones).
10. **Construcción de la Comunidad**—Servant-leaders están interesados en la construcción de comunidades dentro de un ambiente de trabajo. Esto es de gran importancia en especial dado al cambio en muchas sociedades que dejan de ser comunidades pequeñas para convertirse en grandes instituciones que dan forma y que controlan las vidas humanas.

Scrum cree que todos los líderes de proyectos Scrum (incluyendo al *Scrum Master* y al *Propietario del producto*) deben ser líderes serviciales que tengan las características mencionadas.

3.10.5 La Teoría de Jerarquía de Necesidades de Maslow

Maslow (1943) presenta una jerarquía de necesidades que reconoce que diferentes personas se encuentran en diferentes niveles en sus necesidades. Por lo general, la gente se preocupa de las necesidades fisiológicas y después se desplaza progresivamente hacia las necesidades jerárquicas.

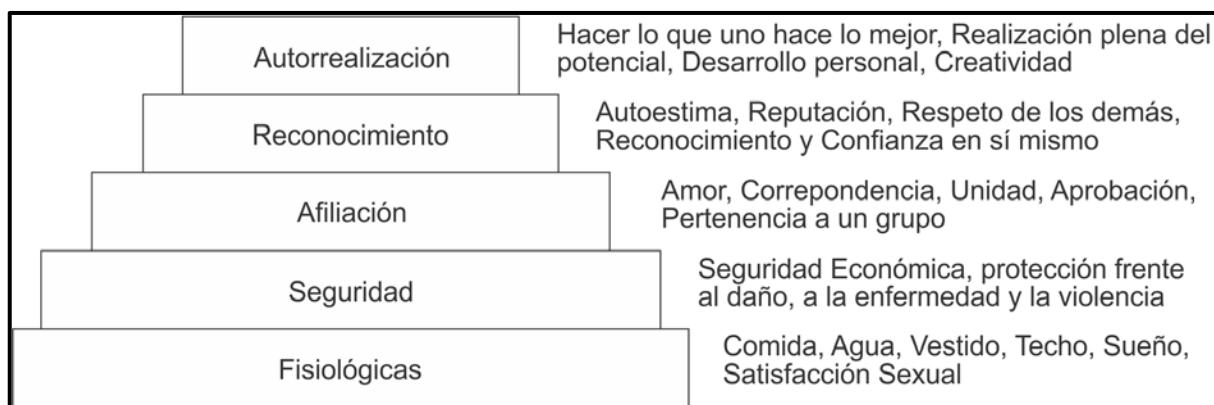


Figura 3-7: Teoría de Jerarquía de Necesidades de Maslow

Para tener éxito, un *Equipo Scrum* necesita que tanto los miembros básicos como no básicos del equipo hayan alcanzado la estima o niveles de autorrealización. El concepto de equipos auto-organizados, lo cual es un principio clave en Scrum, requiere que los miembros del equipo sean auto-motivados, que participen y contribuyan plenamente en el cumplimiento de los objetivos del proyecto.

Como líder, el *Scrum Master* tiene que entender dónde es que cada persona del equipo está en relación a la pirámide. Este entendimiento le ayudará a determinar el mejor enfoque para motivar a cada individuo.

Todas las personas fluctúan hacia arriba y abajo en los niveles de jerarquía de necesidades a lo largo de la vida. Esto puede ser debido a su propia motivación y esfuerzos para avanzar en la jerarquía o, a veces debido a factores fuera de su control. El objetivo del *Scrum Master* es trabajar con personas en el equipo para mejorar sus habilidades y conocimientos y ayudarlo/la a ascender en la jerarquía de necesidades. Este apoyo resulta en un equipo que está formado por individuos que están motivados y son fuertes contribuyentes al project y a la organización.

3.10.6 *Theory X and Theory Y*

Douglas McGregor (1960) propuso dos teorías de gestión:

- ***Theory X***—Los líderes de *Theory X* asumen que los empleados no están motivados intrínsecamente y evitarán el trabajo si es posible, lo que justifica un estilo autoritario de gestión.
- ***Theory Y***—Los líderes de *Theory Y*, por otro lado, asumen que los empleados son personas automotivadas y tratan de aceptar una mayor responsabilidad. *Theory Y* implica un estilo de gestión más participativo.

No es probable que los proyectos Scrum tengan éxito si el *Scrum Master* o el *Propietario del producto* son líderes de *Theory X*. Los líderes de proyectos Scrum deben suscribirse a *Theory Y* y ver a los empleados pos sus cualidades importantes, a la misma vez que se les ayuda a desarrollar sus habilidades y a sentirse con poder. Es de suma importancia también expresarles a los miembros del equipo un gran agradecimiento por el trabajo que han hecho para lograr los objetivos del proyecto.

4. JUSTIFICACIÓN DEL NEGOCIO

4.1 Introducción

El propósito de este capítulo es entender el concepto y el propósito de *Justificación del negocio* en lo que se refiere a los proyectos Scrum. Es importante para una organización llevar a cabo un adecuado *Justificación del negocio* y crear un *Declaración de la Visión del Proyecto* viable antes de iniciar cualquier *project*. Esto ayudará a las personas claves que toman decisiones a entender si hay necesidad de que la empresa proporcione un cambio o un nuevo producto o servicio y proporcionar la justificación para seguir adelante con un *project*. También ayuda a que el *Propietario del producto* cree un *Prioritized Product Backlog*, tomando en cuenta las expectativas del negocio de la Alta Dirección y del/de los *Stakeholder(s)*.

"*Justificación del negocio*", según se define en *Una guía para el conocimiento de Scrum* (Guía SBOK™), es aplicable a los siguientes:

- *Portfolios, programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se les entregará a los socios
- *Projects* de cualquier tamaño y complejidad

El término "producto" en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier *project* en cualquier industria - desde pequeños *projects* o equipos con tan sólo seis miembros por equipo, hasta *projects* grandes y complejos que cuentan con cientos de miembros por equipo.

Este capítulo está dividido en las siguientes secciones:

4.2 Guía de los roles—En esta sección se proporciona orientación sobre qué secciones son relevantes para cada uno de los papeles fundamentales de Scrum: El *Propietario del producto*, *Scrum Master* y *Equipo Scrum*.

4.3 Value-driven Delivery (Entrega basada en el valor)—En esta sección se describe el concepto de valor del negocio y su importancia en cualquier *project*. También proporciona información sobre las responsabilidades de los distintos individuos, incluyendo al *Propietario del producto*, que participan en la consecución de valor para el negocio.

4.4 Importancia de Justificación del negocio—En esta sección se detalla la importancia de *Justificación del negocio*, los factores que la determinan, y cómo se mantiene y se verifica durante todo el proyecto.

4.5 Técnicas de Justificación del negocio—Esta sección describe en detalle cómo *Justificación del negocio* es evaluado y verificado utilizando diversas herramientas.

4.6 Continuous Value Justification—En esta sección se detalla la importancia de *Continuous Value Justification* y se expande en la forma en la que se logra.

4.7 Confirmar la realización de beneficios—En esta sección se describe la forma en la que se logran los beneficios.

4.8 Resumen de las responsabilidades—Esta sección define las responsabilidades pertinentes al *Justificación del negocio* para los miembros del equipo del *project* en función de sus roles.

4.9 Scrum vs Gestión de Proyecto Tradicional—Esta sección destaca los beneficios empresariales del método Scrum con respecto a los modelos tradicionales de gestión de *projects*.

4.2 Guía de Roles

1. *Propietario del producto*—*Justificación del negocio* se lleva a cabo principalmente por el *Propietario del producto*; por lo tanto, todo este capítulo es más aplicable a este papel.
2. *Scrum Master*—El *Scrum Master* debe estar familiarizado con todo este capítulo, con el enfoque principal en las secciones 4.3, 4.4, 4.6, 4.7 y 4.8.
3. *Equipo Scrum*—El *Equipo Scrum* debería centrarse principalmente en las secciones 4.3, 4.7 y 4.8.

4.3 Entrega basada en valor

Un proyecto es una empresa de colaboración para cualquiera que desee crear nuevos *products* o servicios, o para obtener resultados según han sido definidos en el *Declaración de la Visión del Proyecto*. Los *projects* son por lo general afectados por limitaciones de tiempo, costo, alcance, la calidad, la gente, y la capacidad de la organización. Por lo general, se espera que los resultados generados por proyectos resulten en algún tipo de negocio o servicio de valor.

Dado a que el valor es una razón principal de cualquier organización para seguir adelante con un proyecto, Entrega basada en valor (*Value-Driven Delivery*) debe ser el foco principal. El ofrecer valor es algo que está arraigado en el marco de Scrum. Scrum facilita la entrega de valor muy temprana en el *project* y lo sigue haciendo a lo largo del ciclo de vida del *project*.

Una de las características claves de cualquier *project* es la incertidumbre de los resultados. Es imposible garantizar el éxito del *project*, independientemente del tamaño o la complejidad de un *project*. Por lo tanto, teniendo en cuenta esta incertidumbre de alcanzar el éxito, es importante comenzar a obtener resultados exitosos tan pronto sea posible. Esta entrega temprana de buenos resultados, y por lo tanto valor, proporciona una oportunidad para la reinversión y le demuestra el valor del *project* a los socios.

Con el fin de aportar *Value-driven Delivery* (*Entrega basada en valor*), es importante:

1. Entender lo que le agrega valor a los *customers* y a los usuarios y dar prioridad a las necesidades de alto valor del *Prioritized Product Backlog*.
2. Disminuir la incertidumbre y constantemente encargarse de los *Risks* que potencialmente puedan disminuir el valor en caso de materializarse. También es importante trabajar en estrecha colaboración con el *Stakeholder* del *project* y mostrar incrementos de productos al final de cada *Sprint*, lo que permite la gestión efectiva de cambios.
3. *Crear entregables* basados en las prioridades determinadas al producir incrementos potencialmente entregables del producto durante cada *Sprint*. De esta forma, los *customers* empiezan a darse cuenta del valor desde el principio del *project*.

El concepto de *Value-driven Delivery* (*Entrega basada/impulsada en el valor*) en Scrum hace que el marco de Scrum sea muy atractivo para los *Socios* y la alta dirección de negocios. Este concepto es muy diferente en comparación con los modelos tradicionales de gestión de *projects* donde:

1. los requisitos no son priorizados por el valor del negocio.
2. los cambios de requisitos después de la iniciación del proyecto son difíciles y sólo se pueden hacer a través de un proceso de gestión que lleva mucho tiempo.
3. el valor se realiza sólo al final del *project*, cuando se entrega el producto o servicio final.

La figura 4-1 contrasta el *Value-driven Delivery* (*Entrega basada/impulsada por el valor*) en Scrum frente a los *projects* tradicionales.

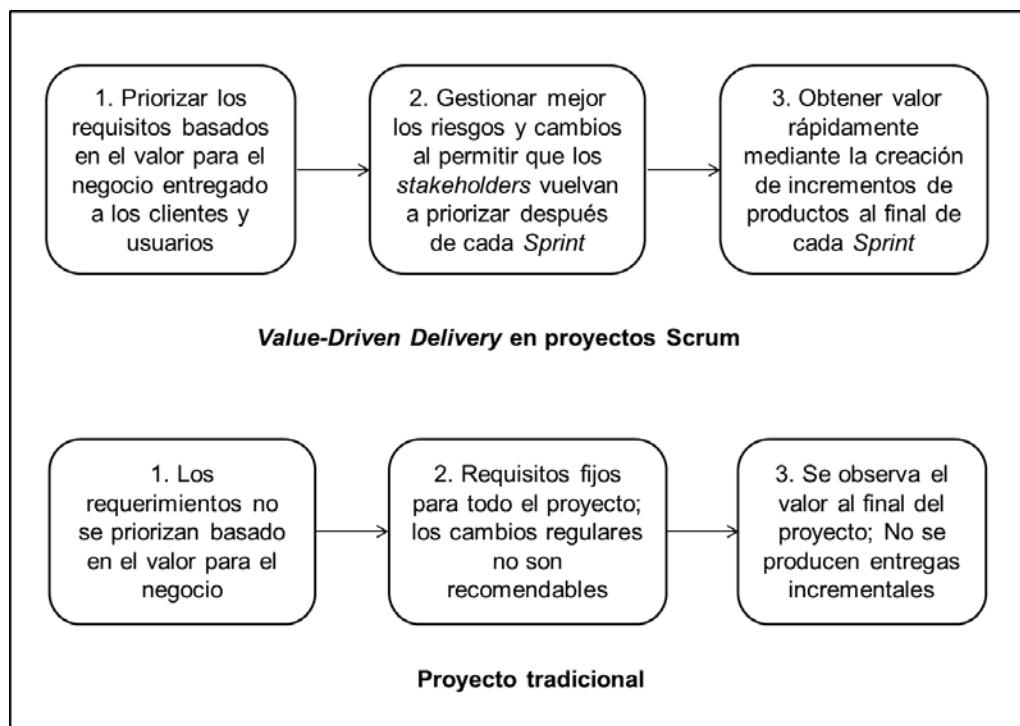


Figura 4-1: Entrega de valor en Scrum vs Proyectos Tradicionales

4.3.1 Las responsabilidades del *Propietario del producto* en la creación de *Justificación del negocio*

La responsabilidad de priorizar y entregar valor de negocio para los proyectos en una organización le corresponde ante todo al *Propietario del producto*. Para los *programs* y *portfolios*, la responsabilidad recae en el *Program Propietario del producto* y *Portfolio Program Owner*, respectivamente. Su función es la de actuar como representantes efectivos del *Customer* y/o patrocinador. Las directrices para la evaluación y medición del valor de negocio típicamente pueden ser establecidas por un *Cuerpo de asesoramiento de Scrum*.

La figura 4-2 ilustra las responsabilidades de *Justificación del negocio* en un orden jerárquico.

<i>Portfolio Propietario del producto</i>	<ul style="list-style-type: none"> • Proporciona el valor de los <i>portfolios</i> • Crea <i>Justificación del negocio</i> para el <i>portfolio</i> • Proporciona una guía de valor para los <i>programs</i> • Aprueba el <i>Justificación del negocio</i> para los <i>programs</i>
<i>Program Propietario del producto</i>	<ul style="list-style-type: none"> • Proporciona el valor de los <i>programs</i> • Crea <i>Justificación del negocio</i> para los <i>programs</i> • Proporciona una guía de valor para los <i>projects</i> • Aprueba el <i>Justificación del negocio</i> para los <i>projects</i>
<i>Propietario del producto</i>	<ul style="list-style-type: none"> • Proporciona el valor de los <i>projects</i> • Crea <i>Justificación del negocio</i> para los <i>projects</i> • Le confirma la realización de beneficios a los <i>Socios</i>

Figura 4-2: Jerarquía de responsabilidades de *Justificación del negocio*

4.3.2 Responsabilidades de los otros roles de Scrum en *Justificación del negocio*

Es importante señalar que si bien el *Propietario del producto* es el responsable principal del *Justificación del negocio*, otras personas que trabajan en el *project Scrum* también contribuyen de manera significativa de la siguiente manera:

1. El **patrocinador (sponsor)** proporciona fondos para el proyecto y supervisa constantemente el *project* para confirmar la realización de beneficios.

2. Los *customers* y *users* (*clientes* y *usuarios*) están involucrados en la definición de la lista de prioridades de los requisitos y de los *User Stories* en el *Prioritized Product Backlog*, de revisar Entregables (*Deliverables*) después de cada *Sprint* o *release* (*lanzamiento*), y de confirmar los beneficios.
3. El *Cuerpo de asesoramiento de Scrum* puede proporcionar directrices y recomendaciones relacionadas con las técnicas de *Justificación del negocio* y confirmar la realización de beneficios y así sucesivamente. Esas directrices y recomendaciones entonces se les puede referir al *Scrum Core Teams* y a los *Stakeholder(s)*.
4. El *Scrum Master* facilita la creación de entregables del proyecto; gestiona *risks*, cambios e *impediments* durante el *Llevar a cabo el Standup diario*, *Retrospectiva de Sprint* y otros procesos de Scrum. El *Scrum Master* se coordina con el *Equipo Scrum* para crear los entregables y con el *Propietario del producto* y otros *Socios* para asegurar que los beneficios del proyecto se realicen.
5. El *Equipo Scrum* trabaja en la creación de los entregables del proyecto y contribuye a la realización de valor del negocio para todos los *Socios* y el proyecto. El *Equipo Scrum* también está involucrado en los siguientes: *Desarrollo de épica(s)*; *Creación de la lista priorizada de pendientes del producto*; *Elaborar historias de usuario*; *Approve, Estimate and Commit User Stories*; al igual que con los procesos asociados donde los *business requirements* se definen y priorizan. El *Equipo Scrum* también ayuda en la identificación de *risks* y presenta *Change Requests* para las mejoras durante el *Sprint Retrospect Meeting* y otras reuniones.

4

4.4 Importancia de *Justificación del negocio*

Justificación del negocio demuestra las razones para emprender un proyecto. Responde la pregunta "¿Por qué es necesario este proyecto?" Es *Justificación del negocio* lo que impulsa todas las decisiones relacionadas con un proyecto. Así que es importante evaluar la viabilidad de un *project*, no sólo antes de comprometerse a gastos o inversiones significativas en las etapas iniciales, sino también a lo largo del ciclo de vida del *project*. Un *project* debe terminarse si se encuentra que es inviable; la decisión debe ser escalada a los *Socios* pertinentes y para la alta dirección. El *Justificación del negocio* de un proyecto debe ser evaluado al inicio del proyecto, en intervalos predefinidos durante todo el proyecto, y en cualquier momento cuando surgen grandes *issues* o *risks* que amenazan la viabilidad del proyecto.

4.4.1 Factores utilizados para determinar *Justificación del negocio*

Existen numerosos factores que un *Propietario del producto* debe tener en cuenta para determinar el *Justificación del negocio* de un *project*. Los siguientes son algunos de los factores más importantes:

1. *Project Reasoning*

Project Reasoning incluye todos los factores que requiere el *project*, ya sea positivo o negativo, elegido o no (por ejemplo, la capacidad insuficiente para satisfacer la demanda existente y prevista, disminución en la satisfacción del *customer*, baja ganancia, requisito legal, etc.)

2. *Business Needs*

Business Needs son los resultados de negocio que debe cumplir el *project*, tal como es documentado en el *Declaración de la Visión del Proyecto*.

3. *Project Benefits*

Project Benefits incluyen todas las mejoras cuantificables en un *product*, servicio o resultado que se podría ofrecer a través de la finalización exitosa de un *project*.

4. *Opportunity Cost*

Opportunity Cost cubre la segunda opción favorecida de negocio o *project* que fue descartada en favor del proyecto actual.

5. Los *risks* principales

Risks se refiere a eventos inciertos o no planificados que puedan afectar la viabilidad y el éxito potencial del *project*.

6. *Project Timescales*

Las escalas de tiempo reflejan la longitud o duración de un *project* y el tiempo durante el cual se realizarán sus beneficios.

7. *Project Costs*

Project Costs es la inversión y otros costos de desarrollo de un *project*.

4.4.2 *Justificación del negocio* y el ciclo de vida del proyecto

Business Justification se evalúa antes de iniciar un *project* y se verifica de forma continua durante todo el ciclo de vida del *project*. Los pasos siguientes captan cómo se determina lo que se llama *Justificación del negocio*:

1. Evaluar y presentar un caso de negocio

Justificación del negocio para un *project* normalmente es analizado y confirmado por el *Propietario del producto*. Está documentado y presentado en la forma de Caso de Negocios de un *project* antes del *Initiate phase* y consiste en considerar los diversos factores especificados en la sección 4.4.1. Una vez documentado, el *Propietario del producto* debe crear una *Declaración de la Visión del Proyecto* y obtener la aprobación de aquellos que toman las decisiones claves en la organización. En general, se trata de ejecutivos y/o una junta de gestión de *program* o *project*.

Continuous Value Justification

Una vez que los encargados de tomar decisiones aprueban el *Declaración de la Visión del Proyecto*, se usa como referencia y se forma el *Justificación del negocio*. El *Justificación del negocio* se valida durante toda la ejecución del *project*, por lo general en intervalos predefinidos, como durante las reuniones de *portfolio*, *program*, y *Prioritized Product Backlog*, las reuniones de revisión y cuando los principales *Issues* y *Risks* que amenazan la viabilidad del proyecto son identificado. Esto podría ocurrir en varios procesos de Scrum incluyendo *Llevar a cabo el Standup diario* y *Mantenimiento de la lista priorizada de pendientes del producto*. A lo largo del proyecto, el *Propietario del producto* debe mantener el *Justificación del negocio* en el *Declaración de la Visión del Proyecto* actualizado con información relevante del proyecto para que los que toman decisiones importantes continúen tomando decisiones informadas.

2. Confirmar la Realización de Beneficios

El *Propietario del producto* confirma el logro de beneficios organizacionales a través del proyecto, así como sobre la terminación de los *User Stories* en el *Prioritized Product Backlog*. Los beneficios de los proyectos Scrum se realizan durante los proceso *Demostración y validación del Sprint*, *Retrospectiva de Sprint*, *Envío de entregables* and *Retrospectiva del proyecto*.

La figura 4-3 resume los pasos para determinar *Justificación del negocio*.

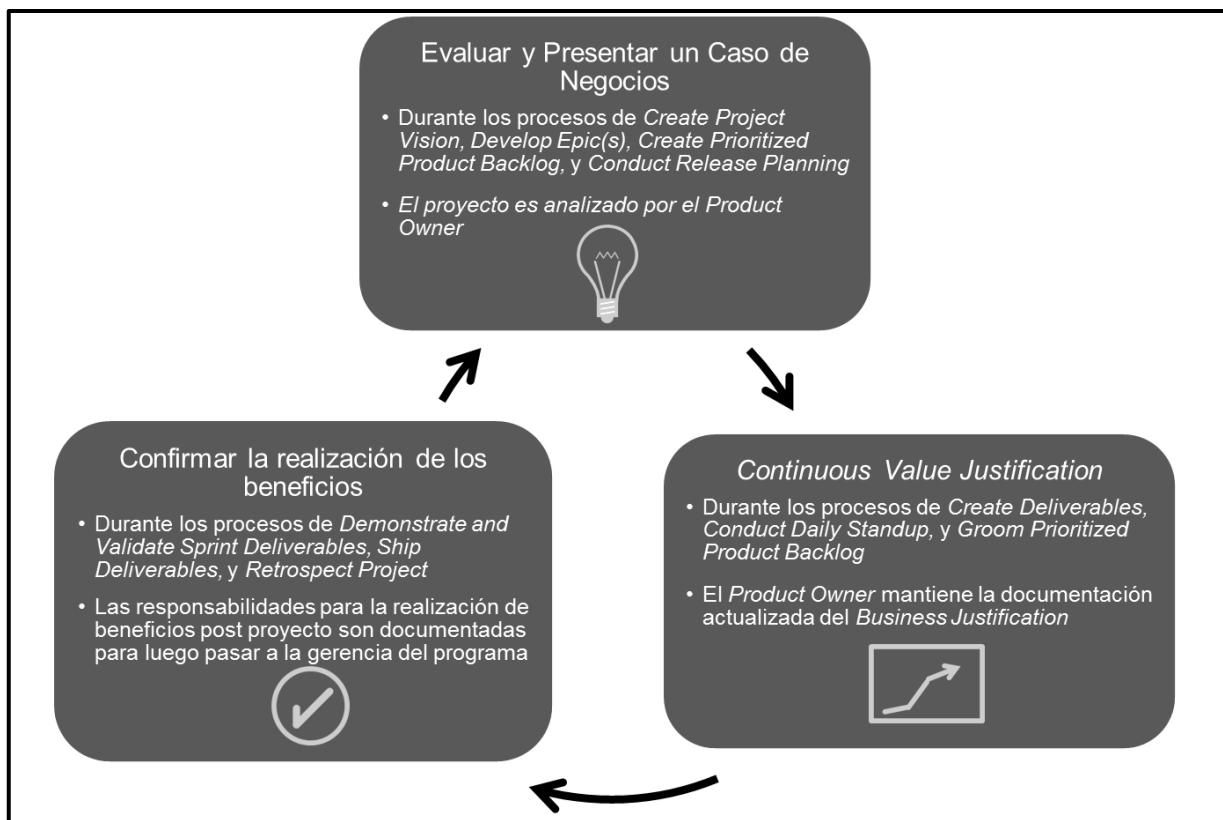


Figura 4-3: Justificación del negocio y el Ciclo de Vida del Proyecto

4.5 Técnicas de *Justificación del negocio*

Las siguientes secciones tratan sobre algunas de las herramientas que se utilizan para valorar y evaluar el *Justificación del negocio*, así como otros aspectos relacionados con la justificación y selección del *project*. No es necesario, ni siquiera recomendable utilizar todas las técnicas disponibles para cada proyecto. Algunas técnicas no son apropiadas dependiendo del proyecto específico, y las técnicas se pueden usar para evaluar los *projects* de forma individual o para comparar el valor esperado de múltiples *projects*.

El *Cuerpo de asesoramiento de Scrum (SGB)*, que puede ser un grupo de expertos o un conjunto de documentos sobre normas y procedimientos de la organización, define los lineamientos y parámetros que se utilizarán para evaluar el valor del negocio. Cada *Propietario del producto*, sin embargo, es responsable de realizar las actividades que comprueban el valor de negocio de sus respectivos *projects, programs* o *portfolios*.

4.5.1 Estimación del valor del proyecto

El posible valor de proyectos empresariales puede estimarse mediante diversos métodos, tales como *Return on Investment (ROI)*, *Net Present Value (NPV)*, e *Internal Rate of Return (IRR)*.

1. *Return on Investment (ROI)*

Return on Investment (ROI), cuando se utiliza para la justificación del *project*, evalúa los ingresos netos que se esperan obtener a partir de un *project*. Se calcula restando los costos o inversiones estimadas de un *project* de su ingreso, y dividiendo esto (beneficio neto) por los costos previstos, con el fin de obtener una tasa de retorno. Otros factores como la inflación y las tasas de interés sobre el dinero prestado se pueden tener en cuenta en los cálculos de *ROI*.

Fórmula ROI:

$$\text{ROI} = (\text{Ingresos del proyecto} - \text{Costo del project}) / \text{Costo del project}$$

Ejemplo: El ROI para un proyecto que tendrá un costo de \$125.000 a desarrollar, con beneficios económicos estimadas en \$300.000, se calcula de la siguiente manera:

$$\text{ROI} = (\$300.000 - \$125.000) / \$125.000 = 1.4$$

Por lo tanto, el ROI es 1,4 veces la inversión (o 140%).

Los incrementos frecuentes de servicio o producto, es una base fundamental de Scrum que permite la verificación temprana de *ROI*. Esto ayuda en la evaluación de la justificación de valor continuo.

2. Net Present Value (NPV)

Net Present Value (NPV) es un método utilizado para determinar el valor neto actual de un futuro beneficio económico, dada una inflación prevista o tasa de interés. En otras palabras, el *NPV* es el ingreso total esperado o los ingresos de un *project*, menos el costo total previsto del *project*, teniendo en cuenta el valor temporal del dinero.

Ejemplo: ¿Cuál de los siguientes dos proyectos es la mejor selección si se utiliza NPV como el criterio de selección?

- El proyecto A tiene un valor actual neto de \$1.500 y se completará en 5 años.
- El proyecto B tiene un valor actual neto de \$1.000 y se completará en dos años.

Solución: Proyecto A, puesto que su valor actual neto es mayor; el hecho de que el proyecto B tenga una duración más corta que el proyecto A no se considera aquí, porque el tiempo ya está representado para los cálculos de valor actual neto (es decir, debido a que es el valor actual, y no el valor futuro que está siendo considerado en el cálculo).

3. Internal Rate of Return (IRR)

Internal Rate of Return (IRR) es una tasa de descuento de una inversión en la que el valor presente de los flujos de efectivo se hace igual al valor presente de los flujos de salida de efectivo para evaluar la tasa de rentabilidad de un proyecto. Al comparar proyectos, uno con un TIR mayor es típicamente mejor.

Aunque *IRR* no se utiliza con frecuencia para justificar *projects* como algunas otras técnicas, tales como VPN, es un concepto importante de saber.

Ejemplo: Basado en IRR, ¿qué proyecto es más deseable?

- Proyecto A, que tiene un IRR del 15% y se completará en 5 años.
- Proyecto B, que tiene una IRR del 10% y se completará en 1 año.

Solución: Proyecto A, puesto que su IRR es mayor; el hecho de que el proyecto B tenga una menor duración que el proyecto A no es considerado aquí porque el tiempo ya está tomado en cuenta en los cálculos de la IRR (es decir, como con NPV, es el valor actual, no el futuro, que se utiliza para determinar el IRR).

4.5.2 Planning for Value

Después de justificar y confirmar el valor de un proyecto, el *Propietario del producto* debería considerar las políticas de la organización, procedimientos, plantillas y normas generales dictadas por el *Cuerpo de asesoramiento de Scrum* (o junta similar del *project* de organización u oficina) en la planificación de un proyecto; al mismo tiempo que se maximiza el *Value Driven Delivery*. La responsabilidad de determinar *cómo* se crea el valor cae en las *socios* (patrocinadores del *customer* y/o los usuarios), mientras que el *Equipo Scrum* se concentra en lo que está por desarrollar. Algunas herramientas comunes recomendadas por un *Cuerpo de asesoramiento de Scrum* podrían ser las siguientes:

1. *Value Stream Mapping*

Value Stream Mapping utiliza diagramas de flujo, para ilustrar el flujo de información necesaria para completar un proceso. Esta técnica se puede utilizar para simplificar un proceso, ayudando a determinar elementos que no añaden valor.

2. *Customer Value-based Prioritization*

Customer Value-based Prioritization le otorga importancia primordial al *customer* y se esfuerza por poner en práctica *User Stories* con el valor más alto primero. Estos *User Stories* con alto valor se identifican y se colocan en la parte superior del *Prioritized Product Backlog*.

Un equipo puede utilizar una variedad de esquemas de *Prioritization* para determinar las características de alto valor.

a. *Simple Schemes*

Simple Schemes implica etiquetar elementos como prioridad "1", "2", "3" o "Alto", "Medio" y "Bajo" y así sucesivamente. Aunque se trata de un método sencillo y directo, puede llegar a ser problemático porque a menudo hay una tendencia a etiquetar todo como prioridad "1" o "Alto". Incluso los métodos de priorización como "alto", "medio" y "bajo" pueden encontrarse con dificultades similares.

b. *MoSCoW Prioritization*

El esquema *MoSCoW Prioritization* deriva su nombre de las primeras letras de las frases "Must have" (debe tener), "Should have" (debería tener), "Could have" (podría tener), y "Won't have" (no tendrá). Este método de *priorization* es generalmente más efectivo que *Simple Schemes*. Las etiquetas están en orden de prioridad decreciente con características de "Debe tener" sin los que el *product* no tendrá valor, y las características de "no tendrá" son las que, a pesar de que sería bueno tener, no se necesitan incluir.

c. *Monopoly Money*

Esta técnica consiste en darle al "*Customer*" "*Monopoly Money*" o "dinero falso" igual a la cantidad del *project budget* y pedirle que lo distribuya entre los *User Stories* bajo

consideración. De esta manera, el *customer* prioriza basado en lo que está dispuesto a pagar por cada *User Story*.

d. 100-Point Method

El *100-Point Method* (Método de 100 Puntos) fue desarrollado por Dean Leffingwell y Don Widrig (2003). Se trata de darle al *customer* 100 puntos que puede usar para votar por las características que considera más importantes.

e. Kano Analysis

El *Kano Analysis* fue desarrollado por Noriaki Kano (1984) y consiste en clasificar las características o requisitos en cuatro categorías en función de las preferencias del *customer*:

1. *Los que complacen o delietan (Delighters)*: Características que son nuevas, o de gran valor para el *customer*
2. *Satisfactores (Satisfiers)*: Características que le ofrecen valor al *customer*
3. *Insatisfactores (Dissatisfiers)*: Características de las cuales, si no están presente, es probable al *customer* no les gusta el *product*, pero no afectan el nivel de satisfacción si están presentes
4. *Indiferente (Indifferent)* : Características que no afectarán al *customer* de ninguna manera y deben ser eliminadas

La figura 4-4 muestra una ilustración de *Kano Analysis*.

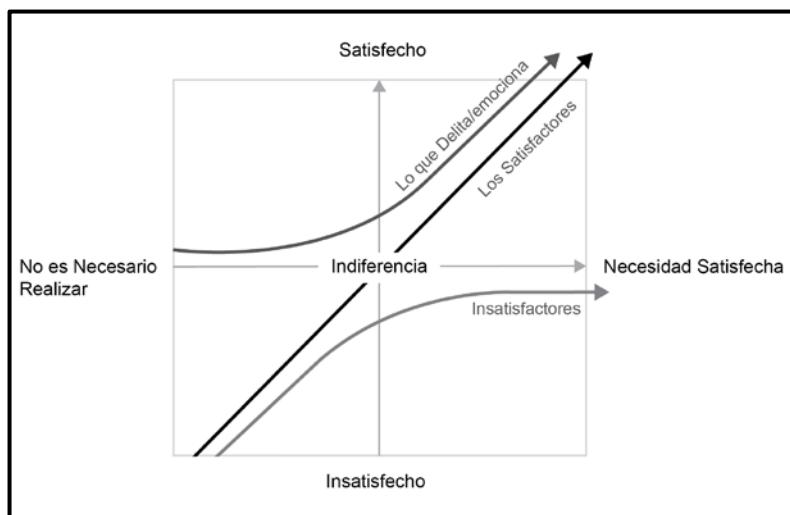


Figura 4-4: *Kano Analysis*

Curiosamente, las características por lo general se mueven hacia abajo en la lista de clasificación; los *customers* esperan ciertas características (por ejemplo, las cámaras en

los teléfonos) y sus funciones pasaron de causar *deleite* a ser *satisfactores* y eventualmente a *insatisfactores*.

4.5.3 *Relative Prioritization Ranking*

Una simple lista de los *User Stories* en el orden de prioridad es un método eficaz para determinar los deseados *User Stories* para cada iteración o versión del *product* o servicio. El objetivo es crear una lista simple, sólo con el objetivo de dar prioridad a las funciones, en lugar de distraerse con múltiples esquemas de prioritization.

Esta simple lista también proporciona una base para incorporar los cambios y *risks* identificados cuando sea necesario. Cada cambio o riesgo identificado se puede insertar en la lista en función de su prioridad relativa a los otros *User Stories* en la lista. Normalmente, los nuevos cambios se incluirán en detrimento de las características que se han asignado como con prioridad más baja.

El definir las Características Mínimas de Mercado (*Minimum Marketable Features - MMF*) es extremadamente importante en este proceso, de modo que la primera versión o iteración ocurre tan pronto como sea posible, lo que lleva a un aumento de rendimiento de la inversión. Normalmente, estos *User Stories* se ubicarían como alta prioridad.

4.5.4 *Story Mapping*

Esta es una técnica para proporcionar un esquema visual del *product* y sus componentes claves. *Story Mapping*, formulada por primera vez por Jeff Patton (2005), es comúnmente utilizado para ilustrar la trayectoria del *product*.

Los mapas de los cuentos representan la secuencia de iteraciones de desarrollo de los productos y trazan las características que se incluirán en la primera, segunda, tercera, y versiones posteriores.

4.6 *Continuous Value Justification*

El valor del negocio se debe evaluar con regularidad para determinar si la justificación o la viabilidad de la ejecución del *project* siguen existiendo. La evaluación frecuente de la inversión en el *project* en relación con el valor del negocio que se está creando califica la viabilidad de un *project*. Los requisitos esperados del *project* pueden cambiar con frecuencia, lo que puede afectar tanto la inversión del *project* como la creación de valor. Un aspecto clave de Scrum es su capacidad para adaptarse rápidamente al caos creado por un modelo de negocio que cambia rápidamente. En los proyectos donde los requerimientos de usuario son ambiguos y los cambios frecuentes, Scrum ofrece ventajas considerables sobre otros modelos de desarrollo.

La monitorización de la tasa de entrega de valor es un requisito importante para los proyectos Scrum. Periódicamente el seguimiento y notificación de la creación de valor ayuda a evaluar el estado del *project* y proporciona información importante para el *customer* y otros *Socios*.

4.6.1 *Earned Value Analysis*

Aunque se usa comúnmente, las herramientas tales como diagramas de barras y diagramas de Gantt tienen limitaciones en el seguimiento y presentación de informes de progreso cuando se trata de rendimiento del *project*. *Earned Value Analysis (EVA)* se usa para este propósito.

EVA analiza el rendimiento real del *project* con respecto al rendimiento previsto en un punto dado en el tiempo. Para que las técnicas de rastreo sean eficaz, el plan inicial del *project* debe ser exacto. *EVA* a menudo utiliza gráficos y otros elementos visuales (por ejemplo, la curva S), como una forma de representar la información sobre el estado del *project*.

Earned Value Analysis mide las variaciones actuales en la agenda del proyecto y el costo de funcionamiento y las previsiones del costo final basado en el rendimiento actual determinado. *EVA* se realiza normalmente al final de cada *Sprint* después que los *User Stories* en el *Sprint Backlog* se han completado.

La tabla 4-1 resume las fórmulas utilizadas en el *Earned Value Analysis*

Definición de Término	Siglas	Fórmula
<i>Planned Value (Valor Planificado)</i>	PV	
<i>Earned Value (Valor Ganado)</i>	EV	
<i>Actual Cost (Costo Actual)</i>	AC	
<i>Budget at Completion (Presupuesto al Finalizar)</i>	BAC	
<i>Schedule Variance (Variación en Programación)</i>	SV	EV - PV
<i>Cost Variance (Variación en Costo)</i>	CV	EV - AC
<i>Schedule Performance Index (Efectividad sobre la Planificación Realizada)</i>	SPI	EV / PV
<i>Cost Performance Index (Indice de Costo de Rendimiento)</i>	CPI	EV / AC
<i>Percent Complete (Porcentaje Completado)</i>	% Complete	(EV / BAC) x 100
<i>Estimate at Completion (Estimación al Finalizar)</i> 1. Estimación no válida 2. Las diferencias actuales son atípicas 3. Las variaciones actuales son típicas	EAC	1. AC + ETC 2. AC + BAC - EV 3. BAC / CPI
<i>Estimate to Complete (Estimado a Completer)</i>	ETC	EAC - AC

<i>Variance at Completion (Variación al Finalizar)</i>	VAC	BAC - EAC
--	-----	-----------

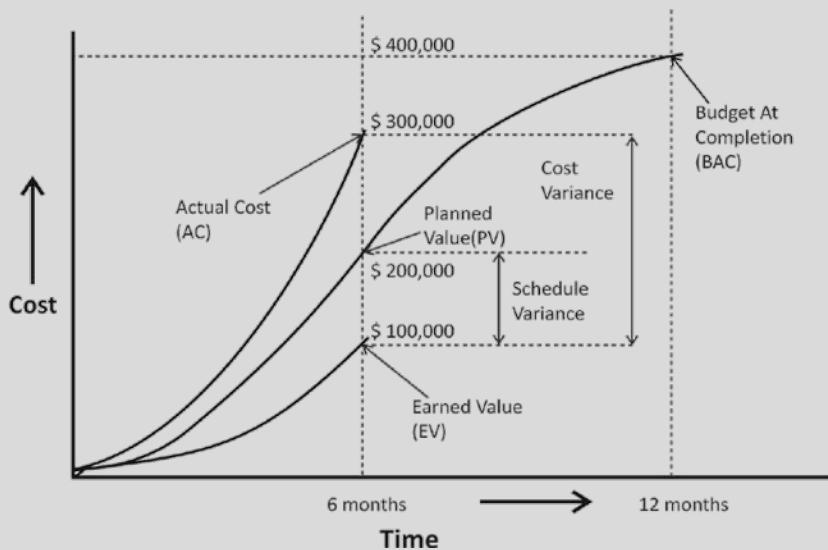
Tabla 4-1: Fórmulas de Valor Ganado

Ejemplo: Un sitio web con 4.000 páginas necesita ser desarrollado - asumimos que cada página web toma el mismo tiempo para completar, y que cada página web es un *User Story* único de igual prioridad en el *Prioritized Product Backlog*. El costo estimado para completar el proyecto es de \$400.000 y el límite de tiempo para el proyecto es de 12 meses. Después de 6 meses, se ha gastado \$300.000 y el trabajo realizado es 1.000 páginas web.

¿Con qué hemos sido proporcionados?

- *Budget at Completion (BAC)* = \$400.000 (*Cost Baseline* para el proyecto)
- *Planned Value (PV)* = \$200.000 (ya que habíamos planeado para completar 2.000 páginas web)
- *Earned Value (EV)* = \$100.000 (valor de 1.000 páginas web que se han completado)
- *Actual Cost (AC)* = \$300.000 (lo que se ha gastado hasta el momento)

Curva S para los datos:



Fórmulas:

- *Schedule Variance (SV)* = $EV - PV = \$100,000 - \$200,000 = -\$100,000$
- *Cost Variance (CV)* = $EV - AC = \$100,000 - \$300,000 = -\$200,000$
 - Las variaciones negativas en nuestro proyecto indican que estamos atrasados en el programa y por encima del presupuesto.
- *Schedule Performance Index (SPI)* = $EV / PV = \$100,000 / \$200,000 = 0.5$
 - SPI < 1 indica que el trabajo realizado hasta ahora es sólo el 50% de lo que habíamos planeado terminar en 6 meses.
- *Cost Performance Index (CPI)* = $EV / AC = \$100,000 / \$300,000 = 0.33$
 - CPI < 1 indica que sólo hemos terminado el 33% por la cantidad de dinero que se ha gastado.
- *Percent Complete* = $EV / BAC \times 100 = \$100,000 / \$400,000 \times 100 = 25\%$
 - Así, el 25% del trabajo del proyecto está completo en este punto en el tiempo.

4.6.2 Cumulative Flow Diagram (CFD)

Un *Cumulative Flow Diagram (CFD)* es una herramienta útil para la elaboración de informes y el seguimiento de los resultados del *project*. Proporciona una representación sencilla y visual del progreso del *project* en un punto determinado. Se utiliza generalmente para proporcionar un estado de mayor nivel de la totalidad del proyecto y no para actualizaciones diarias de *Sprints* individuales.

La figura 4-5 es un ejemplo de un *CFD* para un proyecto grande. Se muestra el número de *user stories* que no se han creado, están en proceso de ser creados, y los que se han creado. A medida que cambian los requisitos de los *customers*, se produce un cambio en los *User Stories* acumulados que han de ser entregados. Los cambios de puntos 1 y 2 es donde el *Propietario del producto* ha eliminado *User Stories* que existen en el *Risk Adjusted Prioritized Product Backlog*, y los puntos 3 y 4 es donde el *Propietario del producto* agregó nuevos *User Stories* en el *Risk Adjusted Prioritized Product Backlog*.

Este tipo de diagrama puede ser una gran herramienta para la identificación de obstáculos y embottellamiento en los procesos. Por ejemplo, si el diagrama muestra una banda cada vez más estrecha, mientras la banda anterior con el tiempo está cada vez más amplia, puede haber un embottellamiento y pueden ser necesarios los cambios para aumentar la eficiencia y/o mejorar el desempeño del *project*.

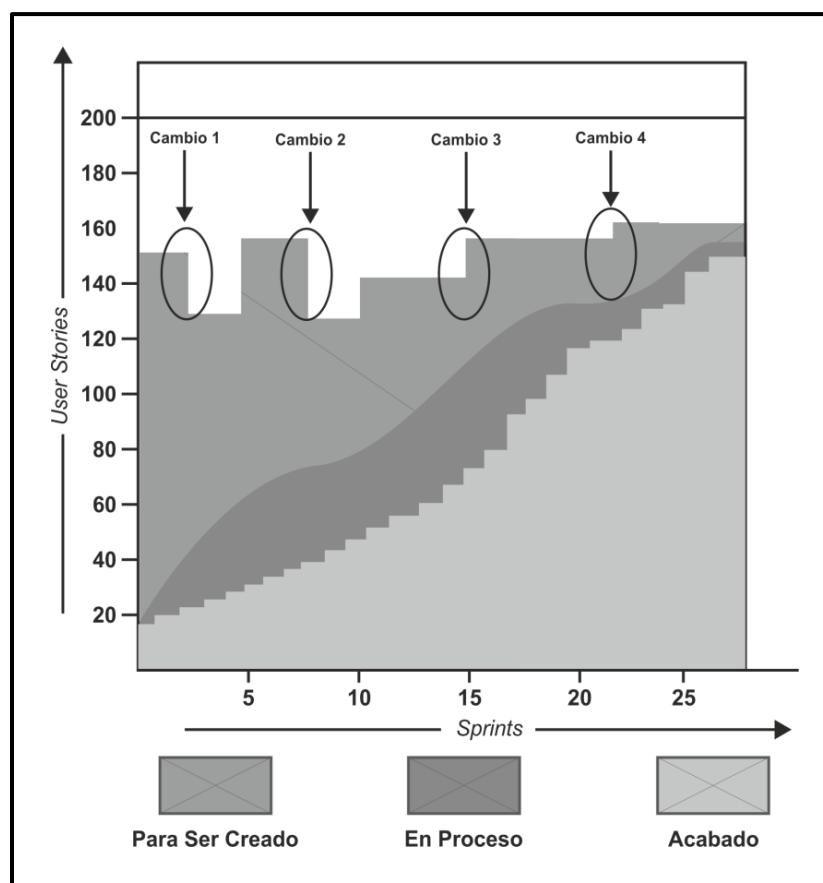


Figura 4-5: Ejemplo de *Cumulative Flow Diagram (CFD)*

4.7 Confirmar la realización de beneficios

A través de un proyecto, es importante verificar si los beneficios se están cumpliendo. Ya sea que los productos de un *project* Scrum son tangibles o intangibles, se requieren técnicas de verificación adecuadas para confirmar que el equipo está creando los entregables que lograrán los beneficios y el valor definido en el inicio del *project*.

4.7.1 Prototipos, simulaciones y demostraciones

La demostración de prototipos a los *customers* y simular su funcionalidad es una técnica común para confirmar el valor.

A menudo, después de usar las características o de demostrarlas a los *customers*, ellos pueden determinar con mayor claridad si las características son suficientes y adecuadas para sus necesidades. Pueden darse cuenta de la necesidad de características adicionales, o pueden decidir modificar los requisitos de características previamente definidas. En el desarrollo de productos, esta experiencia del *Customer* ha llegado a ser conocida como *IKIWISI* (*I'll Know It When I See It*)- *Lo sabré cuando lo vea*.

A través de demostraciones o el acceso a las iteraciones tempranas, *Customers* pueden también evaluar en qué medida el equipo ha sabido interpretar sus necesidades y cumplido con sus expectativas.

4.8 Resumen de responsabilidades

4

Función	Las responsabilidades
<i>Cuerpo de asesoramiento de Scrum</i>	<ul style="list-style-type: none"> • Establece las directrices generales y las métricas para evaluar el valor • Tiene la función de consultor y proporciona una guía para los <i>projects</i>, <i>programs</i> y <i>portfolios</i> tal como se requiere
<i>Portfolio Propietario del producto</i>	<ul style="list-style-type: none"> • Garantiza la entrega de valor para los <i>portfolios</i> • Crea <i>Justificación del negocio</i> para los <i>portfolios</i> • Proporciona una guía de valor de los <i>programs</i> dentro de los <i>portfolios</i> • Aprueba el <i>Justificación del negocio</i> de los <i>programs</i> dentro de un <i>portfolio</i>
<i>Portfolio Scrum Master</i>	<ul style="list-style-type: none"> • Garantiza que se logren los resultados deseados del <i>portfolio</i> • Realiza <i>Continuous Value Justification</i> para los <i>portfolios</i>
<i>Program Propietario del producto</i>	<ul style="list-style-type: none"> • Garantiza la entrega de valor de los <i>programs</i> • Crea el <i>Justificación del negocio</i> para los <i>programs</i> • Proporciona una guía de valor para los <i>projects</i> dentro de un <i>program</i> • Aprueba el <i>Justificación del negocio</i> de los proyectos dentro de un <i>program</i>
<i>Program Scrum Master</i>	<ul style="list-style-type: none"> • Asegura que los resultados esperados del <i>program</i> se comuniquen y se entiendan • Realiza <i>Continuous Value Justification</i> para los <i>programs</i>
<i>Stakeholder(s)</i>	<ul style="list-style-type: none"> • Ayuda a priorizar <i>user stories</i> y los requisitos de la <i>Prioritized Product Backlog</i> • Se comunica con el <i>Equipo Scrum</i> y confirma realización del valor al final de cada <i>Sprint</i>, <i>release</i>, y del <i>project</i>
<i>Propietario del producto</i>	<ul style="list-style-type: none"> • Garantiza la entrega de valor de los <i>projects</i> • Mantiene el <i>Justificación del negocio</i> de los <i>projects</i> • Confirma y comunica <i>project benefits</i> para los socios
<i>Scrum Master</i>	<ul style="list-style-type: none"> • Garantiza que los resultados esperados del <i>project</i> sean comunicados y comprendidos por el <i>Equipo Scrum</i> • Realiza <i>Continuous Value Justification</i> para los <i>projects</i>
<i>Equipo Scrum</i>	<ul style="list-style-type: none"> • Asegura que los entregables del <i>project</i> se han completado de acuerdo con los <i>acceptance criteria</i> acordados • Realiza <i>Continuous Value Justification</i> para los <i>projects</i>

Tabla 4-2: Resumen de las responsabilidades pertinentes a *Justificación del negocio*

4.9 Scrum vs gestión de proyectos tradicional

Los proyectos tradicionales hacen hincapié en una amplia planificación y la adhesión al plan de proyecto creado por el director del *project* inicial. Por lo general, los cambios se gestionan a través de un sistema formal de gestión del cambio y el valor se crea al final del *project*, cuando se entrega el producto final.

En los proyectos Scrum, no se realiza una amplia planificación antes de la ejecución del *project*. La planificación se realiza de manera sistemática antes de cada *Sprint*. Esto permite una respuesta rápida y eficaz a los cambios, lo que se traduce en menores costos y en última instancia, aumenta la rentabilidad y el *Return on Investment (ROI)*. Por otra parte, *Value-Driven Delivery* (sección 4.3) es un beneficio clave del marco de Scrum y proporciona mejor *Prioritization* de forma significativa y la realización más rápida del valor negocio. Debido a la naturaleza iterativa del desarrollo Scrum, siempre hay al menos una versión disponible del producto con las características mínimas negociables, *Minimum Marketable Features (MMF)*. Incluso si un proyecto se termina, por lo general hay algunos beneficios o plusvalías creadas antes de la terminación.

5. CALIDAD (*QUALITY*)

5.1 Introducción

El propósito de este capítulo es definir la calidad (*quality*) en lo que respecta a los *projects* y presentar el enfoque de Scrum para alcanzar los niveles de calidad requeridos.

Quality, tal como se define en *Una guía para el conocimiento de Scrum (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolios programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se les entregarán a los *socios*
- *Projects* de cualquier tamaño y complejidad

El término "producto" (*product*) en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria-desde pequeños proyectos o equipos con tan sólo seis miembros por equipo, hasta proyectos grandes y complejos que cuentan con cientos de miembros por equipo.

Este capítulo está dividido en las siguientes secciones:

5.2 Guía de los roles—En esta sección se proporciona orientación sobre qué secciones son relevantes para cada rol de Scrum: *Propietario del producto*, *Scrum Master*, y *Equipo Scrum*.

5.3 Definición de Calidad (*Quality*)—En esta sección se proporciona la definición Scrum de *quality*, con una clara distinción de alcance, y describe la relación entre la calidad y el valor del negocio.

5.4 Acceptance Criteria y Prioritized Product Backlog—Esta sección hace hincapié en la importancia de *Acceptance Criteria*, *Prioritized Product Backlog* y la relación entre estos dos. También se explica la definición de *Done* en Scrum.

5.5 Quality Management en Scrum—Esta sección proporciona detalles sobre *Quality Planning*, *Quality Control*, y *Quality Assurance* en el marco de Scrum.

5.6 Resumen de las responsabilidades—En esta sección se describen las responsabilidades pertinentes a la calidad para cada persona o papel en un *project*.

5.7 Scrum vs gestión de proyecto tradicional—Esta sección destaca los beneficios de *Quality Management* en el método Scrum con respecto a los modelos tradicionales de gestión de *projects*.

5.2 Guía de los roles

1. *Propietario del producto*— Es importante para cualquier persona que asume el papel de *Propietario del producto* en proyectos Scrum que lea este capítulo en su totalidad.
2. *Scrum Master*—El *Scrum Master* también debe estar familiarizado con todo este capítulo con enfoque principal en las secciones 5.3, 5.4, 5.5.3 y 5.6.
3. *Equipo Scrum*—El *Equipo Scrum* debe centrarse principalmente en las secciones 5.3, 5.4, y 5.6.

5.3 Definición de Calidad (*Quality*)

Hay muchas maneras de definir la calidad (*quality*).

En Scrum, la calidad se define como la capacidad del *product* o *products* entregables completados para cumplir los *Acceptance Criteria* y alcanzar el valor de negocio que espera el *customer*.

Para asegurar que un *project* cumpla con los requisitos de calidad, Scrum adopta un enfoque de *Continuous Improvement* en el que el equipo aprende de sus experiencias y del compromiso de los *socios*. Esto ayuda a mantener al día el *Prioritized Product Backlog* con los cambios en los requisitos. El *Prioritized Product Backlog* no está completo hasta el cierre o la terminación del *project*. Cualquier cambio en los requisitos refleja los cambios en el entorno empresarial interno y externo y permiten que el equipo funcione continuamente y se adapte a alcanzar esos requisitos. Scrum requiere que el trabajo se realize en incrementos durante los *Sprints*, esto hace que los errores o defectos sean detectados durante las pruebas de calidad repetitivas, y no cuando el producto final o servicio esté casi terminado. Por otra parte, las tareas importantes relacionadas con la calidad (por ejemplo, el desarrollo, prueba y documentación) se completan como parte del mismo *Sprint* por el mismo equipo - esto asegura que la calidad sea inherente a cualquier entregable *Done* creado como parte de un *Sprint*. Por lo tanto, *Continuous Improvement* con pruebas repetitivas optimiza la probabilidad de alcanzar los niveles esperados de *quality* en un proyecto Scrum. Las discusiones constantes entre el equipo principal de Scrum y los *Stakeholders* (incluyendo los *customers* y los *users*) con incrementos reales del *product* que se entregan al final de cada *Sprint*, aseguran que la brecha entre las expectativas de los *customers* sobre el *project* y los entregables producidos se reduzca constantemente.

5.3.1 Calidad y Alcance (*Qualiaty and Scope*)

El ámbito de aplicación y requisitos de *quality* para un *project* se determinan tomando en cuenta varios factores, como los siguientes:

- La necesidad del negocio que el *project* cumplirá
- La capacidad y la buena disposición de la organización para cumplir con las necesidades del negocio
- Las necesidades futuras y actuales de la audiencia

El alcance de un *project* es la suma total de todos los incrementos del *product* y el trabajo necesario para desarrollar el producto final. Calidad (*quality*), es la capacidad de las entregas para cumplir con los requisitos de calidad del producto y satisfacer las necesidades del *customer*. En Scrum, el alcance (*scope*) y la calidad del proyecto son capturados en el *Prioritized Product Backlog* y el alcance de cada Sprint está determinado por la refinación de los grandes *Prioritized Product Backlog Items (PBIs)* en un conjunto de pequeños pero detallados *User Stories* que pueden ser planeados, desarrollados y verificados dentro de un *Sprint*.

El *Prioritized Product Backlog* es continuamente cuidado por el *Propietario del producto*. El *Propietario del producto* se asegura de que cualquier *User Story* que el Equipo Scrum haga en un *Sprint* sea refinado antes de comenzar el *Sprint*. En general, al solucionar los problemas del *customer*, los requisitos más valiosos se priorizan como alto y los restantes reciben una prioridad menor. Los *User Stories* de menos importancia se desarrollan en posteriores *Sprints*, o se pueden eliminar por completo de acuerdo con los requisitos del *customer*. Durante la ejecución del *Sprint*, el *Propietario del producto*, *customer*, y el *Equipo Scrum* pueden discutir la lista de características del *product* para cumplir con las necesidades cambiantes de los clientes.

5.3.2 Calidad y Valor empresarial

La calidad y el valor empresarial están estrechamente vinculados. Por lo tanto, es fundamental entender la calidad y el alcance de un proyecto con el fin de asignar correctamente los resultados y beneficios del *project*, y el *product* debe tener un buen alcance con el fin de ofrecer valor empresarial. Para determinar el valor empresarial de un *product*, es importante entender la necesidad del negocio que impulsa los requisitos del *product*. Por lo tanto, la necesidad de la empresa determina el *product* requerido, y el *product*, a su vez, proporciona el valor empresarial esperado.

Quality es una variable compleja. Un aumento en el *scope*, sin incremento en el tiempo ni los recursos, tiende a reducir la calidad. Del mismo modo, una reducción en el tiempo o los recursos sin disminuir el *scope* generalmente resulta en una disminución de *quality*. Scrum cree en el “*sustainable pace*” (*ritmo sostenible*) de la obra, lo que permite mejorar la calidad a medida que transcurre el tiempo.

El *Cuerpo de asesoramiento de Scrum* puede definir los requisitos mínimos de *quality* y las normas que se deben cumplir en todos los *projects* de la organización. Estas normas deben ser cumplidas por todos los *Equipos Scrum* en la empresa.

5.4 Acceptance Criteria y Prioritized Product Backlog

El *Prioritized Product Backlog* es un documento de requisitos individuales que define el alcance (*scope*) del *project*, proporcionando una lista de prioridades de las características del *product* o servicio a ser entregado por el *project*. Las características necesarias se describen en forma de *User Stories*. Los *User Stories* son requisitos específicos señalados por varios *Socios* que se relacionan con el *product* o servicio propuesto. Cada *User Story* estará asociado con los *User Story Acceptance Criteria* (también conocidos como *Acceptance Criteria*), que son los componentes objetivos por los cuales se juzga la funcionalidad de un *User Story*. Los *Acceptance Criteria* son desarrollados por el *Propietario del producto*, de acuerdo a su conocimiento experto del requisito del *customer*. El *Propietario del producto*, entonces les comunica los *User Stories* en el *Prioritized Product Backlog* a los miembros del *Equipo Scrum* y solicita su acuerdo. Los *Acceptance Criteria* deben describir explícitamente las condiciones que los *User Stories* deben satisfacer. Estos criterios claramente definidos son cruciales para la entrega oportuna y eficaz de las funciones mencionadas en los *User Stories*, que en última instancia determinan el éxito del *project*.

Al final de cada *Sprint*, el *Propietario del producto* utiliza estos criterios para verificar los entregables completados, y puede aceptar o rechazar los entregables y sus *User Stories*. Si las prestaciones son aceptadas por el *Propietario del producto*, entonces los *User Stories* se consideran como *Done*. Una definición clara de *Done* es fundamental porque ayuda a clarificar los requisitos y permite que el equipo se adhiera a las normas de *quality*. También ayuda al equipo a pensar desde la perspectiva del usuario cuando se trabaja con *User Stories*.

Los *User Stories* correspondientes como *Rejected Deliverables* se añaden de nuevo al *Prioritized Product Backlog* durante el proceso de *Mantenimiento de la lista priorizada de pendientes del producto*, que se completará en los siguientes *Sprints*. El rechazo de unos entregables individuales y sus correspondientes *User Stories* no es un rechazo del *product* final o del incremento del producto. El incremento del *product* o el *product* podría ser potencialmente entregable incluso si algunos *User Stories* son rechazados.

La figura 5-1 ilustra el concepto de *Acceptance Criteria*, junto con el flujo de incremento del *product*.

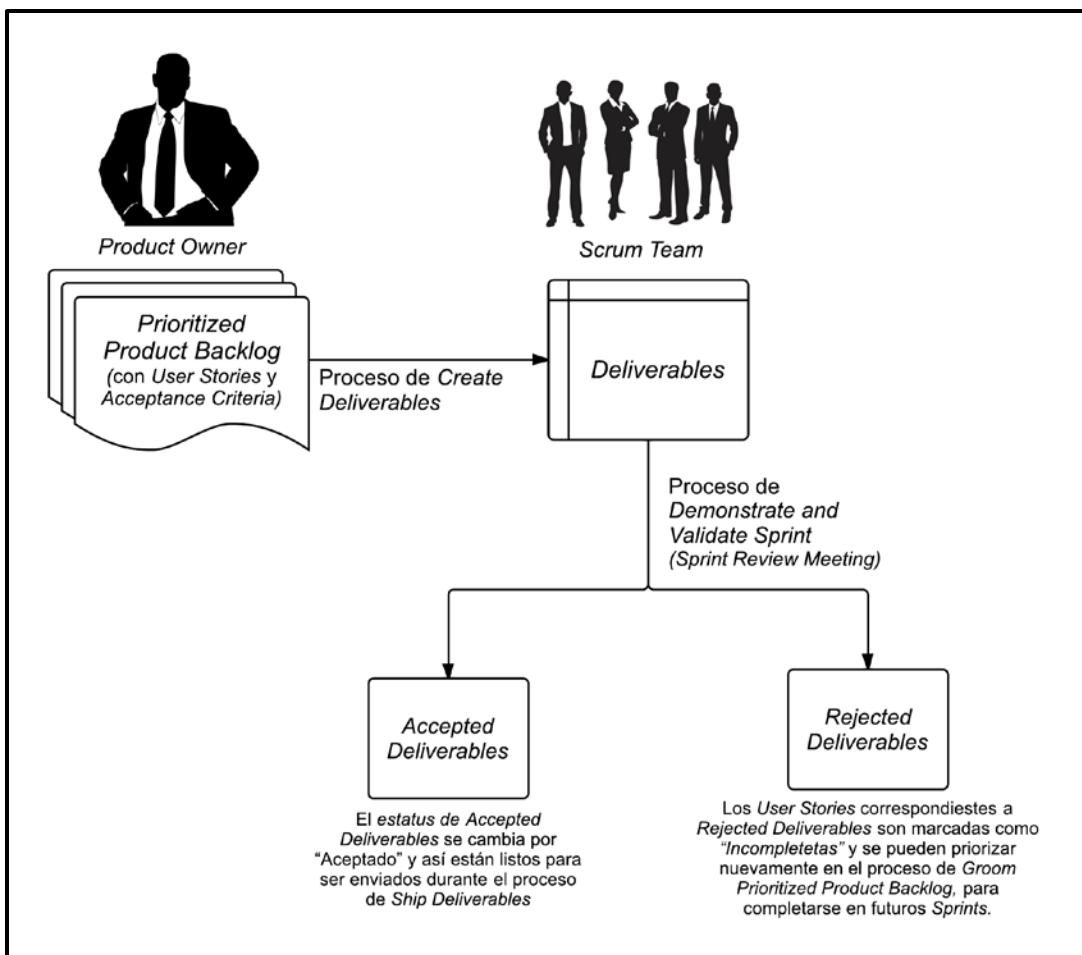


Figura 5-1: Diagrama de Flujo del Incremento del Proyecto

5.4.1 Escribiendo *Acceptance Criteria*

Los *Acceptance Criteria* son específicos para cada *User Story* y no son sustitutos de la lista de requisitos.

Ejemplo:

Personaje: Janine es una mujer profesional de 36 años, es casada y tiene tres hijos. Ella es una mujer ocupada, de éxito que equilibra su vida profesional y personal. Se siente cómoda con la tecnología y le gusta adoptar los productos y servicios innovadores. Siempre está conectada al internet a través de múltiples dispositivos y regularmente hace compras en portales de comercio electrónico.

User Story: "Como alguien quien hace las compras de comida en línea", dice Janine, "debería tener la opción de guardar y ver mi orden de cualquiera de mis dispositivos, así poder completar el proceso de pedido a mi conveniencia."

Acceptance Criteria:

- Todos los pedidos en curso deben guardarse cada 5 segundos en la cuenta del usuario como una orden de draft
- Nuevas órdenes draft deben aparecer como notificaciones o cualquier dispositivo que el usuario quiera usar

Es importante para un *Propietario del producto* señalar que los *User Stories* que cumplen con la mayoría de los *Acceptance Criteria*, pero no todos, no pueden ser aceptados como *Done*. Los *projects Scrum* operan de forma *Time-boxed Sprint*, con un *Sprint Backlog* dedicado para cada *Sprint*. A menudo, la última parte del trabajo puede ser la más complicada del *User Story* y puede tardar más de lo esperado. Si a los *User Stories* incompletos se les da crédito parcial como si estuvieran *Done*, y se les deja para el próximo *Sprint*, entonces el progreso del *Sprint* posterior podría ser interrumpido. Por lo tanto, el estado de *Done* es blanco y negro. En otras palabras, un *User Story* está hecho o no.

5.4.2 *Minimum Acceptance Criteria*

Una unidad de negocio de alto nivel puede anunciar un *Acceptance Criteria* mínimo y obligatorio, lo cual luego se puede convertir en parte de los *Acceptance Criteria* para cualquier *User Story* de esa unidad de negocio. En ese caso, cualquier funcionalidad definida por la unidad de la empresa debe satisfacer estos *Acceptance Criteria* mínimos, si ha de ser aceptada por el *Propietario del producto* respectivo. La introducción de estos *Acceptance Criteria* pueden dar lugar a una serie de *Acceptance Criteria* para el *portfolio, program* y *project* (véase la Figura 5-2). Las normas de *quality*, directrices y plantillas para todo un *portfolio* son establecidas por el *Portfolio Propietario del producto*, mientras que los *Acceptance Criteria* mínimos para los *programs* son establecidos por el *Program Propietario del producto*. Así, los *Acceptance Criteria* para un *User Story* en un *project* incluirán implícitamente todos los *Acceptance Criteria* mínimos, según corresponda.

<i>Portfolio Propietario del producto</i>	<ul style="list-style-type: none"> • Establece los <i>Acceptance Criteria</i> mínimos para todo el <i>Portfolio</i>. • Repasa los entregables de los <i>Portfolios</i>
<i>Program Propietario del producto</i>	<ul style="list-style-type: none"> • Establece los <i>Acceptance Criteria</i> mínimos para la totalidad del <i>program</i>, lo que incluye los <i>Acceptance Criteria</i> del <i>Portfolio</i> • Repasa los entregables de los <i>programs</i>
<i>Propietario del producto</i>	<ul style="list-style-type: none"> • Establece los criterios mínimos de aceptación para el <i>project</i>, lo que incluye los <i>Acceptance Criteria</i> del <i>program</i> • Repasa los entregables de los <i>projects</i>

Figura 5-2: *Acceptance Criteria* en Cascada (*Waterfall*)

Una vez que los *Acceptance Criteria* mínimos se definen, tales criterios, entonces pueden ser documentados en el *Cuerpo de asesoramiento de Scrum*, algo que los *Equipos Scrum* pueden utilizar como referencia según sea necesario.

5.4.3 Definición de *Done* (Terminado)

Hay una diferencia fundamental entre *Done Criteria* y *Acceptance Criteria*. Mientras que los *Acceptance Criteria* son únicos para los *User Stories* individuales, *Done Criteria* es un conjunto de reglas que se aplican a todas los *User Stories* en un determinado *Sprint*. Los *Done Criteria* generales podrían incluir cualquiera de los siguientes:

- Fueron repasados por otros miembros del equipo
- Se completó la prueba de unidad del *User Story*
- Llevar a cabo las pruebas de *Quality Assurance*
- Finalización de toda la documentación relacionada con los *User Stories*
- Todos los *issues* están arreglados
- La demostración exitosa a los *Socios* y/o representantes de la empresa

5

Al igual que con los *Acceptance Criteria*, todas las condiciones de los *Done Criteria* se deben cumplir para que el *User Story* sea considerado como *Done*.

El *Equipo Scrum* debe utilizar una lista de verificación de los *Done Criteria* generales para garantizar que una tarea está terminada y el resultado cumple con la *Definición de Terminado - Definition of Done (DoD)*. Una clara definición de *Done* es fundamental, ya que ayuda a eliminar la ambigüedad y les permite a los equipo a que se adhieran a las normas de calidad requeridas. La definición de *Done* es típicamente determinada y documentada por el *Cuerpo de asesoramiento de Scrum*.

Los registros y datos necesarios para cumplir con los requisitos de documentación del *project* se pueden generar a medida que el equipo procede a través de *Sprints* y publicaciones.

La inclusión de actividades tales como las reuniones de revisión y la escritura de documentos de diseño pueden ayudar a asegurar el cumplimiento de las normas de calidad interna y externa. Los principios básicos de Scrum como iteraciones cortas, la construcción gradual, la participación del cliente, adaptación a las nuevas necesidades, y el constante ajuste de tiempo y costo en el *project* son aplicables.

5.4.4 Aceptación o rechazo de elementos de *Prioritized Product Backlog*

Hacia el final de cualquier iteración, la unidad de negocio correspondiente y los *Socios* participan en un *Sprint Review Meeting* en el que el incremento del *product* se le demuestre al *Propietario del producto*, patrocinador (*sponsor*), *customer*, y a los usuarios. Si bien la opinión de todos los *Socios* se toma en cuenta, sólo el *Propietario del producto* tiene el poder de aceptar o rechazar un *User Story* en particular, de acuerdo al acordado *Acceptance Criteria*. Por lo tanto, el papel de los *Acceptance Criteria* en el mantenimiento de *quality* es crítico y debe ser claramente entendido por el equipo. Es la responsabilidad del *Scrum Master* asegurar que los *Acceptance Criteria* para un *User Story* no se modifiquen por el *Product Owner* durante un *Sprint*. Los *User Stories* casi por terminar son rechazados por no estar terminados y se trasladan de nuevo al *Prioritized Product Backlog*.

5.5 Quality Management en Scrum

El *customer* es el *Stakeholder* más importante para cualquier *project*. Por lo tanto, es importante entender las necesidades y requerimientos de los *customers*. La voz del cliente, *Voice of the Customer (VOC)*, se puede denominar como los requisitos explícitos e implícitos del client, que deben ser entendidos antes del diseño de un *product* o servicio. En general, en un entorno de Scrum, el *Propietario del producto* se centra en *Business Requirements* y objetivos, que en conjunto representan la voz del *customer*. El *Propietario del producto* puede beneficiarse mucho de la orientación que ofrece el *Cuerpo de asesoramiento de Scrum* (ya sea a través de documentos o normas de *quality*, o de expertos en *quality*). Estos especialistas deben trabajar con el *Propietario del producto* y el *customer* para garantizar el nivel adecuado de detalle y la información en los *User Stories*, ya que los *User Stories* son la base para el éxito de cualquier *project* de Scrum.

Cabe señalar que los *Socios* externos no participan directamente en el nivel del *Equipo Scrum* y, en cambio, interactúan principalmente con el *Propietario del producto*. Para cualquier *project* Scrum, el *customer* puede ser cualquiera de los siguientes:

- Interno (es decir, dentro de la misma organización)
- Externo (es decir, fuera de la organización)

Quality Management (Control de Calidad) en Scrum le permite a los *customers* tomar conciencia de los problemas en el *project* desde el principio y les ayuda a reconocer si un *project* les va a funcionar o no. En Scrum, *quality* es la satisfacción del *customer* y de un *product* que funciona, y no el cumplir necesariamente con métricas arbitrarias. Esta distinción se vuelve muy importante desde el punto de vista del *customer*, porque son ellos los que invierten tiempo y dinero en el *project*.

Quality Management en Scrum se facilita a través de tres actividades interrelacionadas:

1. *Quality planning*
2. *Quality control*
3. *Quality assurance*

5.5.1 Quality Planning

Uno de los principales rectores de Scrum es, en primer lugar, el desarrollo de la funcionalidad de más alta prioridad para el *customer*. Las características menos importantes se desarrollan en los siguientes *Sprints*, o se pueden dejar por completo de acuerdo con los requisitos del *customer*. Este enfoque le da al *Equipo Scrum* el tiempo necesario para centrarse en la calidad de la funcionalidad esencial. Un beneficio clave de *Quality Planning* es la reducción de *Technical Debt*. *Technical Debt (Deuda Técnica)*- también conocida como la *deuda de diseño* o *deuda de código* se refiere al trabajo que los equipos priorizan como inferior, omiten o no se completan a medida que trabajan hacia la creación de los entregables principales asociados con el *product* del *project*. *Technical Debt* se acumula y se debe pagar en el futuro.

Algunas causas de *Technical Debt* pueden incluir los siguientes:

- Una solución rápida y la construcción de entregables que no cumplen con los estándares de *quality*, seguridad y objetivos de arquitectura a largo plazo, etc.
- Prueba inadecuada o incompleta
- Documentación incorrecta o incompleta
- La falta de coordinación entre los diferentes miembros del equipo, o si diferentes *Equipos Scrum* comienzan a trabajar de manera aislada, con menos énfasis en la integración final de los componentes necesarios para realizar un *project* o *program* exitoso
- Pobre intercambio de conocimiento del negocio y el conocimiento del proceso entre los *Socios* y equipos de *project*
- Demasiado énfasis en los objetivos del *project* a corto plazo en lugar de los objetivos a largo plazo de la empresa. Este descuido puede resultar en mala calidad de *Working Deliverables* que incurren alto mantenimiento y costos significativos de actualización.

En los *projects Scrum*, cualquier *Technical Debt* no llegará más allá de un *Sprint*, porque debe haber *Acceptance and Done Criteria* que estén claramente definidos. La funcionalidad debe satisfacer estos criterios para ser considerado como *Done*. A medida que se arregla el *Prioritized Product Backlog* y los *User Stories* son priorizados, el equipo crea *Working Deliverables* con regularidad, lo que impide la acumulación de *Technical Debt*. El *Cuerpo de asesoramiento de Scrum* también puede incluir documentación y definición de los procesos que ayuden en la disminución de *Technical Debt*.

Para mantener una cantidad mínima de *Technical Debt*, es importante definir el *product* que se requiere de un *Sprint* y el proyecto junto a los *Acceptance Criteria*, cualquier método de desarrollo que se debe seguir y las responsabilidades claves de los miembros del *Equipo Scrum* en lo que se refiere a la calidad. La definición de *Acceptance Criteria* es una parte importante de *Quality Planning* y permite que *Quality Control* sea eficaz durante el *project*.

Technical debt es un reto muy grande con algunas técnicas de gestión de proyectos tradicionales, donde el desarrollo, la pruebas, la documentación, etc. se realizan secuencialmente y, a menudo por diferentes personas, sin que una persona sea responsable de ningún *Working Deliverables*. Como resultado, *Technical Debt* se acumula, lo que resulta en mayor mantenimiento, integración y costo de lanzamiento del *product* en la etapa final de la entrega de un *project*. Además, el costo de los cambios es muy alto en tales circunstancias ya que muchos problemas salen a la luz durante las etapas posteriores del *project*. El marco de Scrum evita los *issues* relacionados con *Technical Debt* al asegurar que los *Entregables Hechos (Done Deliverables)* con *Acceptance Criteria* se definan como parte del *Sprint Backlog* y que las tareas clave, incluyendo el desarrollo, las pruebas y la documentación se realicen como parte del mismo *Sprint*, y por el mismo *Equipo Scrum*.

5.5.1.1 Integración continua y *Sustainable Pace*

El mantenimiento de un *sustainable pace* es uno de los principios más importantes de Scrum. *Sustainable pace* resulta en una mayor satisfacción de los empleados, estabilidad y aumento de la precisión de la estimación, todo lo cual en última instancia conduce a una mayor satisfacción del *customer*. Para desarrollar un *product* verdaderamente de alta calidad y mantener un ambiente de trabajo saludable, es importante llevar a cabo actividades de tipo de integración con regularidad, en lugar de retrasar el trabajo de integración hasta el final en este tipo de circunstancias. Para proporcionar valor en intervalos frecuentes, el equipo debe desarrollar funcionalidades de forma continua, hacer pruebas, e integrar las funcionalidades de cada elemento de *Prioritized Product Backlog Item* (*PBI*) en cada *Sprint* con el uso de técnicas como la integración continua y pruebas automáticas del *product*. También es importante, desde el punto de vista del equipo, asegurar que el esfuerzo realizado en el *Sprint* actual sea similar al esfuerzo invertido en el *Sprint* anterior para así mantener un ritmo constante durante todos los *Sprints* del *project*. Esto ayuda al equipo a evitar fases de intensos períodos de trabajo, asegurando que siempre estén en condiciones de poner el nivel de esfuerzo requerido para llevar a cabo el trabajo que hay que hacer.

5.5.2 *Quality Control* y *Quality Assurance*

Quality control se refiere a la ejecución de las actividades de calidad previstas por el *Equipo Scrum* en el proceso de creación de Entregables que están potencialmente listos para la entrega. También incluye el aprendizaje de cada conjunto de actividades realizado con el fin de lograr *Continuous Improvement*. Dentro del equipo multi-funcional, es importante contar con las habilidades necesarias para llevar a cabo actividades de *Quality Control*. Durante la reunión de *Sprint Retrospect*, los miembros del equipo discuten las lecciones aprendidas. Estas lecciones actúan como entradas en *Continuous Improvement* y contribuyen a la constante mejora de *Quality Control*.

Se requiere de la calidad no sólo en *products*, sino también en los procesos. *Quality Assurance* se refiere a la evaluación de los procesos y normas que rigen *Quality Management* en un *project* para asegurarse de que siguen siendo actividades relevantes. *Quality assurance* se lleva a cabo como parte de la obra. De hecho, *quality assurance* es un factor importante de la definición de *Done*. La entrega no está completa si *quality assurance* no se ha llevado a cabo. A menudo, *quality assurance* se demuestra durante el *Sprint Review Meeting*.

Propietario del producto de los respectivos *projects*, *programs* y *portfolios* pueden monitorear y evaluar las actividades de *quality assurance* para garantizar que cada equipo siga estando de acuerdo y cumpla con los estándares de *quality* que se han establecido. *Quality assurance* puede ser abordado durante los ensayos finales del *product*, un lanzamiento, o un *Sprint*. Una comparación de la cantidad de *issues* que se han encontrado frente a la cantidad de *User Stories* completados se puede hacer. Los componentes de los *products* que tienen defectos pueden ser incorporados como elementos en el *Prioritized Product Backlog Items* (*PBIs*), los cuales pueden ser arreglados por cualquier equipo o por una persona en ciertos momentos durante el *Sprint*, dependiendo del número de defectos.

A veces, el *Cuerpo de asesoramiento de Scrum* puede definir los procesos y documentos que pueden aplicar los *Equipos Scrum* al hacer sus *projects* para asegurar que las normas de calidad uniformes sean seguidas en todos los *projects* dentro de la empresa.

5.5.3 Plan-Do-Check-Act (PDCA) Cycle

Plan-Do-Check-Act Cycle (Ciclo de Planear-Hacer-Revisar-Actuar) conocido como el Deming o Shewhart Cycle fue desarrollado por el Dr. W. Edwards Deming, considerado el padre de *quality control* moderna y por el Dr. Walter A. Shewhart. Los siguientes son algunos puntos importantes de la filosofía de Deming:

- Las directrices de gestión definen la calidad. Cuando la administración es capaz de proporcionar un entorno propicio y es capaz de motivar a sus empleados para mejorar la calidad de forma continua, cada empleado será capaz de hacer una contribución para un *product* de calidad superior. La "Teoría del Conocimiento Profundo" (Theory of Profound Knowledge) de Deming defiende lo que la administración debe hacer con el fin de crear un entorno en el que cada empleado puede lograr contribuciones significativas a la mejora de la calidad.

5

Deming modificó *Plan-Do-Check-Act* a *Plan-Do-Study-Act* (*PDSA*, *(Planear-Hacer-Estudiar-Actuar)*) porque sentía que el término "*Study*" (estudiar) enfatizaba el análisis en lugar de simplemente *inspection* (*inspección*), como lo implica el término "*Check*" (verificar).

Tanto Scrum y el Ciclo Deming/Shewhart/PDCA son métodos iterativos que se centran en *continuous improvement*. La figura 5-3 ilustra las etapas del ciclo PDCA y su correlación con los diversos procesos de Scrum.

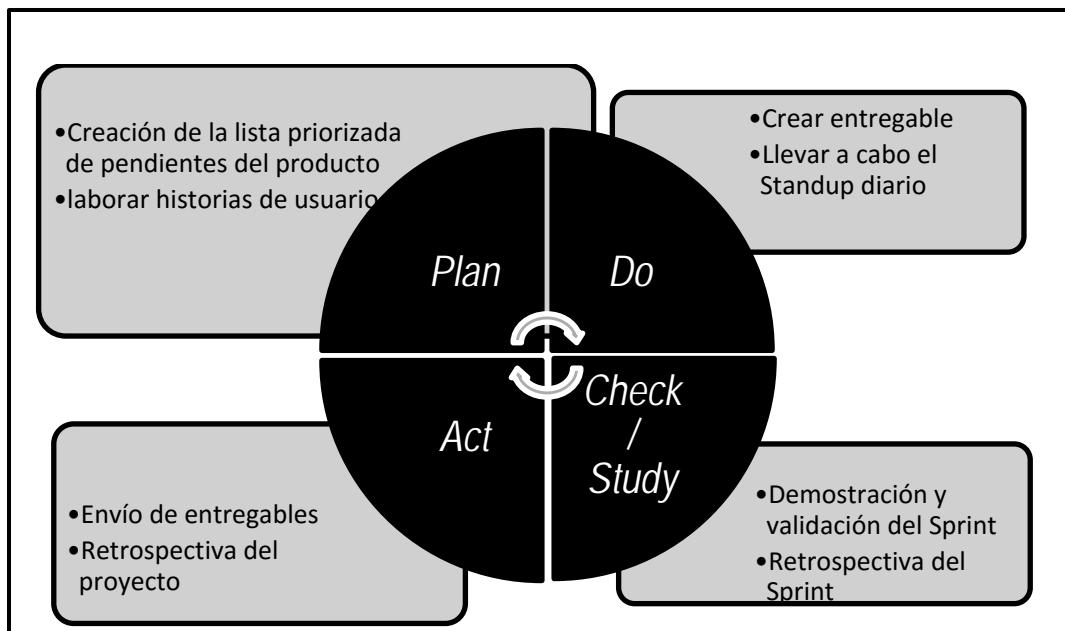


Figura 5-3: PDCA Cycle en Scrum

5.6 Resumen de responsabilidades

Función	Las responsabilidades
Cuerpo de asesoramiento de Scrum	<ul style="list-style-type: none"> • Proporciona definición de <i>Done</i> • Proporciona un marco y una guía para el desarrollo de <i>Acceptance Criteria</i> • Define la gama de herramientas que se puede utilizar por el <i>Equipo Scrum</i> para desarrollar y verificar el <i>product</i>
Portfolio Propietario del producto	<ul style="list-style-type: none"> • Establece un mínimo <i>Acceptance Criteria</i> para todos los <i>portfolios</i> • Repasa los entregables del <i>portfolio</i>
Portfolio Scrum Master	<ul style="list-style-type: none"> • Asegura que un <i>Sustainable Pace</i> se mantenga en el que la atención se centra en la calidad de las características y no estrictamente en la velocidad
Program Propietario del producto	<ul style="list-style-type: none"> • Establece los mínimos <i>Acceptance Criteria</i> para todo el programa • Revisa los entregables del <i>program</i>
Program Scrum Master	<ul style="list-style-type: none"> • Asegura que un <i>Sustainable Pace</i> se mantenga en el que la atención se centra en la calidad de las características y no estrictamente en la velocidad
Stakeholder(s)	<ul style="list-style-type: none"> • Revisa y acepta los entregables y el producto final
Propietario del producto	<ul style="list-style-type: none"> • Define los <i>business requirements</i> para el <i>product</i> y define los requisitos con claridad del <i>Prioritized Product Backlog</i> • Evalúa la viabilidad y asegura que los entregables cumplan con los requisitos de <i>quality</i> • Establece los <i>Acceptance Criteria</i> mínimos para todo el <i>project</i>, incluyendo los <i>Acceptance Criteria</i> del <i>program</i> respectivo • Facilita la creación de <i>Acceptance Criteria</i> para los <i>User Stories</i> • Comenta y valida sobre los Entregables durante el <i>Demostración y validación del Sprint</i>
Scrum Master	<ul style="list-style-type: none"> • Facilita una mentalidad de "primero el equipo", cuando se trata de <i>quality</i> • Elimina obstáculos ambientales que puedan afectar la calidad de los resultados y los procesos • Asegura que un <i>sustainable pace</i> se mantenga en el que la atención se centra en la calidad de las características y no estrictamente en la velocidad • Asegura que los procesos de Scrum se sigan correctamente por todos los miembros del equipo, incluyendo el <i>Propietario del producto</i>
Equipo Scrum	<ul style="list-style-type: none"> • Desarrolla y mantiene todas los entregables durante los <i>Sprints</i> hasta que sean entregados a los usuarios finales • Practica y alienta una buena comunicación para que se aclaren los requisitos y se entiendan por completo • Comparte el conocimiento para asegurar que los miembros del equipo se familiaricen con todo el conjunto de funciones y, por lo tanto, se beneficien de la experiencia de otros • Hace cambios apropiados rápidamente de los Entregables

Tabla 5-1: Resumen de las responsabilidades pertinentes a la Calidad (*Quality*)

5.7 Scrum vs gestión de proyectos tradicional

Aunque hay similitudes en Scrum y los métodos tradicionales de gestión de *projects* en relación con la definición de "calidad", "quality", (es decir, la capacidad del *product* para cumplir con los *Acceptance Criteria* acordados y lograr el valor empresarial que espera el *customer*), existen diferencias en términos de cómo los enfoques abordan la aplicación y el logro de los niveles de *quality* exigidos.

En los métodos tradicionales de gestión de *projects*, los usuarios aclaran sus expectativas; el director del *project* define las expectativas en términos medibles y acuerdo de ganancias de los usuarios. Después de una planificación detallada, el equipo del *project* desarrolla el *product* durante un período de tiempo acordado. Si alguno de los criterios acordados ha de ser cambiado, los cambios se suceden sólo a través de un sistema formal de gestión del cambio en el que se estima el impacto de los cambios y el director de *project* consigue la aprobación de todos los *socios* relevantes.

En Scrum, sin embargo, el *Propietario del producto* colabora con el *Equipo Scrum* y define los *Acceptance Criteria* para los *user stories* en relación con el *product* que debe ser entregado. El *Equipo Scrum*, entonces desarrolla el *product* de una serie de iteraciones cortas denominadas *Sprints*. El *Propietario del producto* puede realizar cambios en los requisitos para mantenerse al ritmo de las necesidades del usuario y estos cambios pueden ser abordados por el *Equipo Scrum* ya sea al concluir el *Sprint* actual, o al incluir los requisitos ajustados en el próximo *Sprint*, ya que cada *Sprint* es de muy corta duración (de una a seis semanas).

Una de las principales ventajas de Scrum es el énfasis en la creación de Entregables potencialmente listos para ser entregados al final de cada ciclo de *Sprint*, en lugar de esperar al final de todo el *project*. Así, el *Propietario del producto* y los *customers* constantemente inspeccionan, aprueban y aceptan Entregables después de cada *Sprint*. Incluso, si un *project* Scrum se terminó antes de tiempo, hay de todas formas algo de valor que fue creado antes de la terminación como resultado de los entregables creados en *Sprints* individuales.

6. Cambio

6.1 Introducción

Cada *project*, independientemente de su método o marco está expuesto a cambios. Es imperativo que los miembros del equipo del *project* entiendan que los procesos de desarrollo de Scrum están diseñados para aceptar el cambio. Las organizaciones deben tratar de maximizar los beneficios que se derivan de los cambios y disminuir los impactos negativos a través de los procesos de gestión de cambio diligente según los principios de Scrum.

El Cambio, tal como se define en *Una guía para el conocimiento de Scrum (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolios, programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se les entregará a los *shareholders*
- *Projects* de cualquier tamaño y complejidad

El término "producto" (*product*) en la *Guía SBOK™* puede referirse a un *product*, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier *project* en cualquier industria-desde *projects* o equipos pequeños con tan sólo seis miembros, hasta *projects* grandes y complejos con cientos de miembros por equipo.

Este capítulo está dividido en las siguientes secciones:

6.2 Guía de los roles—En esta sección se proporciona orientación sobre qué secciones son relevantes para cada uno de los roles principales de Scrum: *The Propietario del producto*, *Scrum Master*, y *Equipo Scrum*.

6.3 Descripción—Esta sección define el concepto de cambio, específicamente dentro del contexto de los procesos de Scrum. También ilustra la forma en la que *Change Requests* se maneja en los procesos de Scrum.

6.4 Cambio en Scrum—En esta sección se detalla la importancia de gestionar con eficacia el cambio en un *project* Scrum. También se habla de cómo la flexibilidad y la estabilidad se puede lograr a través del manejo adecuado de los *Change Requests* que surgen a lo largo de un *project*.

6.5 Integración del cambio—Esta sección detalla cómo los *Change Requests* son evaluados y aprobados (o rechazados) en la aplicación del marco de Scrum.

6.6 Cambio al *Program* y de *Portfolio*—En esta sección se describe el impacto de los cambios en los *programs* y *portfolios*.

6.7 Resumen de las responsabilidades—Esta sección define las responsabilidades de los miembros de un equipo de proyecto al gestionar cambios.

6.8 Scrum vs proyecto tradicional de gestión—Esta sección discute los beneficios de gestionar cambios utilizando Scrum en comparación a los modelos tradicionales de gestión de proyectos.

6.2 Guía de los roles

1. *Propietario del producto*—La responsabilidad de iniciar el cambio en un *project* recae principalmente en el *Propietario del producto*, por lo tanto, todo este capítulo es aplicable a este papel.
2. *Scrum Master*—El *Scrum Master* también debe estar familiarizado con todo este capítulo con enfoque principal en las secciones 6.3, 6.4, 6.5, y 6.7.
3. *Equipo Scrum*—El *Equipo Scrum* debe centrarse principalmente en las secciones 6.3, 6.4.2, 6.5 y 6.7.

6.3 Descripción

El cambio es inevitable en todos los *projects*. En el mundo hipercompetitivo de hoy, donde la tecnología, las condiciones del mercado, y los patrones de negocio están cambiando de forma continua, el cambio (*change*) es el único constante.

Un principio fundamental de Scrum es su reconocimiento de que a) los *socios* (por ejemplo, *customers* usuarios y patrocinadores) cambian de opinión acerca de lo que quieren y necesitan en todo el *project* (a veces denominado "*requirements churn*" a) y b) que es muy difícil, si no imposible, para los *socios* definir todos los requisitos durante la iniciación del *project*.

Los *projects* de desarrollo de Scrum le dan la bienvenida al cambio mediante el uso de pequeños ciclos de desarrollo que incorporan retroalimentación del *customer* de los entregables del *project* después de cada *Sprint*. Esto permite que el *customer* interactúe regularmente con los miembros del *Equipo Scrum*, vea los incrementos del *product* a medida que estén listos, y cambie los requisitos temprano en el ciclo de desarrollo. De esa forma, los equipos de gestión de *portfolio* o de *programs* pueden responder a los *Change Requests* pertenecientes a los *projects* Scrum aplicables a su nivel.

Scrum encarna un principio primordial del *Agile Manifesto* (Fowler y Highsmith, 2001): "*Responding to change over following a plan.*" (*Responder al cambio, en vez de seguir un plan*). La práctica de Scrum se basa en la aceptación del cambio y de convertirlo en una ventaja competitiva. Por lo tanto, es más importante ser flexible que seguir un plan estricto y predefinido. Esto significa que es esencial abordar la gestión de *projects* de una manera adaptativa que permita el cambio durante el desarrollo del *product* o de los ciclos rápidos de servicio.

Ser adaptable al cambio es una ventaja clave del marco de Scrum. Aunque Scrum funciona bien en todo tipo de *projects* y sectores, puede ser muy eficaz cuando el *product* o los requisitos del *project* no se identificaron o no están bien definidos desde el principio. Scrum también es eficaz cuando el mercado del

product es volátil, y/o cuando la atención se centra en hacer que el equipo sea lo suficientemente flexible para incorporar los requisitos cambiantes. Scrum es especialmente útil para *projects* complejos con mucha incertidumbre. La planificación y previsión a largo plazo suele ser ineficaz para este tipo de proyectos e implican grandes cantidades de riesgo. Scrum lleva al equipo a obtener resultados valiosos de negocio utilizando *Transparencia, inspection y adaptation*.

6.3.1 Unapproved and Approved Change Requests

La solicitud de cambio se presenta por lo general como *Change Requests*. *Change Requests* no son aprobados hasta que se obtiene una aprobación formal. El *Cuerpo de asesoramiento de Scrum* por lo general define el proceso de decisión y gestión de los cambios en la organización. En ausencia de un proceso formal, se recomienda que los pequeños cambios que no tienen un impacto significativo en el *project* sean aprobados directamente por el *Propietario del producto*. La tolerancia de estos pequeños cambios se podría definir a nivel organizacional o por el patrocinador de un *project* en particular. En la mayoría de los *projects*, el 90% de los *Change Requests* podrían clasificarse como pequeños cambios que deben ser aprobados por el *Propietario del producto*. Así que el *Propietario del producto* juega un papel muy importante en la gestión de los cambios en un *project* Scrum.

Los cambios que están más allá del nivel de tolerancia del *Propietario del producto* pueden necesitar la aprobación de socios que trabajan con el *Propietario del producto*.

A veces, si un cambio solicitado puede tener un impacto sustancial en el *project* u organización, la autorización de la dirección (por ejemplo, el Patrocinador Ejecutivo, *Portfolio Propietario del producto*, *Program Propietario del producto*, o Jefe *Propietario del producto*) puede ser necesaria.

Los *Change Requests* para el *project* se discuten y aprueban durante los procesos: *Desarrollo de épica(s)*, *Creación de la lista priorizada de pendientes del producto*, y *Mantenimiento de la lista priorizada de pendientes del producto*. Los *Approved Change Requests* entonces se priorizan junto con otros requisitos del *product* y sus respectivos *User Stories* y luego son incorporados en el *Prioritized Product Backlog*.

La figura 6-1 resume el proceso de aprobación de la modificación y la figura 6-2 explica cómo se actualiza el *Prioritized Product Backlog* con los cambios aprobados.

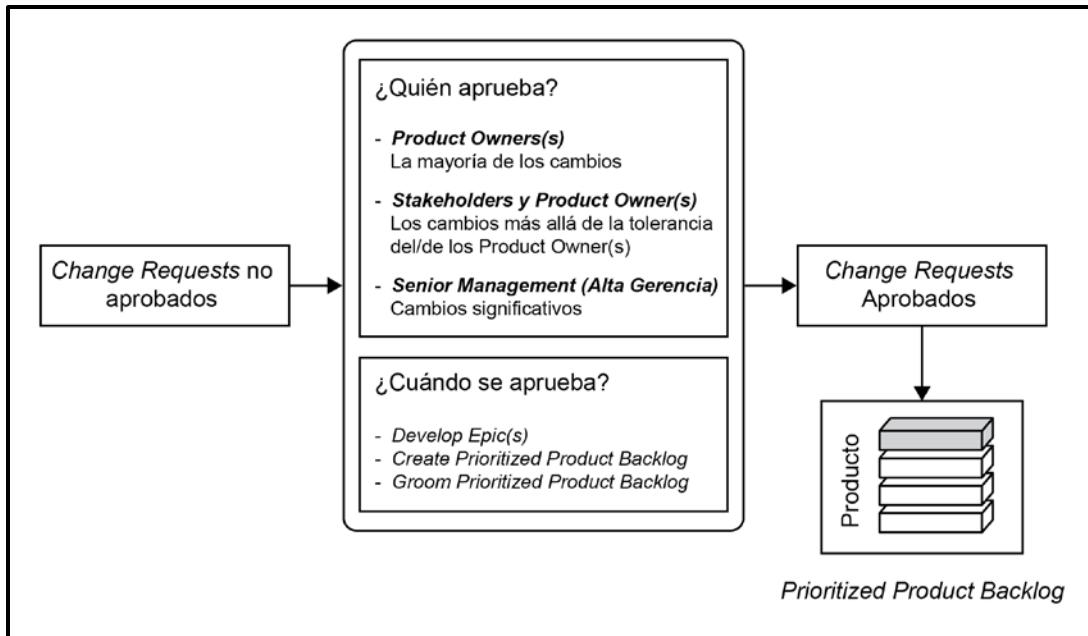
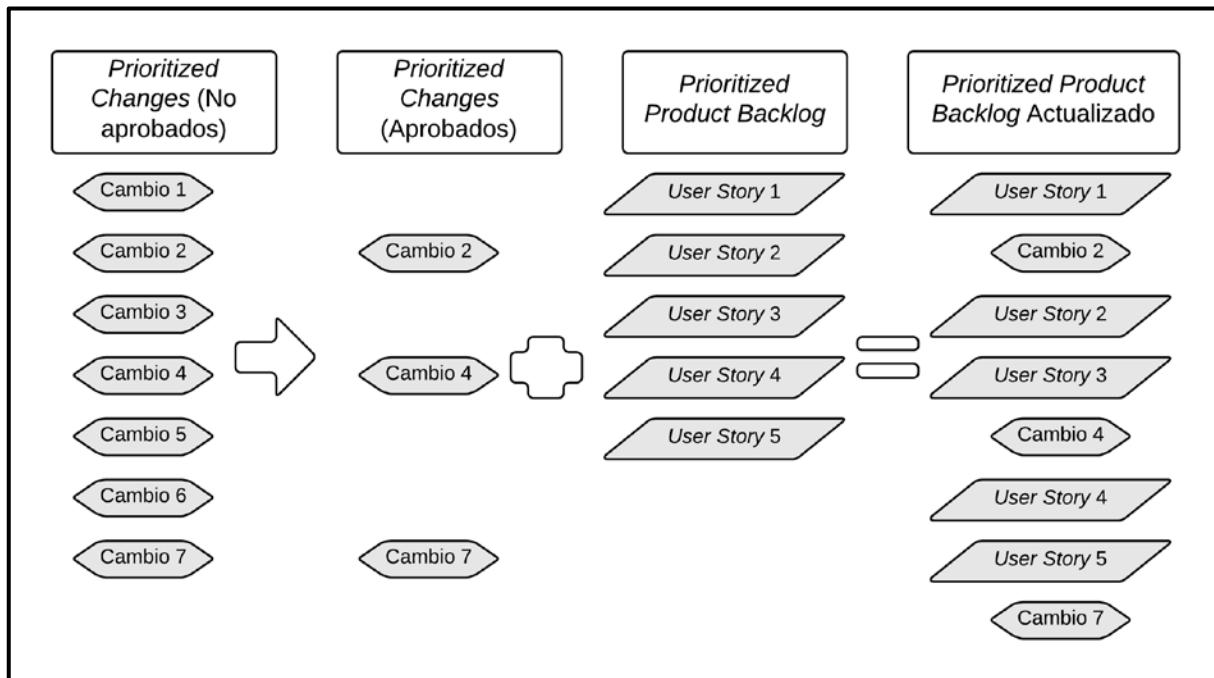


Figura 6-1: Ejemplo de Proceso de Cambio de Aprobación

Figura 6-2: Actualización de *Prioritized Product Backlog* con los cambios aprobados

6.4 Cambio en Scrum

6.4.1 Equilibrio entre flexibilidad y estabilidad

Scrum ayuda a las organizaciones a ser más flexibles y abiertas al cambio. Sin embargo, es importante entender que aunque el marco de Scrum hace hincapié en la flexibilidad, también es importante tener estabilidad durante todo el proceso de cambio. De la misma manera que la rigidez extrema es ineficaz, la flexibilidad extrema también es improductiva. La clave es encontrar el equilibrio adecuado entre la flexibilidad y la estabilidad ya que se necesita la estabilidad con el fin de realizar el trabajo. Por lo tanto, Scrum utiliza desarrollo iterativo y sus otras características y principios para lograr este equilibrio. Scrum mantiene la flexibilidad de que los *Change Requests* pueden ser creados y aprobados en cualquier momento durante el *project*. Sin embargo, consiguen prioridad cuando se crea o se actualiza el *Prioritized Product Backlog*. Al mismo tiempo, Scrum asegura que se mantenga la estabilidad al mantener el *Sprint Backlog* y al no permitir interferencia con el *Equipo Scrum* durante un *Sprint*.

En Scrum, todos los requisitos relacionados con el *Sprint* en curso se congelan durante el *Sprint*. Ningún cambio se introduce hasta que se termina el *Sprint*, a menos que se considere un cambio de ser lo suficientemente importante como para detener el *Sprint*. En el caso de un cambio urgente, el *Sprint* se termina y el equipo se reúne para planificar un nuevo *Sprint*. Así es como Scrum acepta cambios sin cambiar las fechas de lanzamiento.

6.4.2 El logro de la flexibilidad

Scrum facilita la flexibilidad a través de *Transparencia, inspection y adaptation* para lograr los resultados de negocio más valiosos. Scrum proporciona un mecanismo de adaptación para la gestión de *projects* en el que un cambio en los requisitos se puede acomodar sin afectar significativamente el progreso general del *project*. Es necesario adaptarse a las realidades de los negocios emergentes, como parte del ciclo de desarrollo. La flexibilidad en Scrum se logra a través de cinco características claves (véase la Figura 6-3): el desarrollo de productos iterativos, *Tiempo asignado*, equipos multi-funcionales, *Customer Value-based Prioritization* y la integración continua.

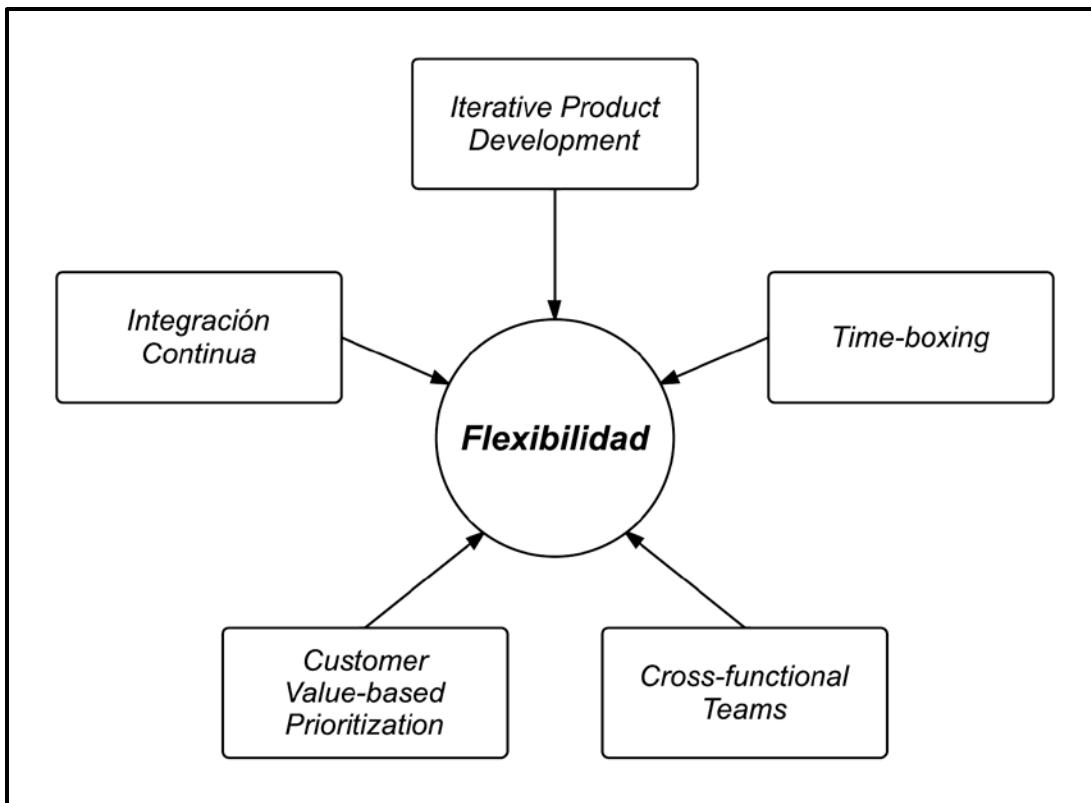


Figura 6-3: Características de Scrum para lograr flexibilidad

6.4.2.1 La flexibilidad a través del desarrollo iterativo del producto

Scrum sigue un enfoque iterativo e incremental de desarrollo de *products* y servicios, por lo que es posible la incorporación de cambios en cualquier paso en el proceso de desarrollo. A medida que se desarrolla el *product*, una solicitud de cambio para el *project* puede provenir de múltiples fuentes de la siguiente manera:

1. Socios

Project socios—Los *Socios* del *project*, en particular los patrocinadores, *customers*, y los usuarios podrán—presentar *Change Requests* en cualquier momento durante todo el *project*. *Change Requests* podrían ser debidos a cambios en las condiciones del mercado, dirección de la organización, *issues* legales o reglamentarias, o varias otras razones. Por otra parte, *socios* pueden presentar *Change Requests*, a medida que van revisando las entregas durante los procesos de *Demostración y validación del Sprint*, *Retrospectiva de Sprint*, o *Retrospectiva del proyecto*. Todos los *Change Request* se añaden al *Project Prioritized Product Backlog* (también denominado *Prioritized Product Backlog* o *Product Backlog*), una vez aprobado. La figura 6-4 muestra algunas de las razones por las que *socios* inician el proceso de solicitud de cambio.

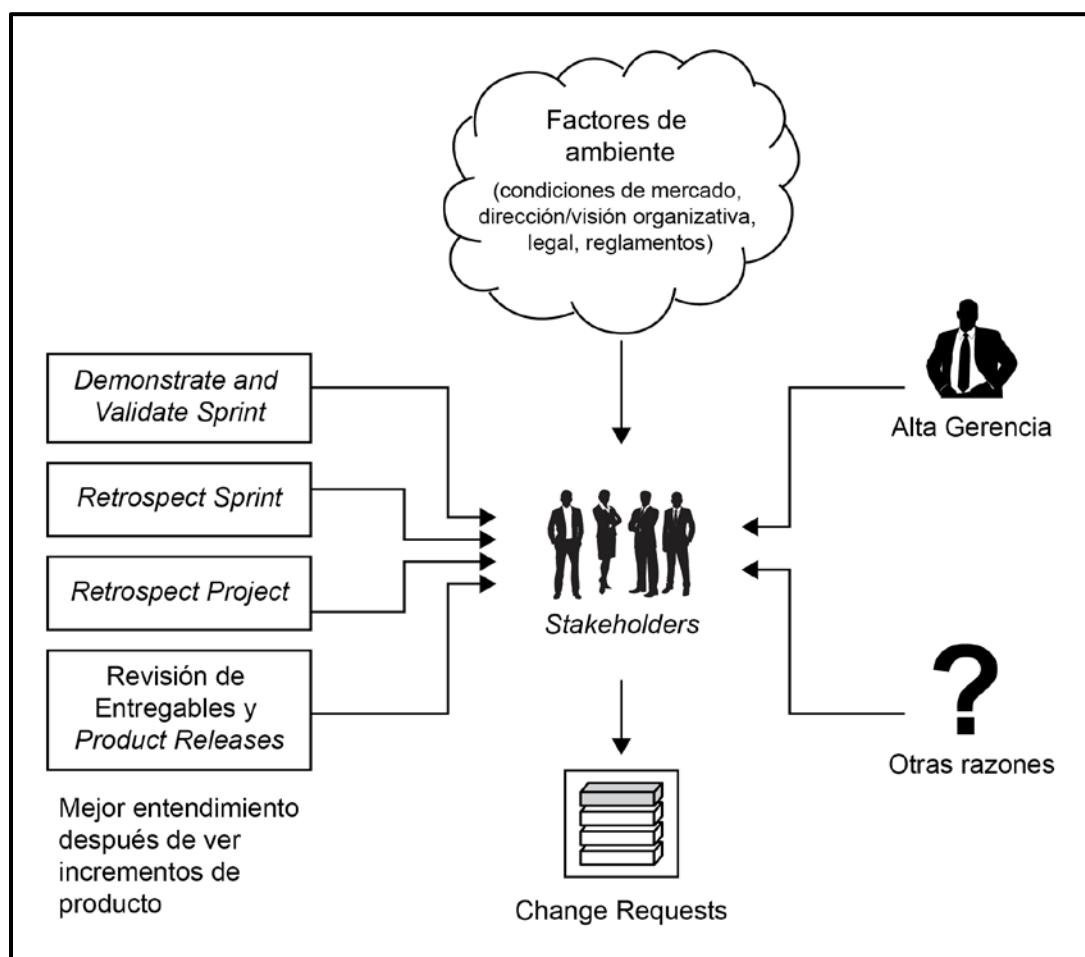


Figura 6-4: Motivación de los *socios* para la solicitud de cambios

2. Scrum Core Team

El *Scrum Core Team* (es decir, el *Propietario del producto*, *Scrum Master*, y *Equipo Scrum*) participa en la creación de los entregables del *product*. La interacción continua entre los miembros principales de un *Equipo Scrum* y otros (como otros *Equipos Scrum* del *project*, y los *socios* internos y externos del *project*) puede motivarlos a sugerir cambios o mejoras en el *product*, servicio, o cualquier otro aspecto del *project*. Normalmente estos cambios- al igual que otros- se incluyen en los *Change Requests*, y el *Propietario del producto* toma una decisión final acerca de que sugerencias de cambio por parte del *Equipo Scrum* o *Scrum Master* deben ser consideradas como *Change Requests* formales.

Puede haber a veces desafíos con la creación de ciertas entregas, lo que puede resultar en *Change Requests*. Por ejemplo, el equipo puede añadir o modificar una característica para mejorar el rendimiento del *product*. En la mayoría de los proyectos Scrum, las recomendaciones de cambios por parte del Scrum Core Team (Equipo Principal de Scrum) suceden durante el proceso de *Crear entregables*, o cuando participan en *Llevar a cabo el Standup diario Meetings* o *Retrospectiva de Sprint Meetings*. La figura 6-5 muestra algunas de las razones por las que el Equipo Principal de Scrum puede iniciar *Change Requests*.

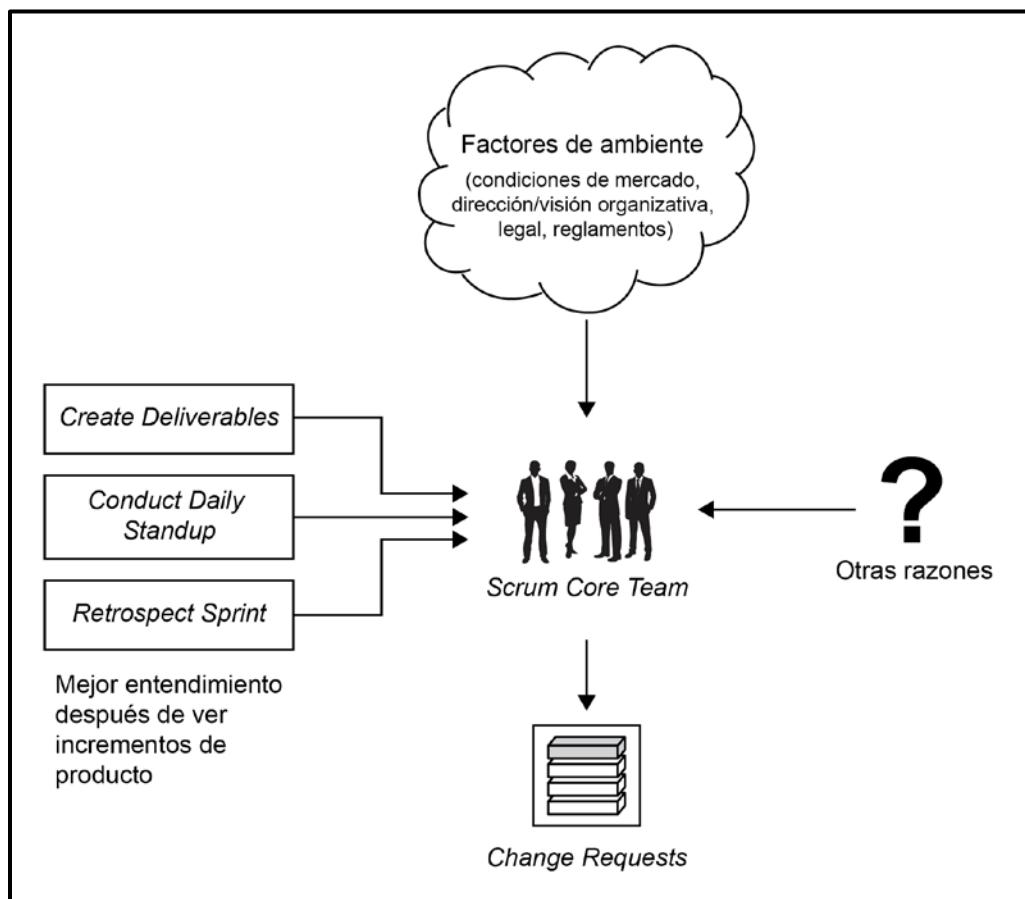


Figura 6-5: La motivación del Equipo Central/Principal de Scrum para solicitar cambios

3. Gerencia general

La gerencia general- incluyendo la gestión del *Portfolio* y el *program*- puede recomendar cambios que afectan al *project*. Esto puede ser debido a cambios estratégicos en la dirección de la empresa, a un entorno competitivo, a cambios de financiación relacionadas con *issues*, y así sucesivamente. Es importante tener en cuenta que estos cambios se añaden al *Prioritized Product Backlog* y tienen que pasar por el proceso normal de gestión del cambio. Si algunos de estos cambios son urgentes, cualquier *Sprint* afectado puede que necesite ser terminado (ver sección 6.6 para más detalles).

4. Cuerpo de asesoramiento de Scrum

El *Cuerpo de asesoramiento de Scrum* puede presentar *Change Request* que afectan a todos los *projects* debido a alguno de los siguientes ejemplos:

- Cambios en las regulaciones del gobierno (por ejemplo, la privacidad, las normas de seguridad, o una nueva legislación)
- Directivas corporativas de *quality*, seguridad, u otras iniciativas de organización que deben ponerse en práctica en toda la compañía
- Puntos de referencia o mejores prácticas para alcanzar a cierto nivel
- Lecciones aprendidas de *projects* anteriores que puedan ser implementadas por otros *Equipos Scrum*

El sello distintivo de Scrum es que es tolerante y se adapta a los cambios. Scrum no promueve determinar y establecer planes con mucha firmeza y anticipación ya que opera en la premisa de que el desarrollo del *project* es muy propenso al cambio y riesgo. El resultado es un alto grado de flexibilidad y tolerancia al cambio. El *project* es ejecutado y gestionado de forma incremental, por lo que suele ser fácil de incorporar cambios a lo largo del *project*.

6.4.2.2 Flexibilidad a través de *Tiempo asignado*

Tiempo asignado se refiere al ajuste de periodos cortos de tiempo para hacer el trabajo. Si el trabajo realizado queda incompleto al final del *Time-box*, se mueve a un *Time-box* posterior. Ejemplos de *Tiempo asignado* incluye la limitación de los *Daily Standup Meetings* de 15 minutos y el establecer la duración del *Sprint* de una a seis semanas. *Time-box* proporciona la estructura necesaria para los *proyectos Scrum* los cuales tienen un elemento de incertidumbre, son dinámicos por naturaleza y son propensos a cambios frecuentes. *Time-boxes* ayudan a medir el progreso del *project* y permiten que el equipo identifique fácilmente cuando se necesitará modificar un proceso o enfoque.

Time-boxed Sprints contribuyen en gran medida hacia el cumplimiento de los plazos y al logro de altos niveles de productividad. *Sprint* promueve el orden y la coherencia en un ambiente de trabajo volátil, y proporciona una plataforma para medir resultados y obtener información en un corto espacio de tiempo.

Sprints también permiten la evaluación frecuente de los progresos y los métodos que se utilizan para gestionar el *project*, incluyendo la gestión del cambio eficaz. Los errores o problemas pueden ser identificados temprano y pueden rectificarse rápidamente.

Mediante el uso de *Tiempo asignado* en *Sprints*, el equipo vuelve a visitar con frecuencia el proceso de estimación del trabajo a realizar, por lo que la proyección de tiempo y esfuerzo que se requiere es más precisa con cada subsiguiente *Sprint* a medida que el *project* avanza. Estos ciclos iterativos también motivan a los miembros del equipo a lograr los objetivos previstos y las metas incrementales durante la trayectoria hacia el objetivo más grande.

6.4.2.3 Flexibilidad a través de equipos multi-funcionales y auto-organizados

Las estructuras multi-funcionales y auto-organizadas del *Equipo Scrum* les permiten a los miembros del equipo enfocarse en los resultados deseados del *Sprint*. El equipo tiene un conjunto definido de objetivos durante cada *Sprint* y la flexibilidad para dar cuenta de un cambio en los objetivos antes de comenzar el próximo *Sprint*.

El uso de equipos multi-funcionales también se asegura de que todas las habilidades y conocimientos necesarios para llevar a cabo el trabajo del *project* existan dentro del equipo. Esto proporciona un modelo de trabajo eficiente que da lugar a la creación de entregables listos para demostrarlos al *Propietario del producto* y/o otros *socios*.

Autol-organización garantiza que los miembros del *Equipo Scrum* decidan por sí mismos *la forma* de hacer el trabajo del proyecto sin la microgestión de las tareas por un alto directivo.

El tener equipos multi-funcionales y auto-organizados le permite al grupo adaptarse y administrar los trabajos en curso y algunos *issues* menores o cambios sin tener que obtener el apoyo o la experiencia de miembros fuera del equipo, y en el proceso, *Crear entregables* que estén listos para ser enviados en caso que sea necesario.

6.4.2.4 Flexibilidad a través de *customer value-based prioritization*

La *prioritization* de las necesidades y el trabajo en un proyecto Scrum siempre se determina en base al valor proporcionado al *customer*. En primer lugar, al inicio de un proyecto, los requisitos iniciales se priorizan en función del valor que cada requisito proveerá - esto está documentado en el *Prioritized Product Backlog*. Cuando se realiza una solicitud para un requisito nuevo o un cambio a uno ya existente, esto se evalúa durante el proceso de *Mantenimiento de la lista priorizada de pendientes del producto*. Si se considera que el cambio proporcionará más valor que otros requisitos existentes, se añadirá y priorizará de acuerdo a la versión actualizada del *Prioritized Product Backlog*. Así, el *Prioritized Product Backlog* ofrece la posibilidad de incorporar cambios y agregar nuevos requisitos cuando sea necesario.

Es importante tener en cuenta que los nuevos requisitos y cambios añadidos al *Prioritized Product Backlog* pueden disminuir la prioridad de otros *User Stories* existentes en el *Backlog*: así, estos *User Stories* de menos prioridad se pueden implementar más adelante dependiendo de su nueva prioritization. Debido a que los *customers* están íntimamente involucrados con la *prioritization* de las necesidades y de sus correspondientes *User Stories* en el *Prioritized Product Backlog*, esta práctica asegura que los requisitos que los *customers* consideran de "alto valor" se completen lo antes posible, y que el proyecto inicie la entrega de valor temprano en el proyecto.

6.4.2.5 Flexibilidad a través de la integración continua

Utilizando técnicas de integración continua, los miembros del *Equipo Scrum* pueden incorporar características nuevas y modificadas en las entregas siempre que sea posible. Esto reduce el riesgo de que varios miembros del equipo hagan cambios en componentes redundantes (por ejemplo, código obsoleto en los productos de software, diseños antiguos para la fabricación de piezas). Esto asegura que sólamente se esté trabajando en la última función de la versión y en evitar *issues* en compatibilidad.

6.5 Integración del cambio

Dependiendo de la industria y el tipo de proyecto, la prioridad de las características y los requisitos para un proyecto pueden permanecer fijos durante períodos significativos de tiempo, o pueden cambiar con frecuencia. Si los requisitos del *project* son generalmente estables, normalmente hay sólo pequeños cambios realizados en el *Prioritized Product Backlog* en todo el desarrollo, y los *Equipos Scrum* pueden trabajar secuencialmente en completar los requisitos que le proporcionan el valor máximo al *customer* como se priorizó en el *Prioritized Product Backlog*. En entornos estables, la longitud del *Sprint* es generalmente más larga, de 4 a 6 semanas.

Si los requisitos del *project* cambian a lo largo del *project*, por ejemplo, debido a las modificaciones de *Business Requirements*, el mismo método sigue siendo eficaz. Antes de comenzar un *Sprint*- durante los procesos de *Creación de la lista priorizada de pendientes del producto* or *Mantenimiento de la lista priorizada de pendientes del producto* - los requisitos de mayor prioridad en el *Prioritized Product Backlog* se seleccionan normalmente para ser completados en ese *Sprint*. Dado a que los cambios se han tenido en cuenta en el *Prioritized Product Backlog*, el equipo sólo tiene que determinar el número de tareas que se pueden realizar en el *Sprint*, basado en el tiempo y los recursos proporcionados. La gestión del cambio se ejecuta en los procesos de priorización actuales y se le agregan tareas al *Prioritized Product Backlog*.

6.5.1 Los cambios a un *Sprint*

Si hay una solicitud de cambio que puede tener un impacto significativo sobre un *Sprint* en progreso, el *Propietario del producto*, después de consultar con *socios* relevantes, decide si el cambio puede esperar

hasta el próximo *Sprint* o si representa una situación urgente que puede requerir finalizar el *Sprint* actual y comenzar uno nuevo.

El marco de Scrum especifica claramente que el alcance de un *Sprint* no se puede cambiar una vez que comienza el *Sprint*. Si el cambio requerido es tan importante que los resultados del *Sprint* no tendrían ningún valor sin él, entonces el *Sprint* debe ser terminado. Si no, entonces el cambio se incorpora en un *Sprint* más adelante (como se muestra en la Figura 6-6).

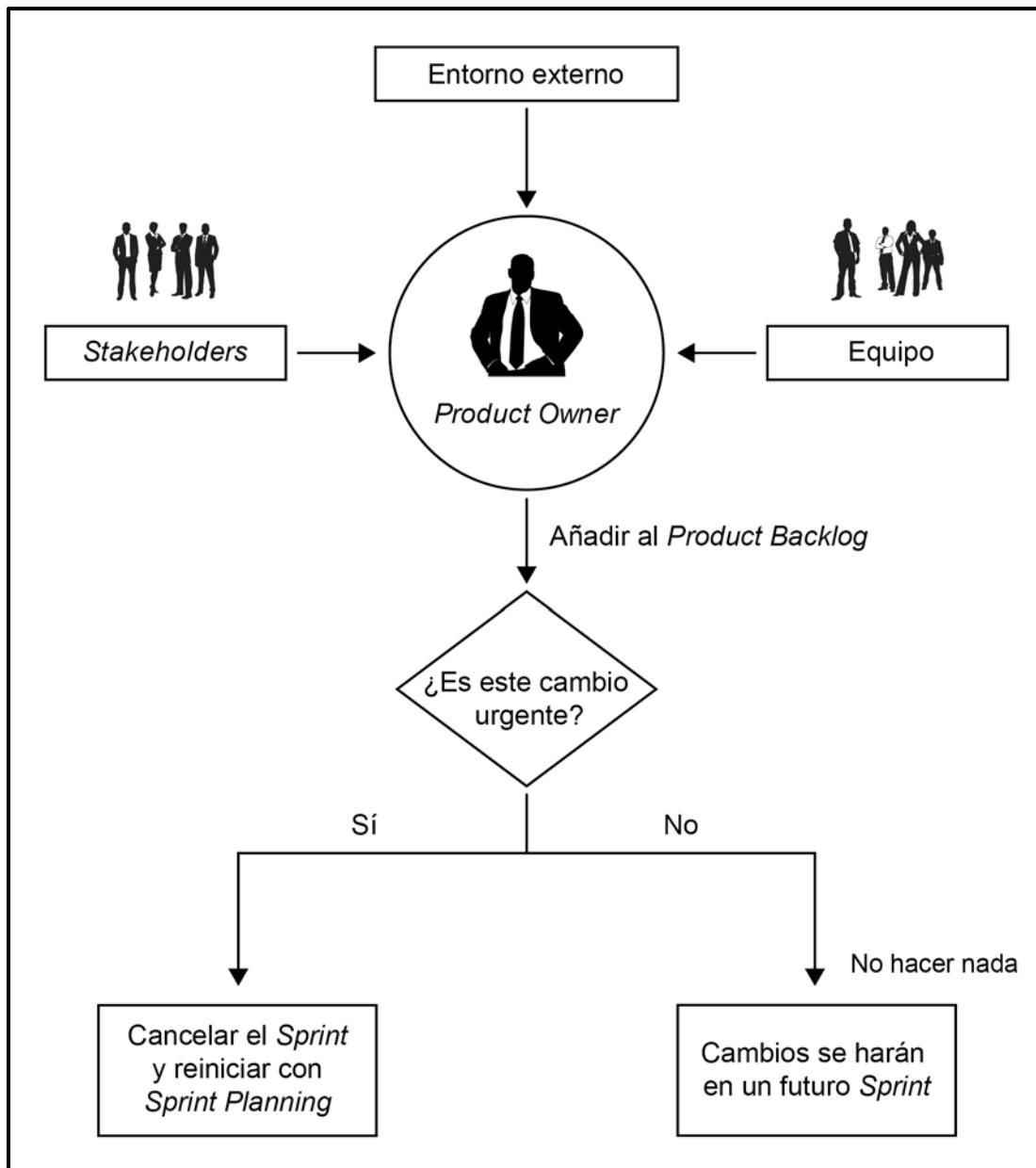


Figura 6-6: Integración del cambio en Scrum

Sólo hay una excepción a esta regla de no modificar el alcance de un *Sprint* una vez que ha comenzado. Si el *Equipo Scrum* determina que se ha sobreestimado en gran medida el esfuerzo durante el *Sprint* y no tiene capacidad para poner en práctica *User Stories* adicionales, el equipo le puede preguntarle al *Propietario del producto* cuáles *User Stories* se han de incorporar en el *Sprint* actual.

Al bloquear el alcance de cada *Sprint*, el equipo es capaz de optimizar y administrar con eficiencia su trabajo y esfuerzo. Un beneficio adicional es que el equipo no tiene que preocuparse por la gestión de los cambios una vez que comienzan a trabajar en un *Sprint*. Esta es una gran ventaja del marco de Scrum en comparación con la gestión tradicional de *projects*.

En la gestión tradicional de *projects*, los cambios pueden ser solicitados y aprobados en cualquier momento durante el ciclo de vida del *project*. Esto a menudo crea confusión entre los miembros del equipo del *project*, disminuye la motivación del equipo debido a la discontinuidad, y da lugar a una falta de concentración y el equipo tiene la sensación de que "nunca se acaba nada." Por otro lado, en los *projects* Scrum, los cambios no se permiten una vez que se inicia un *Sprint*. Esto asegura que en cada *Sprint*, el equipo complete entregables y que las tareas se lleven a cabo. Por otra parte, el negocio reconoce los beneficios tangibles de los Entregables que están potencialmente listos para la entrega al final de cada *Sprint*.

Además, como el *Propietario del producto* y los socios son conscientes de que los cambios no se permiten una vez que el *Sprint* comienza, y un *Sprint* dura entre 1 y 6 semanas, ellos definen y priorizan las necesidades durante los procesos adecuados de *Create Epic(s)*, *Creación de la lista priorizada de pendientes del producto*, y *Mantenimiento de la lista priorizada de pendientes del producto*.

6.5.1.1 Impacto del cambio esperado en el *length of Sprint*

Dado que los cambios no están permitidos durante un *Sprint*, el impacto y la frecuencia de los cambios previstos pueden tener un impacto en la decisión relacionada con la longitud del *Sprint* cuando ésta se determina durante el proceso de *Conduct Realease Planning*.

Si los requisitos del proyecto son generalmente estables y no se esperan grandes cambios en un futuro próximo, la longitud de un *Sprint* se puede ajustar para que sea más larga, de 4 a 6 semanas. Esto les proporciona estabilidad a los miembros del *Equipo Scrum* para trabajar en los requisitos de *Prioritized Product Backlog* durante largos períodos de tiempo sin tener que pasar por los procesos de *Elaborar historias de usuario*, *Aprobar, estimar y asignar historias de usuarios*, *Elaboración de tareas*, *Estimate Task*, y otros procesos relacionados que se llevan a cabo para cada *Sprint*.

Sin embargo, si los requisitos del *project* no están muy bien definidos o si se esperan cambios significativos en el futuro inmediato, el *Length of Sprint* puede ser relativamente corto, de 1 a 3 semanas. Esto les proporciona estabilidad a los miembros del *Equipo Scrum* de trabajar en *Sprints* más cortos y entregar resultados, los que pueden ser evaluados por el *Propietario del producto* y los socios al final del *Sprint*. Esto también proporciona la flexibilidad suficiente para que puedan aclarar los requisitos y realizar cambios en el *Prioritized Product Backlog* al final de cada *Sprint*.

Para obtener los máximos beneficios de un *projecto Scrum*, siempre se recomienda mantener el *Sprint Time-boxed* a 4 semanas, a menos que existan *projects* con requisitos muy estables, donde los *Sprints* se pueden extender hasta 6 semanas.

La figura 6-7 muestra el impacto del cambio esperado en *Length of Sprint*.

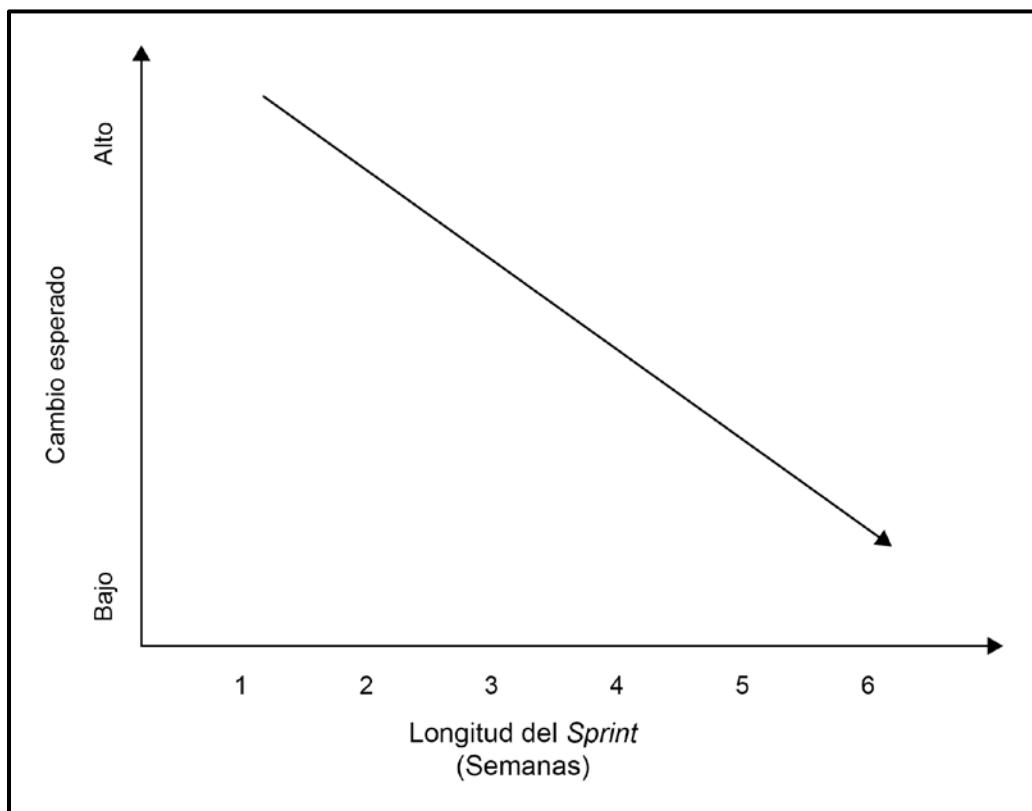


Figura 6-7: Impacto del cambio esperado en *Length of Sprint*

Sin embargo, es importante tener en cuenta que el cambio esperado no es el único factor utilizado para determinar el *Length of Sprint*. Otros factores que también deben tenerse en cuenta son:

- El tiempo real para realizar su trabajo (si el proyecto o entorno corporativo necesita un tiempo específico para realizar tareas de forma, eso podría determinar el *Length of Sprint*)
- Fecha prevista para su lanzamiento (el *Length of Sprint* debe tener en cuenta las fechas de lanzamiento para el *product* o el servicio en general)
- Cualquier otro factor que determine el *Propietario del producto* o el *Scrum Master* que deben tenerse en cuenta al determinar el *Length of Sprint*

Es importante tener en cuenta que el cambio de *Length of Sprint* no debe decidirse a la ligera o de manera periódica (por ejemplo, no es recomendable tener un *Sprint* de 3 semanas, luego uno de 2 semanas, el próximo de 4 semanas, etc). *Length of Sprint* preferentemente debe ser consistente. Uno de los mayores impactos del cambio de *Length of Sprint* es que causa un restablecimiento en todo el seguimiento a nivel de

project. Las velocidades anteriores pueden llegar a ser inútiles para la previsión y la planificación de los futuros *Sprints*. Sin una velocidad precisa (que es una medida primaria en cualquier proyecto Scrum), el *Equipo Scrum* no puede medir la eficacia o elegir adecuadamente el número de *User Stories* para asumir la planificación del próximo *Sprint*. Así que una vez que el *Length of Sprint* se decide, se debe tener preferiblemente constante durante toda la duración del proyecto o a través de múltiples ciclos de *Sprint*.

6.5.1.2 Gestión de cambios y estética a través de *Prioritized Product Backlog Grooming*

Un *Prioritized Product Backlog* típico tendrá todos los *User Stories*, sus estimaciones de tiempo (incluyendo las estimaciones revisadas), y el estado de las necesidades de mayor prioridad. También se incorporan *User Stories* nuevos o revisados que resultaron de cambios en *Business Requirements*, pedidos de los *customers*, condiciones externas del mercado, y/o lecciones aprendidas de *Sprints* anteriores.

Una de las responsabilidades principales de los *Propietario del producto* es preparar el *Prioritized Product Backlog* para garantizar que los requisitos priorizados en el *Prioritized Product Backlog* se incluyan en los próximos dos o tres *Sprints* y se refinen en acuerdo con los *User Stories*. Se recomienda que el *Propietario del producto* pase una cantidad significativa de tiempo en cada *Sprint* para mantener el *Prioritized Product Backlog*. El *Propietario del producto* es responsable de añadir y modificar elementos al *Prioritized Product Backlog* en respuesta a los cambios y es responsable de proporcionar *User Stories* más detallados que se utilizarán en el próximo *Sprint*.

Este mantenimiento ayuda a asegurar que la refinación de los requisitos y sus *User Stories* se hagan mucho antes del *Sprint Planning Meeting* para que el equipo tenga un conjunto de historias muy bien analizado y claramente definido que pueda ser dividido fácilmente en tareas y, posteriormente, estimado. Basado en las lecciones aprendidas del *Sprint* actual, puede haber cambios en los requisitos, o puede haber una repriorización que se pueda incorporar fácilmente en *Sprints* posteriores. Este mantenimiento apoya y mejora la flexibilidad del modelo Scrum mediante la incorporación de los últimos avances técnicos y de negocio en futuros *Sprints*.

Product Backlog Review Meeting (también referido como *Prioritized Product Backlog Grooming Session*) es una reunión formal durante el proceso de *Mantenimiento de la lista priorizada de pendientes del producto*, que ayuda al *Equipo Scrum* a repasar y alcanzar consenso sobre el *Mantenimiento de la lista priorizada de pendientes del producto*. Sin embargo, aparte del *Prioritized Product Backlog Review Meeting*, *Prioritized Product Backlog*, el mantenimiento debería ocurrir durante todo el *project* y puede incluir situaciones en las que el *Propietario del producto* escribe nuevos *User Stories* o vuelve a priorizar *User Stories* en el vigente *Prioritized Product Backlog*, y los miembros del *Equipo Scrum* o socios le dan sus sugerencias sobre los nuevos *User Stories* al *Propietario del producto*, y así sucesivamente.

Es importante tener en cuenta que cualquier elemento de *Prioritized Product Backlog* está siempre abierto para la re-estimación hasta que el *Sprint Backlog* sea finalizado en el proceso *Create Sprint Backlog*. Después de eso, los cambios se podrán seguir haciendo inclusive hasta momentos antes del *Sprint Planning Meeting*, si es necesario.

6.5.1.2.1 Reunión eficaz del *Product Backlog Review* (o sesión de mantenimiento de *Prioritized Product Backlog*)

El *Propietario del producto* es quien está encargado de que se lleve a cabo un *Product Backlog Review Meeting* durante el proceso de *Prioritized Product Backlog*. Es importante que el *Propietario del producto* establezca los objetivos y lo ideal sería desarrollar una agenda antes de comenzar el *Product Backlog Review Meeting*. Sin esto, la sesión no tendría estructura y podría resultar improductiva. También es importante limitar el número de *socios* que participan en la reunión. El tener demasiados participantes tiende a disminuir la eficiencia general de la reunión. El *Propietario del producto* debe invitar sólo a los *socios* cuyas votaciones se requieren para la sesión de la preparación. Todos los miembros del *Equipo Scrum* deben estar incluidos debido a que su opinión es valiosa para el trabajo que se realiza y los *issues* encontrados. Si los resultados de la sesión de preparación o cuidado resultan en *repriorization* o cambio en el *Prioritized Product Backlog*, es importante que el equipo esté de acuerdo con esos cambios.

6

Una sesión de preparación eficaz debe dar lugar a *Prioritized Product Backlog Items (PBIs)* claramente definidos para que el *Equipo Scrum* entienda los requisitos del *customer*. Esto también ayuda a que el equipo se familiarice con todos los *User Stories* en caso de que uno o más de ellos sean incluidos en un *Sprint* a corto plazo. *Acceptance and Done Criteria* también pueden ser discutidos durante las sesiones de preparación.

Los ejercicios de mantenimiento no son *Time-boxed* en Scrum. *Prioritized Product Backlog Grooming* es una actividad continua para el *Propietario del producto*.

6.5.1.3 Gestión de cambios durante *Demostración y validación del Sprint*

Aunque el *Propietario del producto* tiene la última palabra sobre *Prioritized Product Backlog Items* y si se deben aceptar o rechazar los *User Stories* (correspondientes al *Approved Change Requests*) presentados durante el proceso de *Demostración y validación del Sprint*, es la responsabilidad del *Scrum Master* garantizar que los requisitos y *Acceptance Criteria* no se alteren durante el *Sprint Review Meeting* de los *User Stories* completados en el *Sprint* actual. Esto evita el rechazo de *User Stories* futuros basado en el hecho de que no cumplen los requisitos recién cambiados. Si los requisitos se deben cambiar, cualquier PBI correspondiente debe revisarse para adaptarse a los requisitos modificados en un futuro *Sprint*.

6.6 Cambio en *Portfolios* y *Programs*

Cualquier cambio que se produce cualquiera de los *programs* o *portfolios* puede tener un efecto de cascada en todos los *projects* dependientes y *Sprints*. Por lo tanto, es aconsejable minimizar los cambios en estos niveles más altos. Si se requiere un cambio y todos los *socios* están de acuerdo en hacer el cambio a estos niveles, lo siguiente se deberá tener en cuenta.

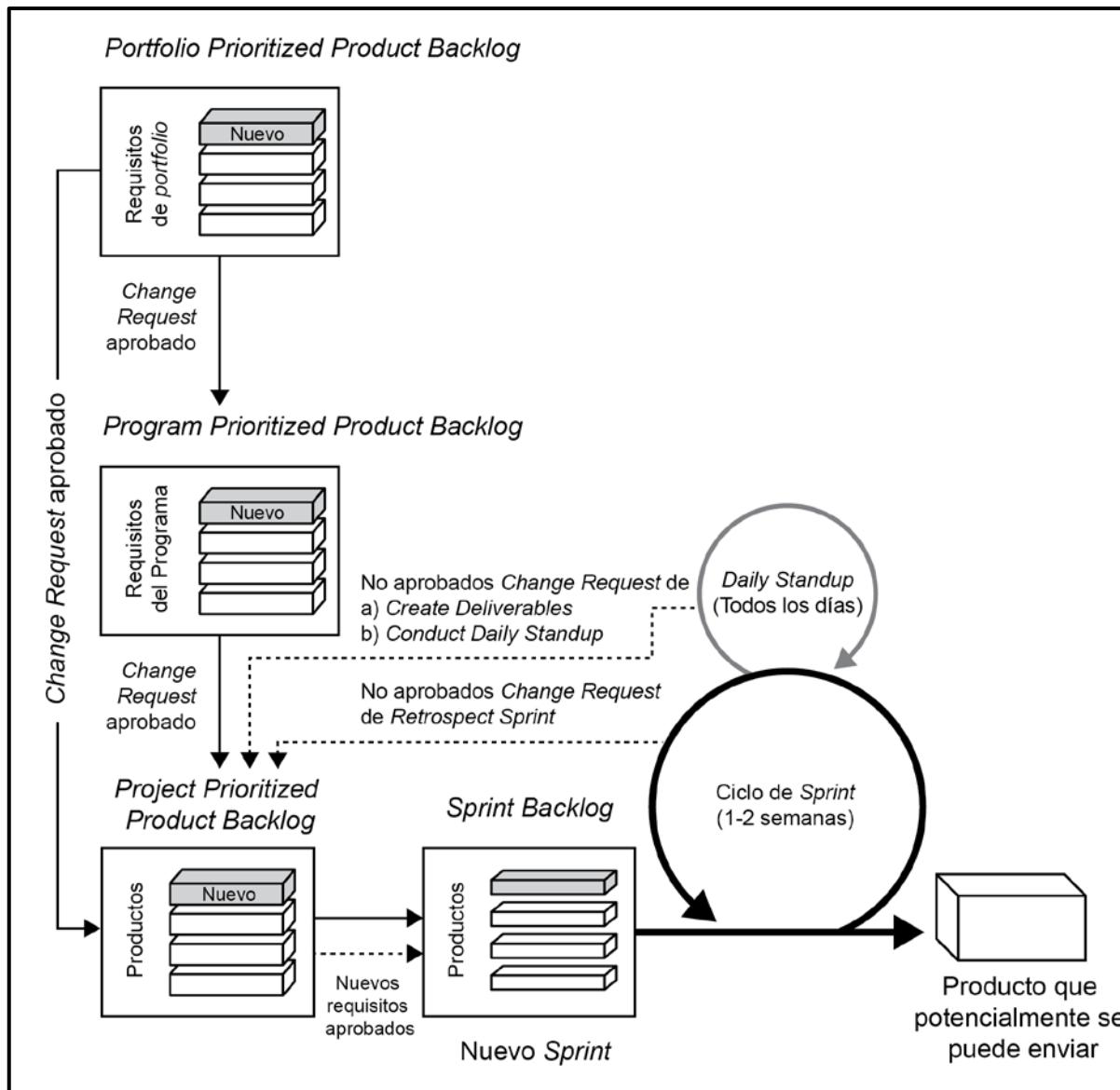
6.6.1 En *Portfolio*

1. No se recomienda hacer cambios entre dos *Portfolio Backlog Meetings*.
2. Si el cambio es menor, el *Portfolio Propietario del producto* debe asegurar la aprobación de los correspondientes *socios* (por ejemplo, el patrocinador, el *customer*, y el usuario final) y luego añadir los requisitos al *Portfolio Backlog*. Los *Propietario del producto* del *program* y del *project* tendrán en cuenta los requisitos para su inclusión en futuros *Sprints*.
3. Si el cambio es importante, los esfuerzos del *portfolio*, junto con los *programs* asociados, *projects* y *Sprints* tienen que parar, y un *Portfolio Backlog Meeting* debe llevarse a cabo para determinar los próximos pasos.
4. *Portfolio Prioritized Product Backlog Meetings* (también denominados *Portfolio Backlog Meetings*), deben llevarse a cabo en intervalos de 4 a 12 meses. La frecuencia y el impacto de los cambios en un *portfolio* determinan en gran medida la duración de tiempo entre dos *Portfolio Backlog Meetings*. Si son varios los cambios esperados en el *portfolio*, es preferible llevar a cabo *Portfolio Backlog Meetings*, en intervalos más regulares (por ejemplo, 4 a 6 meses); pero si hay menos cambios esperados y si los requisitos son estables, la duración entre dos *Portfolio Backlog Meetings* podría incrementarse (por ejemplo, de 9 a 12 meses).

6.6.2 En *Program*

1. No se recomienda hacer cambios entre dos *Program Backlog Meetings*.
2. Si el cambio es menor, el *Program Propietario del producto* debe obtener la aprobación de los relevantes *socios* (por ejemplo, patrocinador, *customer*, y el usuario final) y el *Portfolio Propietario del producto* y luego agregarle los requisitos al *Program Backlog*. Los *Propietario del producto* del *project* tendrán en cuenta los requisitos para la inclusión en futuros *Sprints*.
3. Si el cambio es importante, los esfuerzos de los *programs*, junto con los *projects* asociados y *Sprints* tienen que parar, y la reunión *Prioritized Product Backlog* debe llevarse a cabo para determinar los próximos pasos.
4. *Program Prioritized Product Backlog Meetings* (también conocidas como *Program Backlog Meetings*), deben llevarse a cabo preferentemente en intervalos de 2 a 6 meses. La frecuencia y el impacto de los cambios en un *program* determinan en gran medida la duración de tiempo entre dos *Program Backlog Meetings*. Si hay varios cambios previstos en el *program*, es preferible llevar a cabo *Program Backlog Meetings* en intervalos más regulares (por ejemplo, de 2 a 3 meses); pero si hay menos cambios esperados y si los requisitos son estables, la duración entre dos *Program Backlog Meetings* podrían aumentarse (por ejemplo, de 5 a 6 meses).

La figura 6-8 muestra cómo los cambios pueden ser manejados dentro del flujo de Scrum para ambos *portfolios* y *programs*.

Figura 6-8: La incorporación de cambios en el *portfolio* y *program*

6.7 Resumen de responsabilidades

Función	Responsabilidades
<i>Cuerpo de asesoramiento de Scrum</i>	<ul style="list-style-type: none"> • Proporciona una guía general para los procedimientos de gestión de cambios que se deben seguir durante todo el <i>project</i>
<i>Portfolio Propietario del producto</i>	<ul style="list-style-type: none"> • Proporciona <i>Change Requests</i> para <i>portfolios</i> • Aprueba los <i>products</i> que son modificados, eliminados o añadidos de acuerdo con los requisitos de <i>portfolio</i>
<i>Portfolio Scrum Master</i>	<ul style="list-style-type: none"> • Facilita la identificación, evaluación y gestión de los <i>Change Requests</i> para los <i>portfolios</i>
<i>Program Propietario del producto</i>	<ul style="list-style-type: none"> • Proporciona solicitud de cambio para los <i>programs</i> • Aprueba los <i>products</i> que son modificados, eliminados o añadidos de acuerdo con los requisitos del <i>program</i>
<i>Program Scrum Master</i>	<ul style="list-style-type: none"> • Facilita la identificación, evaluación y gestión de los <i>Change Requests</i> para los <i>programs</i>
<i>Stakeholder(s)</i>	<ul style="list-style-type: none"> • Proporciona solicitud de cambios • Participa en la aprobación y priorización de <i>Change Requests</i>
<i>Propietario del producto</i>	<ul style="list-style-type: none"> • Proporciona solicitudes de modificación de un proyecto • Evalúa el impacto de las solicitudes de cambio planteado por el <i>portfolio</i>, <i>program</i> o <i>project</i> • Prioriza <i>User Stories</i> en el <i>Prioritized Product Backlog</i> del <i>project</i> • Evalúa el impacto de los problemas sobre los objetivos del <i>project</i> identificados por el <i>Equipo Scrum</i> • Les proporciona una comunicación clara a los socios sobre <i>Product Backlog Items</i> que se han vuelto a priorizar
<i>Scrum Master</i>	<ul style="list-style-type: none"> • Facilita la identificación y evaluación de los problemas y <i>Change Requests</i> por el <i>Equipo Scrum</i>
<i>Equipo Scrum</i>	<ul style="list-style-type: none"> • Sugiere mejoras o cambios durante los procesos de <i>Crear entregables</i> y <i>Llevar a cabo el Standup diario</i>

Tabla 6-1: Resumen de las responsabilidades relacionadas con el cambio

6.8 Scrum vs gestión de proyectos tradicional

La gestión del cambio en los *projects* gestionados tradicionalmente está estrechamente relacionada con Configuration Management. Todos los cambios están basados en la magnitud de variación del valor de línea de base. Al director del proyecto se le permite gestionar las actividades y decisiones diarias del proyecto. Cuando un *Change Requests* supera las tolerancias definidas, el director de proyecto debe escalar la propuesta de cambio a niveles superiores de gestión y esperar la decisión antes de hacerla efectiva. El Gerente del Proyecto registra primero la petición de cambio en un *Issue Log* o *Change Log* y luego se les entrega el cambio a las autoridades superiores. Estos podrían incluir el patrocinador del proyecto, así como otros *socios* relevantes y aquellos que toman decisiones sobre el caso. En algún momento, se llevará a cabo una evaluación de impacto. Con base al impacto estimado del cambio, se tomará una decisión con respecto a si el cambio debe aplicarse o no. El director del proyecto también podrá proponer posibles soluciones a los problemas planteados por el cambio. Si las autoridades superiores deciden proceder con el cambio, el director del proyecto es responsable de asegurar que el cambio se implemente correctamente.

El cambio en Scrum funciona de manera muy diferente en comparación con la gestión de proyectos tradicional. El marco de Scrum está muy sintonizado con la gestión de cambios de manera eficaz y eficiente. Cada vez que el *Propietario del producto* o el *Equipo Scrum* reconoce un problema o defecto o identifica un elemento de *Prioritized Product Backlog* que necesita ser modificado, sustituido o añadido, el cambio se realiza en el *Prioritized Product Backlog*. Del mismo modo, la alta dirección, el *Propietario del producto*, o *Stakeholder(s)* puede(n) añadir *Change Requests* en el *Prioritized Product Backlog*. El *Propietario del producto* y los *Stakeholder(s)* aprueban el *Change Requests* y las nuevas prioridades del *portfolio* en consecuencia. Siempre que hay un problema o una nueva exigencia que se debe atender la cual resulta en cambios inmediatos que afectan el *Sprint* actual, el *Propietario del producto* debe terminar el *Sprint* con la aprobación de los *socios* relevantes. Una vez terminado, el *Sprint* se vuelve a planificar y reiniciar para incorporar los nuevos requisitos.

Sin embargo, si el problema o cambio no es importante y no garantiza un cambio dentro del *Sprint* actual, se añadirá el cambio al *Prioritized Product Backlog* y se incorporará en la planificación para un futuro *Sprint*. Esto le da a los *socios* la capacidad de responder a los cambios en el ambiente externo, mientras se mantiene un cierto grado de control sobre las actividades en curso dentro del proyecto. Además, al final de cada *Sprint*, los Entregables clasificados como *Done* son demostrados por el *Equipo Scrum*. Estos aportes potencialmente enviables, pueden ser revisados por el *Propietario del producto* y otros *socios*.

7. RIESGO

7.1 Introducción

El propósito de este capítulo es definir los riesgos, analizar la gestión de los *risks* en un entorno de Scrum, y considerar las herramientas que faciliten la gestión de los *risks*. Para garantizar la viabilidad del negocio, reducir la probabilidad de fracaso de los *projects*, y tomar decisiones de negocio más informadas, es importante que los *risks* se gestionen con eficacia a través de un enfoque bien organizado y metódico.

En un entorno de Scrum, los *risks* generalmente se minimizan, en gran parte debido al trabajo que se realiza en los *Sprints* en los cuales se produce una serie continua de entregables en ciclos muy cortos, los entregables se comparan con las expectativas, y el *Propietario del producto* participa activamente en el *project*. Sin embargo, hasta en el más simple de los *projects*, las cosas pueden salir mal, por lo que es importante contar con una estrategia para identificar y abordar los *risks*.

Risk, tal como se define en la *Una guía para el conocimiento de Scrum (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolios, programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se le entregará a los *socios*
- *Projects* de cualquier tamaño y complejidad

El término "producto" (*product*) en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria - desde pequeños proyectos o equipos con sólo seis miembros por equipo, hasta proyectos grandes y complejos que cuentan con cientos de miembros por equipo.

Este capítulo está dividido en las siguientes secciones:

7.2 Guía de los roles—En esta sección se proporciona la orientación sobre qué secciones son relevantes para cada rol de Scrum: *Propietario del producto*, *Scrum Master*, y *Equipo Scrum*.

7.3 ¿Qué es *Risk*?—En esta sección se define el *risk* y se explica cómo puede afectar los objetivos de un *project* y contribuir al éxito o al fracaso de un *project*.

7.4 Procedimiento de gestión de *Risks*—Esta sección presenta las técnicas claves de la gestión de *risks* y profundiza en el desarrollo de estrategias para identificar, evaluar y gestionar los *risks*.

7.5 Reducción al mínimo de *risks* mediante el uso de Scrum—Esta sección explica los aspectos claves de Scrum que lo convierten en un marco de gestión ideal para manejar con eficacia los *risks* en varios niveles - *portfolio, program* y *project*.

7.6 Resumen de las responsabilidades—En esta sección se describen las responsabilidades de cada persona o papel en un *project* en relación con la gestión de *risks*.

7.7 Scrum vs Gestión de proyecto tradicional—Esta sección discute los beneficios de la gestión del *risk* utilizando métodos de Scrum en lugar de modelos tradicionales de gestión de *projects*.

7.2 Guía de roles

1. *Propietario del producto*—El *Propietario del producto* es quien tiene las principales responsabilidades del manejo de *risks* en un proyecto, por lo tanto, todo el capítulo es aplicable a este papel.
2. *Scrum Master*— El *Scrum Master* debe estar familiarizado con todo este capítulo con un enfoque principal en las secciones 7.3, 7.4 y 7.7.
3. *Equipo Scrum*— El *Equipo Scrum* debería centrarse principalmente en las secciones 7.3 y 7.7.

7.3 ¿Qué es *Risk*?

Risk se define como un evento incierto, que puede afectar los objetivos de un proyecto y puede contribuir a su éxito o fracaso. *Risks*, con un potencial de impacto positivo en el proyecto se denominan *opportunities*, mientras que las amenazas son *risks* que podrían afectar negativamente a un proyecto. La gestión del riesgo debe hacerse con proactividad, y es un proceso iterativo que debería comenzar al inicio del proyecto y continuar durante todo del proyecto. El proceso de gestión del riesgo debe seguir algunos pasos estandarizados para asegurar que los *risks* son identificados, evaluados y un curso de acción está determinado y para actuar en consecuencia.

Es necesario identificar, evaluar, y responder a los *risks* basándose principalmente en dos factores: la probabilidad de ocurrencia y el impacto probable en caso de que ocurra. Los *risks* de alta probabilidad y alto índice de impacto deben ser abordados antes de aquellos con una calificación más baja. En general, una vez que se detecte un riesgo, es importante comprender los aspectos básicos del riesgo con respecto a las posibles causas, el área de la incertidumbre y los efectos potenciales si se produce el riesgo.

7.3.1 Diferencia entre *Risks* e *Issues*

Risks son las incertidumbres relacionadas con un proyecto que podrían alterar significativamente el resultado del proyecto de una manera positiva o negativa. Dado a que *risks* son las incertidumbres relacionadas con el futuro, no tienen ningún impacto actual en el proyecto, pero podrían tener un impacto potencial en el futuro. Los siguientes son algunos ejemplos de *risks*.

- Incluso después de un amplio entrenamiento, es posible que los representantes de servicio al *customer* no estén listos para tomar pedidos el día oficial del lanzamiento.
- Es posible que los pintores se retrasen debido a las fuertes lluvias, lo que podría influir negativamente en el cronograma del *project*.

Issues son generalmente certezas que se están produciendo en el *project*: por lo que no hay necesidad de realizar una evaluación de la probabilidad como lo haríamos para un *risk*. Los *issues* deben ser tratados. Algunos ejemplos de *issues* son los siguientes:

- La financiación no es aprobada.
- Los requisitos no son claros.

Risks, si no se tratan a tiempo, pueden convertirse en *issues*. El objetivo de la gestión del *risk* es estar preparado, con planes en marcha para hacerle frente a cualquier *risk* que pueda ocurrir.

7.3.2 Risk Attitude

Socios incluyen a todas las personas u organizaciones afectadas por el proyecto, así como los que tienen la capacidad de afectar el proyecto. Es importante entender el *risk attitude* de los *socios*. *Risk attitude* está influenciado por los tres factores siguientes:

1. *Risk appetite*: se refiere a la cantidad de incertidumbre que la organización o *stakeholder* está dispuesto a asumir.
2. *Risk tolerance*: indica el grado, cantidad o volumen de riesgo que los *stakeholders* resistirán.
3. *Risk threshold*: se refiere al nivel en el que un riesgo es aceptable para la organización de los *socios*. Si riesgo caerá por encima o por debajo del *risk threshold*. Si está por debajo, entonces el *stakeholder* u organización es más probable que acepte el riesgo.

En esencia, el nivel de *risk attitude* de los *socios* determina cuánto riesgo los *socios* consideran aceptable y por lo tanto, cuando se decidirán a tomar medidas para mitigar los efectos adversos potenciales de los *risks*. Por lo tanto, es importante entender los niveles de tolerancia de los *socios* en relación a diversos factores como el costo, la calidad, el alcance y los plazos .

Utility Function es un modelo utilizado para medir la actitud y preferencia hacia el riesgo por parte de los *socios*. Este modelo define el nivel o la disposición de los *socios* a aceptar el riesgo. Las tres categorías de *Utility Functions* son las siguientes:

1. *Risk averse*: *Stakeholder* no está dispuesto a aceptar el riesgo, no importa cuál sea el beneficio esperado u oportunidad.
2. *Risk neutral*: El *stakeholder* no es ni *risk averse*, ni *risk seeking* y no se ve afectado por el nivel de incertidumbre de los resultados. Cuando dos posibles escenarios tienen el mismo nivel de beneficio, el *risk neutral stakeholder* no va a estar preocupado si una hipótesis es más arriesgada que la otra.

3. *Risk seeking:* En este caso, el *Stakeholder* está dispuesto a asumir un riesgo, aún si no ofrece un aumento marginal de retorno o beneficio al proyecto.

7.4 Procedimiento de gestión de *risks*

La gestión de riesgos se compone de cinco pasos:

1. *Risk identification:* El uso de diversas técnicas para identificar todos los *risks* potenciales.
2. *Risk assessment:* La evaluación y la estimación de los *risks* identificados.
3. *Risk prioritization:* La priorización de riesgo a ser incluido en el *Prioritized Product Backlog*.
4. *Risk mitigation:* Desarrollo de una estrategia adecuada para hacer frente al riesgo.
5. *Risk communication:* La comunicación de los resultados de los primeros cuatro pasos a las *socios* apropiados, y la determinación de su percepción con respecto a los sucesos inciertos.

7.4.1 Risk Identification

Los miembros del *Equipo Scrum* deberían tratar de identificar todos los *risks* que podrían afectar el *project*. Sólo mirando el *project* desde diferentes perspectivas y utilizando una variedad de técnicas, se puede hacer este trabajo a fondo. *Risk identification* se realiza a lo largo del *project* y los *risks* se convierten en entradas para varios procesos de Scrum incluyendo *Creación de la lista priorizada de pendientes del producto*, *Mantenimiento de la lista priorizada de pendientes del producto*, y *Demostración y validación del Sprint*.

Las siguientes técnicas se utilizan comúnmente para identificar *risks*.

7.4.1.1 Técnicas de *Risk Identification*

1. Repasar las lecciones aprendidas de *Retrospectiva de Sprint* o de procesos de *Retrospectiva del proyecto*

El aprender de *projects* similares y de *Sprints* anteriores del mismo *project*, al igual que la exploración de las incertidumbres que afectaron esos *projects* y *Sprints* puede ser una forma útil de identificar *risks*.

2. *Risk Checklists*

Risk checklists puede incluir los puntos claves a tener en cuenta cuando se identifican los *risks*, *risks* comunes encontrados en el *project* Scrum, o incluso las categorías de *risks* que deben ser atendidas por el equipo. Las listas de verificación son herramientas valiosas para ayudar a asegurar un nivel de *risk identification* detallado.

3. Risk Prompt Lists

Risk prompt lists se utilizan para estimular pensamientos con respecto a la fuente de donde los *risks* se pueden originar. *Risk prompt lists* para industrias y proyectos diversos están disponibles al público.

4. Brainstorming

Sesiones donde los *socios* relevantes y los miembros del Equipo Central de Scrum pertinentes comparten abiertamente ideas a través de discusiones y sesiones de intercambio de conocimientos, algo que normalmente es llevado a cabo por un facilitador.

5. Risk Breakdown Structure (RBS)

Una de las herramientas claves que se utilizan en la identificación de *risks* es un *risk breakdown structure*. En esta estructura, *risks* se agrupan en función de sus categorías o elementos comunes. Por ejemplo, *risks* pueden ser categorizados como algo financiero, técnico o relacionado con la seguridad. Esto permite que el equipo planifique mejor y se encargue de cada riesgo.

7.4.1.2 Risk-Based Spike

Un concepto que puede ser útil en la identificación de *risks* es el de *risk-based spike*. Un (spike) pico es un experimento que consiste en la investigación o la creación de prototipos para entender mejor los *risks* potenciales. En un pico, se realiza un ejercicio intenso en dos o tres días (preferiblemente al comienzo de un *project* antes de los procesos *Desarrollo de épica(s)* o *Creación de la lista priorizada de pendientes del producto*) para ayudar al equipo a determinar las incertidumbres que podrían afectar al proyecto. *Risk-based spikes* son útiles cuando el *Equipo Scrum* se está acostumbrando a trabajar con nuevas tecnologías o herramientas, o cuando los *User Stories* son largos. También ayudan a estimar el tiempo y el esfuerzo con mayor precisión.

7.4.2 Risk Assessment

La evaluación de riesgos ayuda a entender el impacto potencial de un *risk*, ¿qué tan probable es que se produzca, y cuándo es posible que el *risk* se materialice? El efecto total sobre el valor del negocio se debe estimar; si el impacto es lo suficientemente importante como para quizás no justificar el *Justificación del negocio*, una decisión debe ser tomada sobre si sería buena idea continuar con el *project*.

La evaluación de los *risks* se realiza con respecto a la probabilidad, proximidad e impacto. La Probabilidad de *risks* se refiere a la probabilidad de que los *risks* se produzcan, mientras que la proximidad se refiere a cuándo podría producirse el *risk*. *Impacto* se refiere al probable efecto de los *risks* en el *project* o la organización.

Para estimar la probabilidad de *risks*, varias técnicas pueden utilizarse, incluyendo *Probability Trees*, *Pareto Analysis*, y *Probability and Impact Matrix*.

Además de la probabilidad, *risk assessment*, también evalúa el posible efecto neto de *risks* en el proyecto u organización. Estos efectos pueden ser estimados usando técnicas tales como los *Risk Models* y *Expected Monetary Value*.

7.4.2.1 Técnicas de *Risk Assessment*

1. *Risk Meeting*

Los *risks* podrían ser fácilmente priorizados por el *Propietario del producto* llamando a una reunión al Equipo Principal de y, opcionalmente, invitando a los socios relevantes a la reunión. El equipo podría reunir y priorizar diferentes *risk*, basado en su evaluación subjetiva del impacto de los *risks* en los objetivos del *project*.

2. *Probability Trees*

Los eventos potenciales se representan en un árbol con una rama extendida para cada resultado posible de un evento de riesgo. La probabilidad de cada posible resultado se indica en la rama correspondiente y luego se multiplica por su impacto evaluado para obtener un valor esperado para cada resultado posible. Los valores resultantes se suman entre sí para calcular el impacto esperado de un *risko* para un *project* (véase la Figura 7-1).

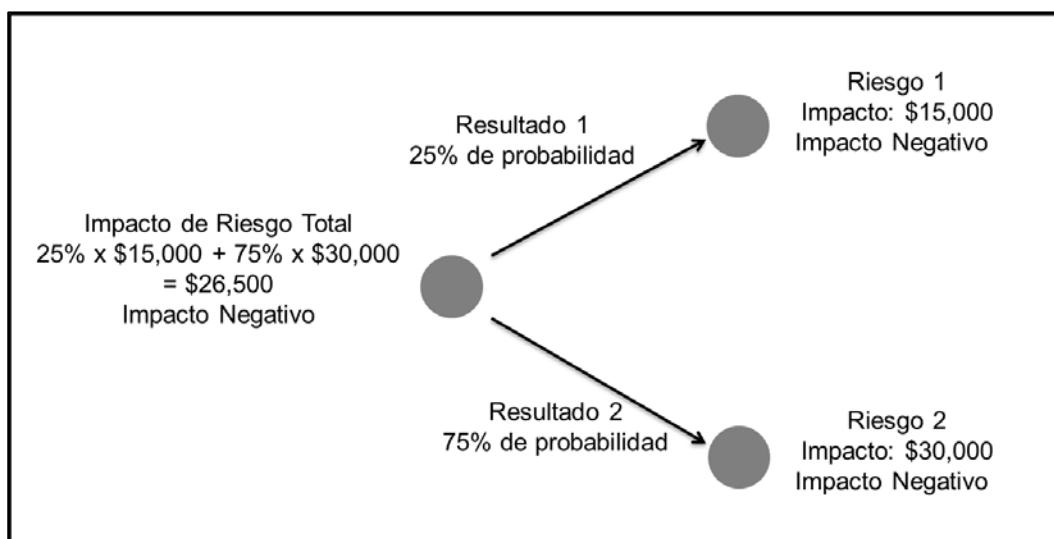


Figura 7-1: Muestra de *Probability Tree*

3. Pareto Analysis

Esta técnica de evaluación del *risk* implica la clasificación de la magnitud de los *risks* lo que ayuda al *Equipo Scrum* identificar los *risks* en el orden de su impacto potencial en el *project*. Por ejemplo, en la figura 7-2, *Riesgo 1* tiene el mayor impacto y preferiblemente debería abordarse primero.

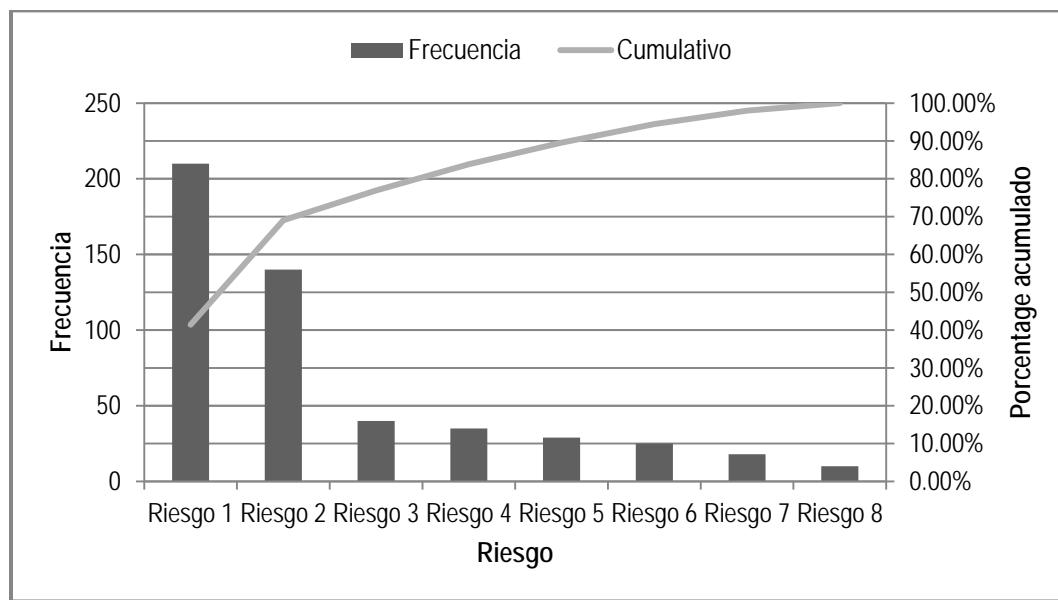


Figura 7-2: Ejemplo del diagrama de Pareto

4. Probability Impact Grid

Cada *risk* se valora por su probabilidad de ocurrencia y de su impacto potencial sobre los objetivos del *project*. En general, una calificación numérica se asigna tanto por la probabilidad y el impacto de forma independiente. Los dos valores se multiplican luego para derivar una puntuación de gravedad de riesgo (o valor de PI), que puede ser utilizado para priorizar *risks*.

Por ejemplo, la puntuación de la gravedad del riesgo para un riesgo con una probabilidad del 50% y una calificación de impacto de 0,6 se calcularía de la siguiente manera:

$$0,5 \text{ (Probabilidad)} \times 0,6 \text{ (Impacto)} = 0,3$$

Los esquemas de calificación utilizados se determinan dentro de la organización o del *project*. A menudo, se utiliza una escala decimal, de cero a uno, donde un rating de probability de 0,5 indicaría un 50% de probabilidad. Otras opciones incluyen una escala del uno al diez, o Alto (3), Medio (2), y Bajo (1).

La figura 7-3 representa el uso de la escala decimal. Cada *risk* se califica en su probabilidad de ocurrencia y el impacto en una escala objetiva.

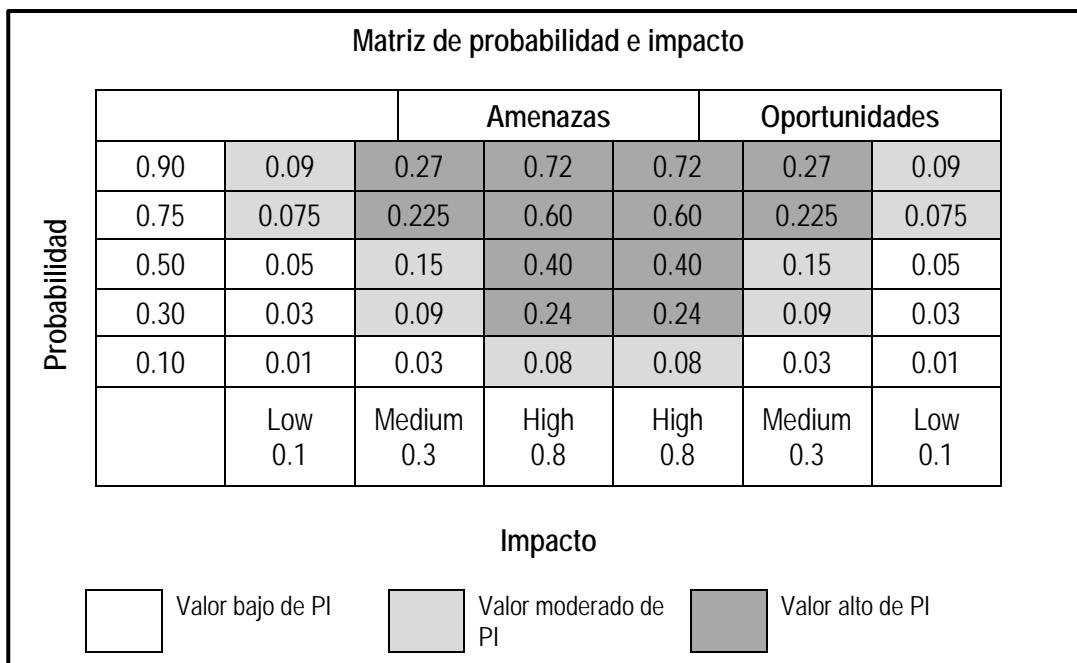


Figura 7-3: Muestra de Matriz de Probabilidad e Impacto

El método de asignación de valores de probabilidad e impacto de *risks* varía en función del *project* y el número de los *risks* que se están evaluando, así como los procesos y los procedimientos organizativos existentes. Sin embargo, aplicando la fórmula simple $P \times I$, la gravedad del riesgo se puede calcular en una escala numérica o categórica.

5. *Expected Monetary Value (EMV)*

El valor monetario de los riesgos se basa en su *Expected Monetary Value (EMV)*. EMV se calcula multiplicando el impacto monetario por la probabilidad de *risk*, según estima el *customer*.

$$\text{Expected Monetary Value} = \text{Impacto del Riesgo (en dólares)} \times \text{Probabilidad de Riesgo (en porcentaje)}$$

Por ejemplo, un *risk* con un impacto negativo estimado de \$ 1.000 y un 50% de probabilidad de que si se produce daría lugar a un EMV de la siguiente manera:

$$\text{EMV} = \$1.000 \times 0,50 = \$500$$

7.4.3 *Risk Prioritization*

Scrum permite la rápida identificación y evaluación de los *risks*. *Risks* identificados se tienen en cuenta al crear un *Prioritized Product Backlog* durante el proceso de *Creación de la lista priorizada de pendientes del producto*, o cuando se actualiza el *Prioritized Product Backlog* durante el proceso *Mantenimiento de la lista*

priorizada de pendientes del producto - por lo que *Prioritized Product Backlog* también podría ser referido como *Risk Adjusted Prioritized Product Backlog*.

Los *risks* podrían ser identificados y evaluados en base a cualquiera de las técnicas mencionadas previamente como *Risk Identification* y *Risk Assessment*.

En los procesos *Create Priorititized Product Backlog* o *Mantenimiento de la lista priorizada de pendientes del producto*, los *User Stories* priorizados del *Prioritized Product Backlog* existente y la lista de prioridades de *risks* se combinan para crear un *Prioritized Product Backlog* actualizado, que incluye los *identified risks*:

Pasos para la actualización de *Prioritized Product Backlog* con *Identified Risks*:

1. Crear una lista con *risks* priorizados. (Por ejemplo, los *risks* pueden ser priorizados por valor utilizando la técnica de *Expected Monetary Value*).
2. Seleccione los *risks* identificados que pueden ser mitigados; y para el cual el equipo decide tomar medidas específicas de riesgo durante el *Sprint* para mitigar tales *risks*.
3. Crear una lista de *User Stories* en el *Prioritized Product Backlog*, que son priorizados por valor (por ejemplo, el valor de cada *User Story* se puede evaluar en función de su esperado *Return on Investment*).
4. Combinar listas en el paso 2 y el paso 3 y darles prioridad por el valor de llegar al *Updated Prioritized Product Backlog*.

La figura 7-4 ilustra el proceso de *risk prioritization*.

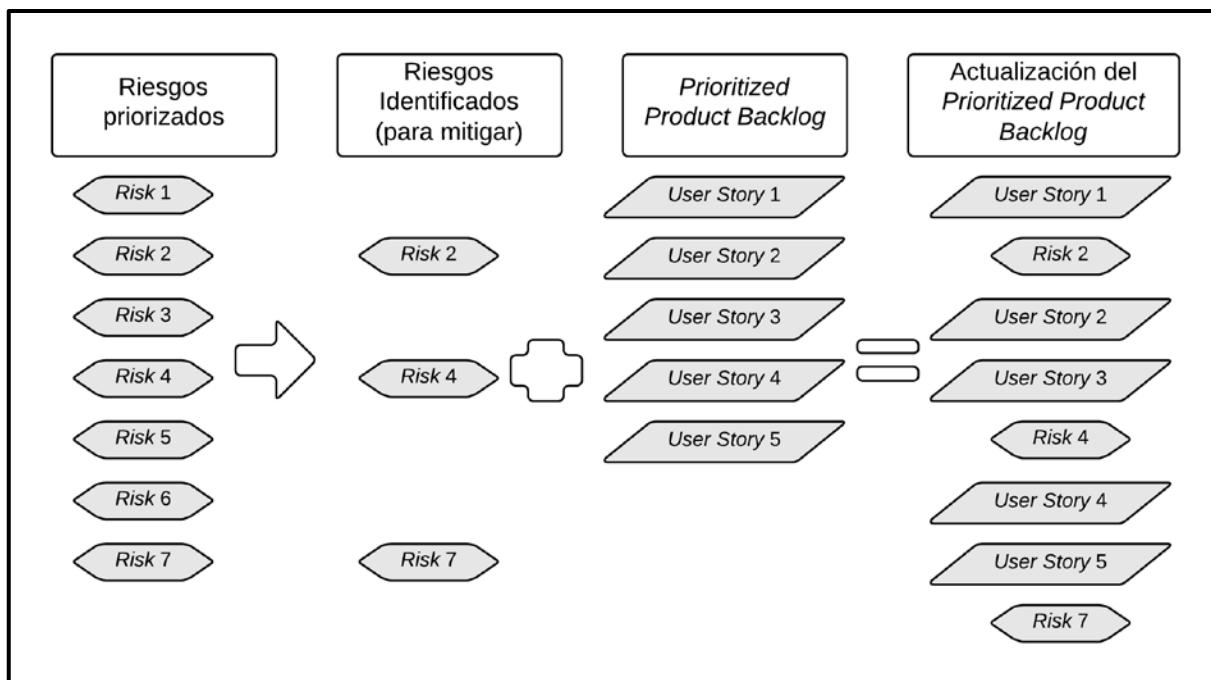


Figura 7-4: Ilustra el proceso de *Risk Prioritization*

7.4.4 Risk Mitigation

La respuesta a cada *risk* dependerá de la probabilidad y el impacto del *risk*. Sin embargo, la naturaleza iterativa de Scrum, con sus ciclos de tiempo de respuesta y retroalimentación rápida permite que las fallas se detecten de forma temprana; por lo tanto, hablando en términos prácticos, tiene una función de mitigación natural construida adentro del sistema.

Risk puede ser mitigado mediante la implementación de una serie de respuestas. En la mayoría de las situaciones, las respuestas son proactivas/preventivas o reactivas. En el caso de un *risk*, un plan B puede ser formulado, que se puede utilizar como una alternativa en caso de que el *risk* se materialice—en este caso, plan B es una respuesta reactiva. A veces, los *risks* se aceptan y son un ejemplo de una respuesta al *risk* que no es ni preventivo ni reactivo. Los *risks* se aceptan debido a varias razones, como en una situación en la que la probabilidad o el impacto de *risk* es muy bajo para una respuesta. La aceptación también puede ser el caso en una situación en la que la aprehensión de *risks* secundarios puede disuadir al *Propietario del producto* de tomar cualquier acción. El esfuerzo realizado por el *Propietario del producto* para reducir la probabilidad o el impacto, o ambos, de riesgo es un ejemplo de una respuesta preventiva a la mitigación de *risks*.

Una vez identificados los *risks* se incluyen como parte del *Prioritized Product Backlog* (vea la Figura 7-4), varios *risks* se mitigan durante el proceso de *Crear entregables* cuando los *Tasks* relacionados con los *User Stories* definidos en el proceso *Prioritized Product Backlog* se completan.

En Scrum, la responsabilidad de *risk* es claramente del *Propietario del producto* para la gestión de *Risks* en relación con los aspectos del negocio, y también del *Scrum Team* por implementar respuestas de los *risks* durante el transcurso de un *Sprint*. *The Cuerpo de asesoramiento de Scrum* puede ser abordado para el asesoramiento sobre la forma en la que las respuestas a los riesgos se aplican y para asegurarse que las acciones estén alineadas con las directrices de la organización en su conjunto. El *Scrum Master* mantiene una estrecha vigilancia sobre los *risks* potenciales que podrían afectar al *project* y mantiene al *Propietario del producto* y al *Equipo Scrum* informado.

7.4.5 Risk Communication

Debido a que los *socios* tienen un interés en el *project*, es importante comunicarse con ellos con respecto a los *risks*. La información proporcionada a los *socios* relacionada con el riesgo debe incluir el impacto potencial y los planes para hacerle frente a cada riesgo. Esta comunicación siempre está en curso y debe ocurrir en paralelo con los cuatro pasos secuenciales discutidos hasta ahora- Identificación de *risks*, la evaluación, *prioritization* y *mitigation*. El *Equipo Scrum* también puede discutir *risks* específicos relacionados con sus *Tasks* con el *Scrum Master* durante *Daily Standup Meetings*. El *Propietario del producto* es responsable de la *prioritzación* de los *risks* y de comunicarle la lista de prioridades al *Equipo Scrum*.

Una herramienta importante que puede ser utilizada para comunicar información relacionada con *risks* es la *Risk Burndown Chart*.

7.4.5.1 Risk Burndown Chart

La gestión de riesgos es esencial para garantizar la creación de valor; Por lo tanto, las actividades de gestión de riesgos se llevan a cabo durante todo el ciclo de vida del *project* y no sólo durante el inicio del *project*.

Cada *risk* se puede evaluar usando diferentes herramientas de *Risk Assessment*. Sin embargo, la herramienta preferida para la evaluación de *risks* para crear un *Risk Burndown Chart* es *Expected Monetary Value (EMV)* como se describe en la sección 7.4.2.1.

La información recopilada durante *risk assessment* se puede utilizar para crear un *Risk Burndown Chart*. Esto representa la severidad del riesgo del *project* acumulativo en el tiempo. Las probabilidades de los diversos *risks* se representan en la parte superior de uno al otro para mostrar riesgo acumulado en el eje y. La identificación y evaluación inicial de los *risks* en el *project* y la creación del *Risk Burndown Chart* se realizan inicialmente. Luego, en intervalos de tiempo predeterminados, los nuevos *risks* pueden ser identificados y evaluados y los *risks* que quedan pueden ser reevaluados y actualizados según la gráfica. Un momento apropiado para hacer esto es durante *Sprint Planning Meeting*. El seguimiento de los *risks* de esta manera le permite al equipo reconocer las tendencias al *risk* y tomar las medidas apropiadas, si fuera necesario.

La figura 7-5 muestra un ejemplo de *Risk Burndown Chart*.

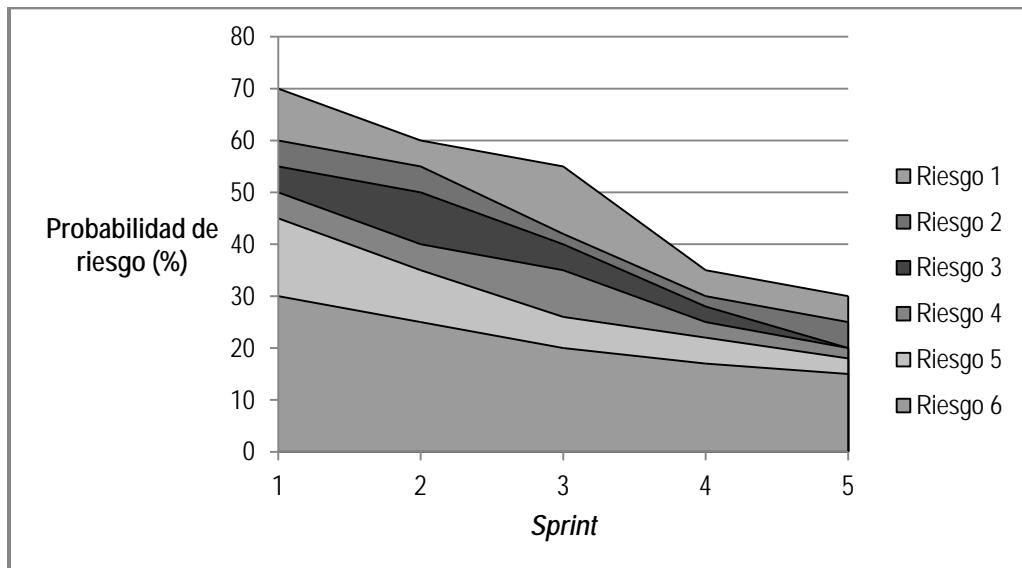


Figura 7-5: Ejemplo de *Risk Burndown Chart*

7.5 Reducción al mínimo de *Risks* a través de Scrum

Al ser un proceso ágil e iterativo, el marco de Scrum minimiza inherentemente el riesgo. Las siguientes prácticas de Scrum facilitan la gestión efectiva del *risk*:

- 1. La flexibilidad reduce el riesgo relacionado con el medio ambiente del negocio**

Risk se reduce en gran medida en Scrum debido a la flexibilidad en la adición o modificación de los requisitos en cualquier momento del ciclo de vida del *project*. Esto le permite a la organización responder a las amenazas o *opportunities* en el entorno empresarial y a las necesidades imprevistas cada vez que surgan, por lo general con un bajo costo de la gestión de tales *risk*.

- 2. La retroalimentación regular reduce el riesgo relacionado con las expectativas**

Al ser iterativo, el marco de Scrum proporciona amplias *opportunities* para obtener información y establecer expectativas en todo el ciclo de vida del *project*. Esto asegura que los *socios* del *project*, así como el equipo, no sean tomados por sorpresa dado a requisitos mal comunicados.

- 3. La propiedad del equipo reduce la estimación de riesgo**

El *Equipo Scrum* hace estimaciones y se hace responsable de los elementos del *Sprint Backlog*, lo que conduce a la estimación más precisa y la entrega oportuna de los incrementos de *products*.

- 4. La transparencia reduce el riesgo de no detectar**

El principio de transparencia (*Transparencia*) de Scrum en torno al cual se construye el marco asegura que los *risk*s se detecten y comuniquen temprano, lo que conduce a un mejor manejo y mitigación de riesgos. Por otra parte, al llevar a cabo *Scrum of Scrums Meetings*, los *Impediments* que un equipo enfrenta en la actualidad pueden considerarse como *risk* para otros *Equipos Scrum* en el futuro. Esto debe ser reconocido en el *Updated Impediment Log*.

- 5. Desarrollo iterativo reduce el riesgo de inversión**

La entrega continua de valor a lo largo del ciclo de vida del *project* Scrum, como Entregables potencialmente listos para la entrega, se crean después de cada *Sprint*, reduciendo así el riesgo de la inversión para el *customer*.

7.6 Risks en Portfolios y Programs

Mientras que algunos *risks* están específicamente relacionados con *projects* individuales, otros pueden tener su origen en los *programs* o *portfolios*, y generalmente serán administrados allí mismo. Sin embargo, *risks* relacionados con un *portfolio* o *program* también tendrán un impacto en los *projects* que forman parte del respectivo *portfolio* o *program*. Durante el *risk assessment* en *portfolios* y *programs*, si se determina que el riesgo puede afectar un proyecto individual, la información relevante sobre el riesgo debe ser comunicada al *Propietario del producto* y al *Equipo Scrum*.

Dependiendo de la gravedad o la prioridad, cuando el *program* o el equipo del *portfolio* comunica un riesgo que impactará un *project* individual, es posible que el *Equipo Scrum* tenga que parar y volver a planificar el *Sprint* actual para encargarse del *riesk*. Para los *risks* menos urgentes, el equipo puede seguir con el *Sprint* actual y atender el *risk* en el siguiente *Sprint*.

7

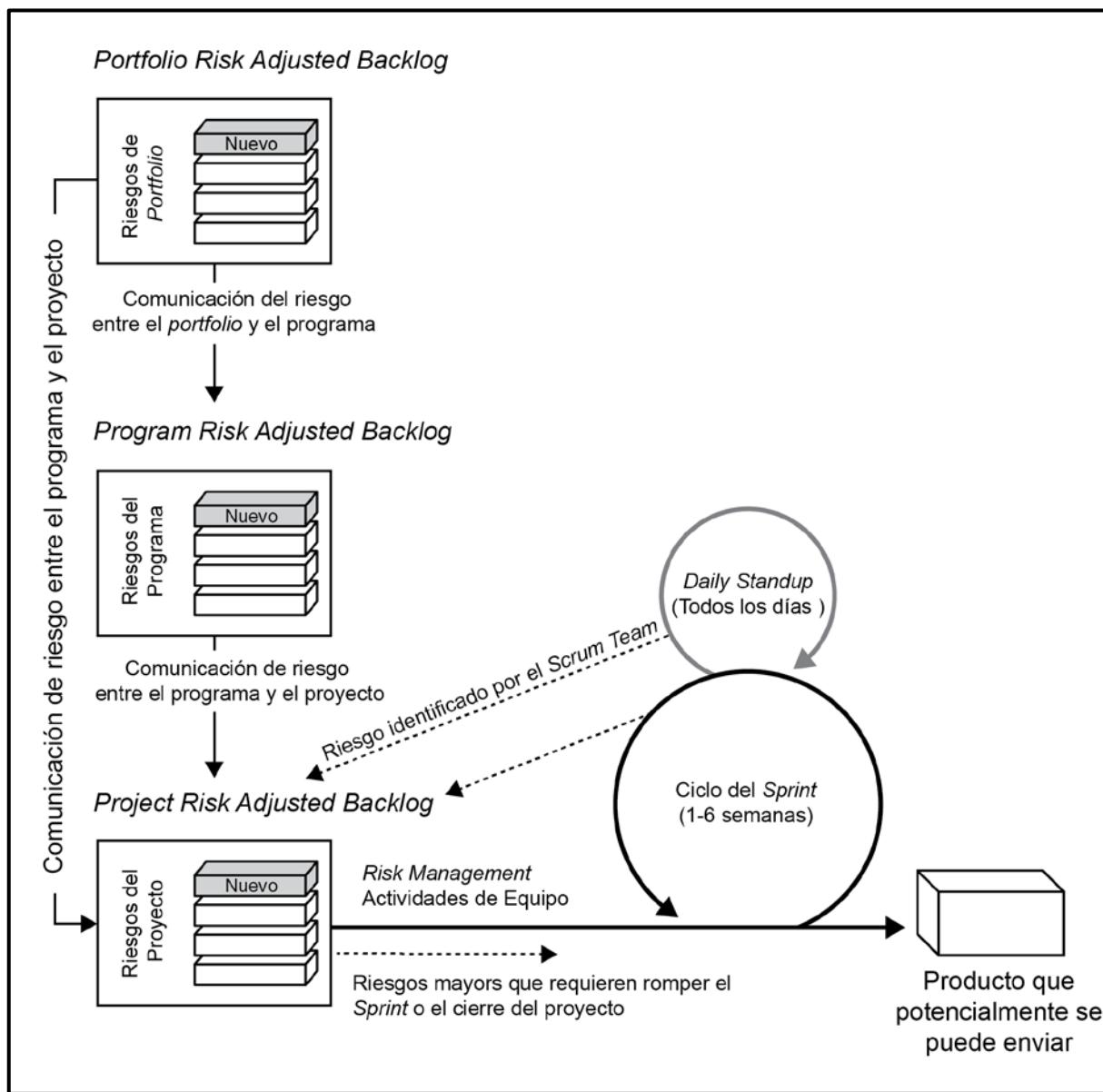
7.6.1 En Portfolio

1. Cuando se identifican los *risks* en *portfolio*, el *Portfolio Propietario del producto* tendrá que captarlos y evaluar la proximidad, la probabilidad y el impacto de cada riesgo identificado, con el fin de priorizar y determinar la respuesta apropiada para el *Portfolio*.
2. El *Portfolio Propietario del producto* también deberá comunicar los *Risks* a la correspondientes *socios*, los equipos de los *programs* y los equipos del *project*. En algunos casos, el equipo del *Portfolio* puede tener que asumir la responsabilidad de *risks* específicos.

7.6.2 En Program

1. Cuando se identifican los *risks* del *program*, el *Program Propietario del producto* debe ponerlos en el programa de *Risk Adjusted Prioritized Product Backlog*, evaluar la proximidad, la probabilidad y el impacto de cada riesgo identificado, a fin de priorizar y determinar las respuestas apropiadas para los programas.
2. El *Program Propietario del producto* también le deberá comunicar los *risks* a las *socios* relevantes y a los equipos de *project*. En algunos casos, el equipo del programa tendrá que asumir la responsabilidad de *risks* específicos.

La figura 7-6 muestra cómo los *risks* se puede manejar dentro del flujo de Scrum tanto en los *portfolios* como en los *programs*.

Figura 7-6: Manejo de *risks* en portfolios y programs

7.7 Resumen de Responsabilidades

En Scrum, las actividades de gestión de *risks* se dividen entre varios papeles con cierta responsabilidad que caen en las manos del *Equipo Scrum*, y en donde el *Scrum Master* facilita el proceso.

Función	Responsabilidades
<i>Cuerpo de asesoramiento de Scrum</i>	<ul style="list-style-type: none"> Proporciona una guía general para el procedimiento de gestión de <i>risks</i> que deben seguirse durante todo el <i>project</i>
<i>Portfolio Propietario del producto</i>	<ul style="list-style-type: none"> Captura y evalúa <i>risks</i> para <i>portfolio</i> Prioriza y comunica <i>risks</i> de relevantes <i>socios</i>, y equipos de <i>projects</i> y <i>programs</i>
<i>Portfolio Scrum Master</i>	<ul style="list-style-type: none"> Facilita la identificación, evaluación y comunicación de los <i>risks</i> de los <i>portfolios</i>
<i>Program Propietario del producto</i>	<ul style="list-style-type: none"> Captura y evalúa <i>risks</i> para los <i>programs</i> Prioriza y les comunica los <i>risks</i> de a los <i>socios</i> y equipos de <i>projects</i> relevantes
<i>Program Scrum Master</i>	<ul style="list-style-type: none"> Facilita la identificación y la evaluación de los <i>risks</i> para los <i>programs</i>
<i>Stakeholder(s)</i>	<ul style="list-style-type: none"> Interactua con el Equipo Central de Scrum para proporcionarles las entradas en la gestión de los <i>risks</i> que afectan el logro de los resultados esperados y los beneficios del <i>project</i>
<i>Propietario del producto</i>	<ul style="list-style-type: none"> Captura y evalúa <i>risks</i> para el <i>project</i> Prioriza y les comunica <i>risks</i> a los relevantes <i>socios</i>, el <i>program</i>, y los equipos de <i>portfolio</i> Asegura que los niveles de riesgo del <i>project</i> se encuentren dentro de los límites aceptables
<i>Scrum Master</i>	<ul style="list-style-type: none"> Facilita la identificación de los <i>risks</i> por el <i>Equipo Scrum</i>
<i>Equipo Scrum</i>	<ul style="list-style-type: none"> Identifica los <i>risk</i> durante el desarrollo del <i>product</i> durante el proceso de <i>Crear entregables</i> Ejecuta las acciones de gestión de riesgos según lo aconsejado por el <i>Propietario del producto</i>

7

Tabla 7-1: Resumen de las responsabilidades pertinentes a *Risks*

7.8 Scrum vs Gestión de proyectos tradicional

Scrum y la mayoría de los métodos tradicionales de gestión de proyecto definen el *risk* como evento(s) inciertos que podrían afectar de forma positiva o negativa los logros de los objetivos del proyecto. Además, los *risks* son identificados, evaluados, planificados y comunicados continuamente.

En los modelos tradicionales de gestión de *projects*, se hace hincapié en la planificación por adelantada detallada para identificar, evaluar y determinar las respuestas de riesgo para todos los *risks* del *project*. Durante la ejecución del *project*, cualquier miembro del equipo del *project* puede identificar los *risks* y el director del *project* o de la oficina de gestión de *projects* o el personal de apoyo a los *projects* puede actualizarlos en el *Risk Log* o *Risk Register*. El director del *project* trata de controla todos los *risks* y por lo general identifica a individuos específicos en el equipo para que se hagan cargo de diferentes aspectos de *risks*.

En Scrum, cualquier miembro del *Equipo Scrum* puede identificar *risks* y el *Propietario del producto* puede actualizar los *risks* identificados en el *Risk Adjusted Prioritized Product Backlog*. Los principios de Scrum de *Control del proceso empírico* y el desarrollo iterativo le permiten al *Equipo Scrum* mantener constantemente la identificación de *risks* y agregarlos al *Prioritized Product Backlog*, donde tales *risk* se priorizan con otros *User Stories* existentes en el *backlog* para ser mitigados en los *Sprints* siguientes. El *Equipo Scrum* tiene responsabilidades colectivas para la gestión de todos los *risks* del *Sprint*.

8. INICIAR

Este capítulo incluye los procesos relacionados con la iniciación de un *project*: *Crear la visión del proyecto*, *Identificar al Scrum Master y al socio(s)*, *Formación de un equipo Scrum*, *Desarrollo de épica(s)*, *Creación de la lista priorizada de pendientes del producto*, y *Realizar el plan de lanzamiento*.

Initiate, tal como se define en la *Una guía para el conocimiento de Scrum (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolio, programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se entregarán a los *Socios*
- *Projects* de cualquier tamaño y complejidad

El término "*product*" (*product*) en la *Guía SBOK™* puede referirse a un *product*, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier *project* en cualquier industria-desde pequeños *projects* o equipos con tan sólo seis miembros del equipo, hasta *projects* grandes y complejos que cuentan con cientos de miembros por equipo.

8

A fin de facilitar la mejor aplicación del marco de Scrum, en este capítulo se identifican las entradas, herramientas y salidas de cada proceso, ya sea como "obligatoria" u "opcional". Las entradas, herramientas y salidas indicadas por asteriscos (*) son obligatorias, mientras que las que no tienen asteriscos son opcionales.

Se recomienda que el *Equipo Scrum* y aquellas personas que recién inician el aprendizaje sobre el marco y los procesos Scrum se centren principalmente en las aportaciones obligatorias, las herramientas y los productos; mientras que los *Propietario del producto*, *Scrum Masters*, y otros practicantes de Scrum con ya más experiencia sobre éste se esfuerzen por alcanzar un conocimiento más profundo de la información de todo el capítulo. También es importante darse cuenta de que, aunque todos los procesos se definen de forma única en la *Guía SBOK™*, no se lleva a cabo de forma secuencial o por separado necesariamente. A veces, puede ser más apropiado combinar algunos procesos, dependiendo de los requisitos específicos de cada *project*.

Este capítulo está escrito desde la perspectiva de un *Equipo Scrum* que está trabajando en un *Sprint* para producir *Entregables (Deliverables)* como parte de un *project* más amplio. Sin embargo, la información que se describe es igualmente aplicable a *projects* completos, *programs* y *portfolios*. Información adicional relacionada con el uso de Scrum para *projects*, *programs* y *portfolios* está disponible desde el capítulo 2 hasta el 7, los que cubren los principios y aspectos de Scrum.

La figura 8-1 proporciona una visión general de los procesos de *Initiate phase*, que son los siguientes:

8.1 Crear la visión del proyecto—En este proceso, el *Caso de Negocio del Proyecto* es revisado para crear un *Declaración de la Visión del Proyecto* que servirá de inspiración y proporcionará un enfoque de todo el *project*. El *Propietario del producto* se identifica en este proceso.

8.2 Identificar al Scrum Master y a los stakeholder(s)—En este proceso, el *Scrum Master* y los *socios* se identifican utilizando criterios de selección específicos.

8.3 Formación de un equipo Scrum—En este proceso, se seleccionan a los miembros del *Equipo Scrum*. Normalmente, el *Propietario del producto* es el responsable principal de la selección de los miembros del equipo, pero a menudo lo hace en *Colaboración con el Equipo Scrum*.

8.4 Desarrollo de épica(s)—En este proceso, el *Declaración de la Visión del Proyecto* sirve como la base para el desarrollo de *epics*. *User Group Meetings* pueden tomar lugar para discutir el/los *Epic(s)* apropiado(s).

8.5 Creación de la lista priorizada de pendientes del producto—En este proceso, los *Epic(s)* son refinados, elaborados, y luego priorizados para crear el *Prioritized Product Backlog* del proyecto. Lo que se conoce como *Done Criteria* también se establece en este punto.

8.6 Realizar el plan de lanzamiento—En este proceso, el *Scrum Core Team* revisa los *User Stories* en el *Prioritized Product Backlog* para desarrollar un *Release Planning Schedule*, que es esencialmente un *program* de implementación por fases que se puede compartir con los *socios del project*. Los *Length of Sprints* también se determinan en este proceso.

<p>8.1 Crear la Visión del Proyecto</p> <p>INPUTS</p> <ol style="list-style-type: none"> 1. Caso de Negocio del Proyecto* 2. Propietario del Producto del Programa 3. Scrum Master del Programa 4. Socio(s) del Programa 5. Jefe Propietario del Producto 6. Lista de Pendientes del Producto del Programa 7. Proyecto de Prueba 8. Prueba del Concepto 9. Visión de la Empresa 10. Misión de la Empresa 11. Estudio del Mercado 12. Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> 1. Reunión de la Visión del Proyecto* 2. Sesiones JAD (Diseño de Aplicación Conjunta) 3. Análisis SWOT 4. Análisis de Brechas <p>OUTPUTS</p> <ol style="list-style-type: none"> 1. Propietario del Producto identificado* 2. Declaración de la Visión del Proyecto* 3. Acta de Constitución del Proyecto 4. Presupuesto del Proyecto 	<p>8.2 Identificar al Scrum Master y al Socio(s)</p> <p>INPUTS</p> <ol style="list-style-type: none"> 1. Propietario del Producto* 2. Declaración de la Visión del Proyecto* 3. Propietario del Producto del Programa 4. Scrum Master del Programa 5. Jefe Propietario del Producto 6. Jefe Scrum Master 7. Socio(s) del Programa 8. Requisitos del personal 9. Disponibilidad y compromiso de los Personajes o Personas 10. Matriz Organizacional de Recursos 11. Matriz de las Destrezas Requeridas 12. Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> 1. Criterio de Selección* 2. Asesoramiento de Expertos de RH (Recursos Humanos) 3. Capacitación y Costos de Capacitación 4. Costos de Recursos <p>OUTPUTS</p> <ol style="list-style-type: none"> 1. Scrum Master Identificado* 2. Socio(s) Identificado* 	<p>8.3 Formar el Equipo Scrum</p> <p>INPUTS</p> <ol style="list-style-type: none"> 1. Propietario del Producto* 2. Scrum Master* 3. Declaración de la Visión del Proyecto* 4. Jefe Propietario del Producto 5. Requisitos de personal 6. Disponibilidad y compromiso de los Personajes o Personas 7. Matriz de Recurso Organizacional 8. Matriz de las Destrezas Requeridas 9. Requerimientos de Recursos 10. Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> 1. Selección del Equipo* 2. Asesoramiento de Expertos de RH (Recursos Humanos) 3. Costos del personal 4. Capacitación y Costos de Capacitación 5. Costos de Recursos <p>OUTPUTS</p> <ol style="list-style-type: none"> 1. Equipo Scrum identificado* 2. Substitutos 3. Plan de Colaboración 4. Plan para la Formación del Equipo
<p>8.4 Desarrollo de Épica(s)</p> <p>INPUTS</p> <ol style="list-style-type: none"> 1. Principal Equipo Scrum* 2. Declaración de la Visión del Proyecto* 3. Socio(s) 4. Lista de Pendientes del Producto del Programa 5. Solicitudes de Cambios Aprobados 6. Solicitudes de Cambios Rechazados 7. Riesgos de la Cartera y el Programa 8. Leyes y Regulaciones 9. Contratos aplicables 10. Información del Proyecto Previo 11. Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> 1. Reunión del Grupo de Usuarios* 2. Talleres de Historias del Usuario 3. Reunión del Grupo de Enfoque 4. Entrevista del Usuario o Cliente 5. Cuestionarios 6. Técnicas de Identificación de Riesgos 7. Experiencia del Cuerpo de Asesoramiento de Scrum <p>OUTPUTS</p> <ol style="list-style-type: none"> 1. Épica(s)* 2. Personajes o Personas* 3. Cambios aprobados 4. Riesgos identificados 	<p>8.5 Crear una Lista Priorizada de Pendientes del Producto</p> <p>INPUTS</p> <ol style="list-style-type: none"> 1. Principal Equipo Scrum* 2. Épica(s)* 3. Personajes o Personas* 4. Socio(s) 5. Declaración de la Visión del Proyecto 6. Lista de Pendientes del Producto del Programa 7. Requisitos de Negocio 8. Solicitudes de Cambios Aprobados 9. Riesgos Identificados 10. Contratos Aplicables 11. Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> 1. Métodos de Priorización de Historias del Usuario* 2. Talleres de Historias del Usuario 3. Planificación de Valor 4. Técnicas de Evaluación del Riesgo 5. Estimación del Valor del Proyecto 6. Métodos de Estimación de Historias del Usuario 7. Experiencia del Cuerpo de Asesoramiento de Scrum <p>OUTPUTS</p> <ol style="list-style-type: none"> 1. Lista Priorizada de Pendientes del Producto* 2. Criterio de Terminado* 	<p>8.6 Realizar la Planificación del Lanzamiento</p> <p>INPUTS</p> <ol style="list-style-type: none"> 1. Principal Equipo Scrum* 2. Socio(s)* 3. Declaración de la Visión del Proyecto* 4. Lista Priorizada de Pendientes del Producto* 5. Criterio de Terminado* 6. Propietario del Producto del Programa 7. Scrum Master del Programa 8. Jefe Propietario del Producto 9. Lista de Pendientes del Producto del Programa 10. Requisitos del Negocio 11. Calendario de Días Festivos 12. Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> 1. Sesiones de Planificación del Lanzamiento* 2. Métodos de Priorización del Lanzamiento* <p>OUTPUTS</p> <ol style="list-style-type: none"> 1. Cronograma de Planificación del Lanzamiento* 2. Duración del Sprint* 3. Clientes meta para el Lanzamiento 4. Lista Priorizada de Pendientes del Producto Refinada

Figura 8-1: Información general sobre *Initiate*

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

La figura 8-2 muestra las entradas obligatorias, herramientas y salidas para los procesos en *Initiate Phase*.

8.1 Crear la visión del proyecto	8.2 Identificar al Scrum Master y al socio(s)	8.3 Formación de un equipo Scrum
ENTRADAS <ul style="list-style-type: none"> 1. Caso de Negocio del Proyecto* HERRAMIENTAS <ul style="list-style-type: none"> 1. Reunión de la Visión del Proyecto* SALIDAS <ul style="list-style-type: none"> 1. Identified Propietario del producto* 2. Reunión de la Visión del Proyecto * 	ENTRADAS <ul style="list-style-type: none"> 1. Propietario del producto* 2. Reunión de la Visión del Proyecto* HERRAMIENTAS <ul style="list-style-type: none"> 1. Selection Criteria* SALIDAS <ul style="list-style-type: none"> 1. Identified Scrum Master* 2. Identified Stakeholder(s)* 	ENTRADAS <ul style="list-style-type: none"> 1. Propietario del producto* 2. Scrum Master* 3. Reunión de la Visión del Proyecto * HERRAMIENTAS <ul style="list-style-type: none"> 1. Equipo Scrum Selection* SALIDAS <ul style="list-style-type: none"> 1. Identified Equipo Scrum*
8.4 Desarrollo de épica(s)	8.5 Creación de la lista priorizada de pendientes del producto	8.6 Realizar el plan de lanzamiento
ENTRADAS <ul style="list-style-type: none"> 1. Scrum Core Team* 2. Declaración de la Visión del Proyecto* HERRAMIENTAS <ul style="list-style-type: none"> 1. User Group Meetings* SALIDAS <ul style="list-style-type: none"> 1. Epic(s)* 2. Personas* 	ENTRADAS <ul style="list-style-type: none"> 1. Scrum Core Team* 2. Epic(s)* 3. Personas* HERRAMIENTAS <ul style="list-style-type: none"> 1. User Story Prioritization Methods* SALIDAS <ul style="list-style-type: none"> 1. Prioritized Product Backlog* 2. Done Criteria* 	ENTRADAS <ul style="list-style-type: none"> 1. Scrum Core Team* 2. Socios* 3. Declaración de la Visión del Proyecto* 4. Prioritized Product Backlog* 5. Done Criteria* HERRAMIENTAS <ul style="list-style-type: none"> 1. Release Planning Sessions* 2. Release Prioritization Methods* SALIDAS <ul style="list-style-type: none"> 1. Release Planning Schedule* 2. Length of Sprint*

Figura 8-2: Información general de *Initiate* (Esenciales)

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

8.1 Crear la visión del proyecto

La figura 8-3 muestra todas las entradas, las herramientas y las salidas para el proceso de *Crear la visión del proyecto*.

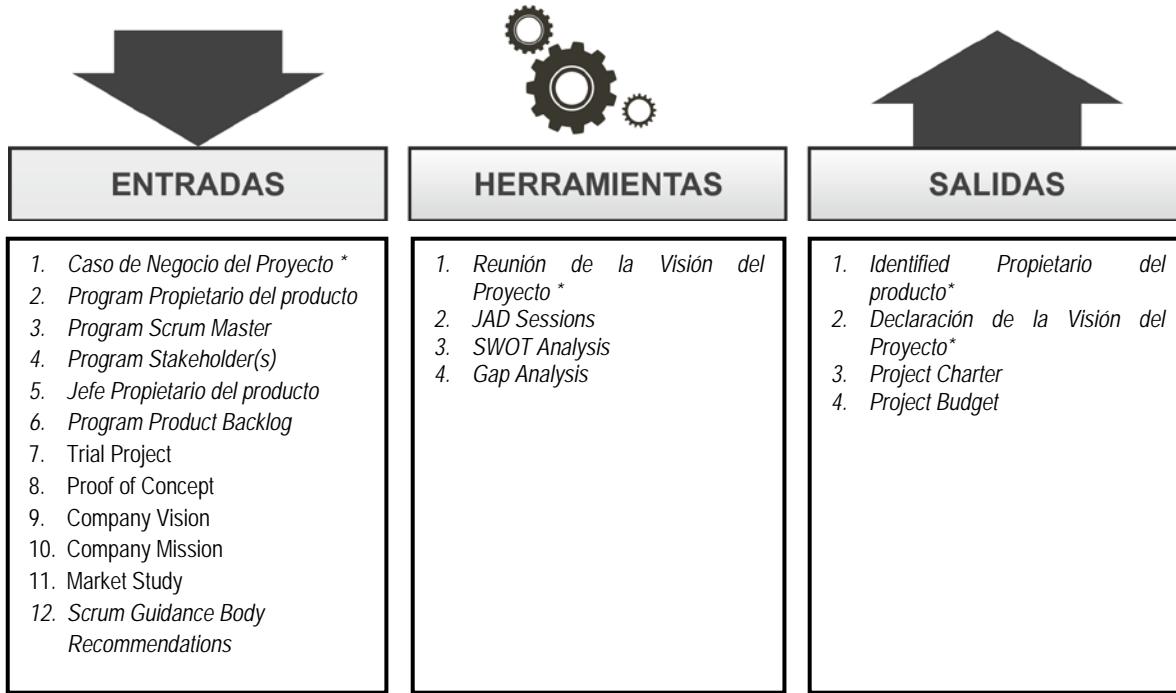
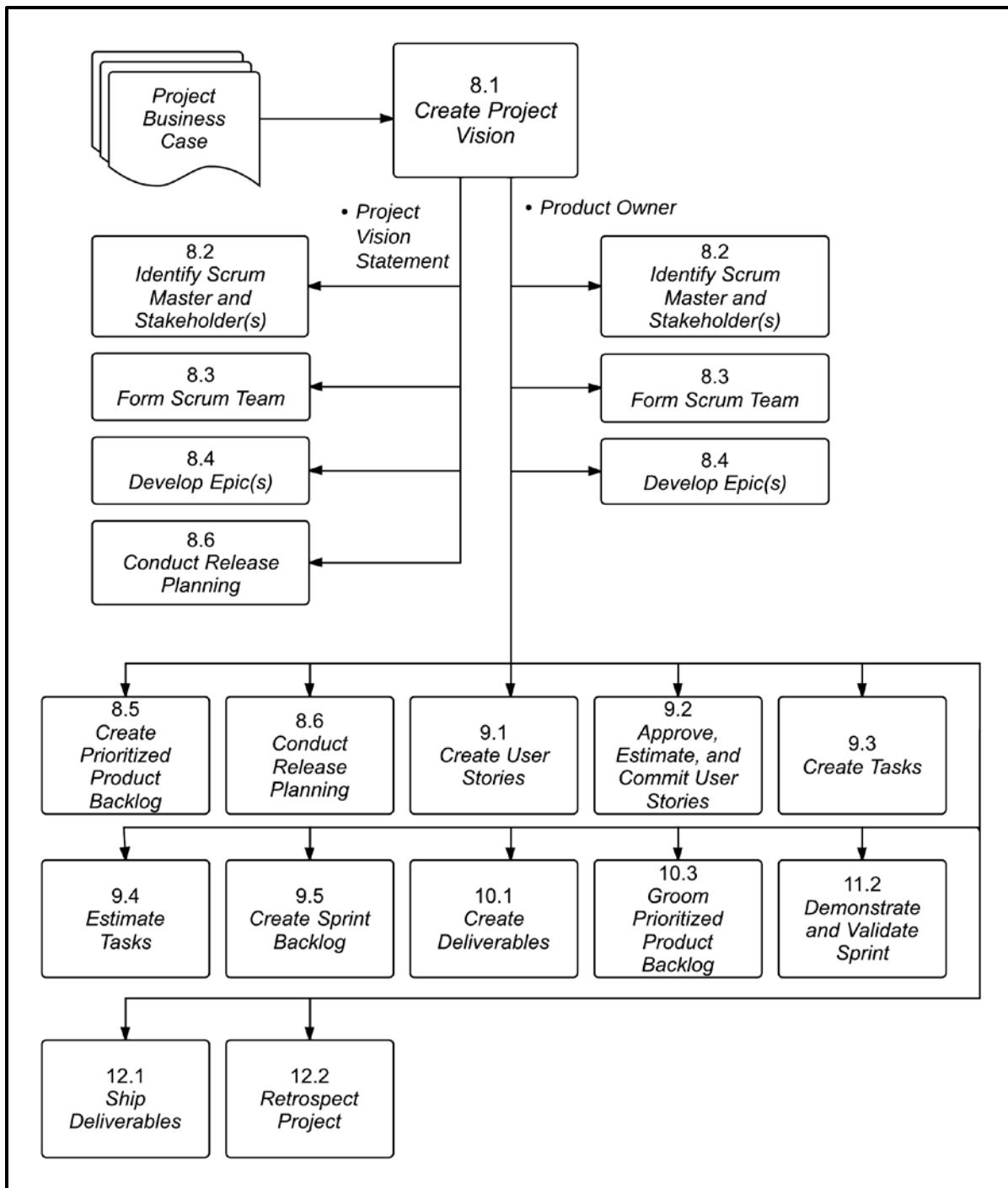


Figura 8-3: *Crear la visión del proyecto* – Entradas, Herramientas y Salidas

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

Figura 8-4: El diagrama de flujos de datos de *Crear la visión del proyecto*

8.1.1 Entradas

8.1.1.1 Caso de Negocio del Proyecto*

Un caso de negocio (*business case*) puede ser un documento bien estructurado o simplemente una declaración verbal que expresa la razón para iniciar un *project*. Puede ser formal y detallado, o informal y breve. Independientemente del formato, a menudo incluye información sustancial sobre los antecedentes del *project*, los objetivos del negocio y los resultados deseados, un *SWOT* y *Gap Analysis*, una lista de los *risks* identificados, y las estimaciones de tiempo, el esfuerzo y costo.

El *project* se inicia con la presentación del *Caso de Negocio del Proyecto*. Un caso de negocio se le presenta a los *socios* y patrocinadores (*sponsors*). Los *socios* así comprenden los beneficios de negocio esperados de tal *project* y los patrocinadores confirman que van a proporcionar los recursos financieros para el *project*.

8.1.1.2 Programa Propietario del producto

8

El *Programa Propietario del producto* es la persona responsable de maximizar el valor del negocio para un *program*. Él/ella es responsable de la articulación de requisitos de los *customers* y de mantener *Justificación del negocio* para el *program* y puede aportar de forma importante sobre qué *projects* deben ser implementados, y de que manera, en un *program*. El *Programa Propietario del producto* también administra el *Program Product Backlog*.

El *Programa Propietario del producto* interface con el *Portfolio Program Owner* para asegurar la alineación del programa con las metas y objetivos del *portfolio*. Él/ella también tiene que ver con el nombramiento de los *Propietario del producto(s)* para los proyectos individuales, y debe asegurar que la visión, los objetivos, los resultados, y las liberaciones de los *projects* individuales en el *program* estén alineados con los del *program*.

8.1.1.3 Program Scrum Master

El *Program Scrum Master* es un facilitador que asegura que todos los equipos del *project* en el *program* dispongan de un entorno propicio para completar con éxito sus *projects*. El *Program Scrum Master* guía, facilita y les enseña las prácticas de Scrum a todos los que participan en el *program*; también sirve como guía para los *Scrum Masters* de los distintos *projects*; elimina los *impediments* de los diferentes equipos del *project*; coordina con el *Cuerpo de asesoramiento de Scrum* para definir los objetivos relacionados con la normativa de calidad, el gobierno, la seguridad y otros parámetros claves de la organización; y asegura que los procesos de Scrum se estén siguiendo de manera efectiva a lo largo del *program*.

El *Program Scrum Master* debe interactuar con el *Portfolio Scrum Master* para seguir la alineación del *program* con las metas y objetivos del *portfolio*. También tiene que ver con el nombramiento del *Scrum*

Masters para *projects* individuales y debe asegurar que la visión, los objetivos, los resultados, y las liberaciones de los *projects* individuales en el *program* estén alineados con los del *program*.

8.1.1.4 Program Stakeholder(s)

Program Stakeholder(s) es un término colectivo que incluye a los *customers*, los usuarios y patrocinadores para un *program*. Influyen en todos los *projects* del *program* durante todo el desarrollo del *project*. El/Los *Program Stakeholder(s)* también puede(n) ayudar a definir la visión del *project* y a proporcionar orientación en relación con el valor del negocio.

El/Los *Program Stakeholder(s)* debe(n) interactuar con los *Portfolio Socios* para asegurar la alineación del *program* con las metas y objetivos del *portfolio*. Ellos también están involucrados con el nombramiento del/de los *Stakeholder(s)* para *projects* individuales y se aseguran que la visión, los objetivos, los resultados, y los lanzamientos de los *projects* individuales en el *program* estén alineados con los del *program*.

8.1.1.5 Jefe Propietario del producto

En el caso de grandes *projects* con numerosos *Equipos Scrum*, podría ser necesario tener un *Jefe Propietario del producto*. Esta función se encarga de coordinar el trabajo de los múltiples *Propietario del producto*. El *Jefe Propietario del producto* prepara y mantiene el *Prioritized Product Backlog* para los *projects* grandes, usándolo para coordinar el trabajo a través de los *Propietario del producto* de los *Equipos Scrum*. Los *Propietario del producto*, a su vez, gestionan sus respectivas partes del *Prioritized Product Backlog*.

El *Jefe Propietario del producto* también se comunica con el *Program Propietario del producto* para asegurar la alineación de los *projects* grandes con las metas y objetivos del *program*.

8.1.1.6 Program Product Backlog

El *Program Propietario del producto* desarrolla el *Program Product Backlog* del *product* que contiene una lista de prioridades de negocios de alto nivel y los requisitos del *project* escritos preferiblemente en forma de *Program Backlog Items* del *program*. Luego, estos son refinados por los *Propietario del producto* de los *projects* individuales, ya que crean y dan prioridad a pedidos pendientes del *product* para sus *projects*. Estos *Prioritized Product Backlogs* tienen *User Stories* más pequeños pero detallados que pueden ser aprobados, estimados, y aplicados por miembros del *Equipo Scrum*.

El *Program Propietario del producto* mantiene de forma continua el *Program Product Backlog* para garantizar que los nuevos *business requirements* se añadan y que los requisitos existentes estén debidamente documentados y priorizados. Esto asegura que a los requisitos más valiosos en el

cumplimiento de los objetivos del *program* se les esté dando alta prioridad y que los restantes reciban una prioridad más baja.

El *Program Product Backlog* creado para el *program* presenta una imagen más grande de todos los *projects* que forman parte del *program*. Por lo tanto, puede servir de orientación en relación con los objetivos del *project*, el alcance, los objetivos y los beneficios esperados del negocio.

8.1.1.7 Trial Project

Si es factible, un demo o prueba del *project* en pequeña escala podría ser ejecutado como un experimento para predecir y evaluar la viabilidad, tiempo, costo, *risks*, y los posibles efectos del *project*. Esto ayuda a evaluar el entorno de práctica y guías del diseño del *project* con anterioridad a la iniciación del *project*.

8.1.1.8 Proof of Concept

Proof of Concept demuestra y verifica que la idea detrás del proyecto actual es potencialmente viable en el entorno real. A menudo esto se hace en la forma de un prototipo diseñado para determinar la viabilidad técnica y financiera, ayudar a comprender los requisitos, y ayudar en la evaluación de las decisiones de diseño al principio del proceso. Sin embargo, *Proof of Concept* no necesita representar necesariamente los verdaderos *Deliverables* (Entregables) del *project*.

8

8.1.1.9 Company Vision

La comprensión del *Company Vision* ayuda a que el *project* mantenga su enfoque en los objetivos de la organización y el futuro probable de la empresa. El *Propietario del producto* se puede guiar por el *Company Vision* para crear el *Declaración de la Visión del Proyecto*.

8.1.1.10 Company Mission

Company Mission ofrece un marco para la formulación de las estrategias de la empresa y orienta la toma de decisiones en general en la empresa. *Project Vision* debe enmarcarse de tal manera que su cumplimiento ayuda a la organización a llevar a cabo su misión.

8.1.1.11 Market Study

Market Study se refiere a la investigación organizada, la recopilación, la comparación y el análisis de datos relacionados con las preferencias de los *customers* sobre los *products*. A menudo incluye numerosos datos sobre las tendencias del mercado, la segmentación del mercado y los procesos de comercialización.

Estudio de mercado podría incluir también un estudio analítico de los competidores que proporciona una mejor comprensión de las fortalezas y debilidades de los competidores y puede ayudar a los que toman decisiones a formular productos mejor posicionados.

8.1.1.12 Recomendaciones del *Cuerpo de asesoramiento de Scrum*

El *Cuerpo de asesoramiento de Scrum* (*SGB*) es una función opcional. Por lo general, se compone de un grupo de documentos y/o un grupo de expertos que normalmente están involucrados en la definición de objetivos relacionados con la calidad, las regulaciones gubernamentales, la seguridad y otros parámetros claves de la organización. Estos objetivos guían la labor llevada a cabo por el *Propietario del producto*, *Scrum Master*, y el *Equipo Scrum*. El *Cuerpo de asesoramiento de Scrum* también ayuda a capturar las mejores prácticas que se deben utilizar en todos los *projects Scrum* en la organización.

El *Cuerpo de asesoramiento de Scrum* no toma decisiones relacionadas con el *project*. Este actúa como una consultoría o una estructura de orientación para todos los niveles de jerarquía en el *project* de organización del *portfolio*, *program* y *project*. Los *Equipos Scrum* tienen la opción de pedirle asesoramiento al *Cuerpo de asesoramiento de Scrum*.

Es importante asegurarse de que la visión del *project* esté alineada con las recomendaciones proporcionadas por el *Cuerpo de asesoramiento de Scrum* y que los procesos cumplan con las normas y directrices establecidas por el *Body* (la Administración).

8.1.2 Herramientas

8.1.2.1 *Reunión de la Visión del Proyecto* *

Reunión de la Visión del Proyecto es una reunión con el/los *Program Stakeholder(s)*, *Program Propietario del producto*, *Program Scrum Master*, y *Jefe Propietario del producto*. Ayuda a identificar el contexto empresarial, *business requirements* y las expectativas de los *socios* con el fin de desarrollar una *Declaración de la Visión del Proyecto* eficaz. Scrum cree en la participación y colaboración cercana con todos los representantes de las empresas para obtener su *buy-in* (*convencimiento de su importancia*) del *project* y para ofrecer un valor más significativo.

8.1.2.2 *JAD Sessions*

Joint Application Design (JAD) Session es una técnica de recopilación de requisitos. Se trata de un taller facilitado altamente estructurado que acelera el proceso de *Create Project Vision*, ya que permite al/a los *stakeholder(s)* y a otros que toman decisiones llegar a un consenso sobre el alcance, los objetivos, y otras especificaciones del *project*.

Esta técnica consiste en métodos para aumentar la participación del usuario, lo que acelera el desarrollo y la mejora de las especificaciones. Los *Relevant Program Stakeholder(s)*, *Program Propietario del producto*,

Program Scrum Master y Jefe Propietario del producto podrían reunirse para delinear y analizar los resultados de negocio deseados y visualizar su visión para el *project Scrum*.

8.1.2.3 SWOT Analysis

SWOT es un enfoque estructurado para la planificación que ayuda a evaluar los/las *Strengths*, *Weaknesses*, *Opportunities* y *Threats* (*puntos fuertes y débiles, oportunidades y amenazas*) relacionados con un *project*. Este tipo de análisis ayuda a identificar tanto los factores internos como externos que podrían afectar el *project*. Las fortalezas y debilidades son factores internos, mientras que *opportunities* y amenazas son factores externos. La identificación de estos factores ayuda a los socios y a aquellos que toman decisiones a finalizar los procesos, las herramientas y las técnicas que se utilizarán para lograr los objetivos del *project*. La realización de un *SWOT* permite la identificación precoz de las prioridades, los cambios potenciales, y los *risks*.

8.1.2.4 Gap Analysis

Gap Analysis es una técnica que se utiliza para comparar el estado actual con el estado deseado. En una organización, se trata de determinar y documentar la diferencia entre las capacidades actuales del negocio y el conjunto final deseado de capacidades. Normalmente, se inicia un *project* para que una organización alcance una situación deseada, por lo que llevar a cabo un *Gap analysis* ayudaría a quienes toman decisiones a determinar la necesidad de un *project*.

Los principales pasos a seguir en *Gap Analysis* se presentan en la Figura 8-5.

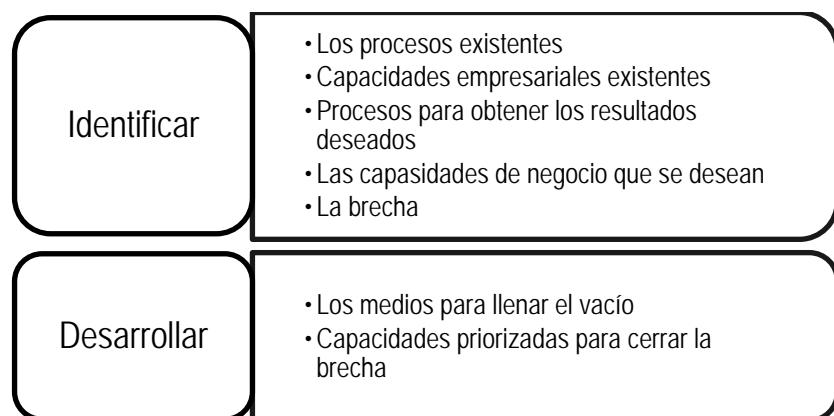


Figura 8-5: El proceso de *Gap Analysis*

8.1.3 Salidas

8.1.3.1 Identified Propietario del producto*

Uno de los resultados de este proceso es identificar al *Propietario del producto*. El *Propietario del producto* es la persona responsable de lograr el valor máximo empresarial para el *project*. Él o ella también es responsable de la articulación de requisitos por parte de los *customers* y de mantener *Justificación del negocio* para el *project*. El *Propietario del producto* representa la voz del *customer*.

Cada *Equipo Scrum* tendrá un *Propietario del producto* designado. Un pequeño *project* puede tener sólo un *Propietario del producto*, mientras que los *projects* más grandes pueden tener varios. Estos *Propietario del producto* son responsables de la gestión de sus secciones del *Prioritized Product Backlog*. Ellos escriben los *User Stories* y gestionan el mantenimiento del *Prioritized Product Backlog*.

El papel del *Propietario del producto* se describe con más detalle en la sección 3.4.

8.1.3.2 Reunión de la Visión del Proyecto *

El resultado clave del proceso *Crear la visión del proyecto* es un *Reunión de la Visión del Proyecto* bien estructurado. Una buena visión del *project* explica la necesidad del negocio, y que es lo que el *project* tiene como objetivo satisfacer, en lugar de cómo se va a satisfacer la necesidad.

El *Reunión de la Visión del Proyecto* no debería ser muy específico y debe ser flexible. Es posible que el conocimiento actual sobre el *project* esté basado en suposiciones que luego vayan a cambiar conforme avanza el *project*, por lo que es importante que la visión del *project* sea lo suficientemente flexible como para adaptarse a estos cambios. La visión del *project* debe centrarse en el problema y no la solución.

Ejemplo:

VMFoods, una cadena de supermercados, quiere ampliar con un portal de comercio electrónico en línea y se ha comunicado con su firma para crear el *product*.

Project Vision: Desarrollar una herramienta fácil de usar y un canal de ventas en línea para VMFoods que sea estéticamente agradable.

8.1.3.3 Project Charter

Un *Project Charter* es una declaración oficial de los objetivos y resultados deseados del proyecto. En varias organizaciones, el *Project Charter* es el documento que oficialmente y formalmente autoriza el *project*, dándole al equipo la autoridad por escrita para comenzar el *project*.

8.1.3.4 Project Budget

El *Project Budget* es un documento financiero que incluye el costo de las personas, materiales y otros gastos relacionados en un *project*. El *Project Budget* típicamente es firmado por el/los patrocinador(es) para asegurar que haya suficientes fondos disponibles. Una vez firmado, el *Propietario del producto* y el *Scrum Master* estarán regularmente envueltos con la gestión del *Project Budget* para garantizar que las personas y los recursos necesarios para las actividades del *project* estén disponibles.

8.2 Identificar al Scrum Master y al socio(s)

La figura 8-6 muestra todas las entradas, las herramientas y las salidas para el proceso *Identificar al Scrum Master y al socio(s)*

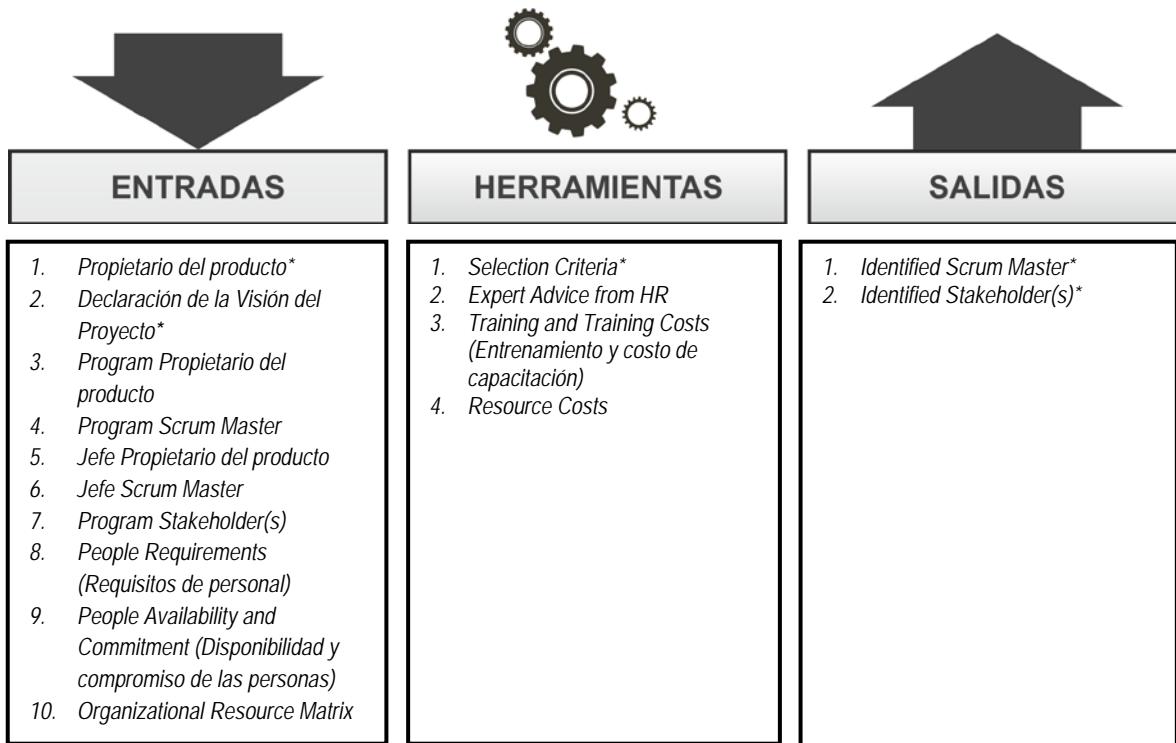


Figura 8-6: *Identificar al Scrum Master y al socio(s)*—Entradas, Herramientas y Salidas

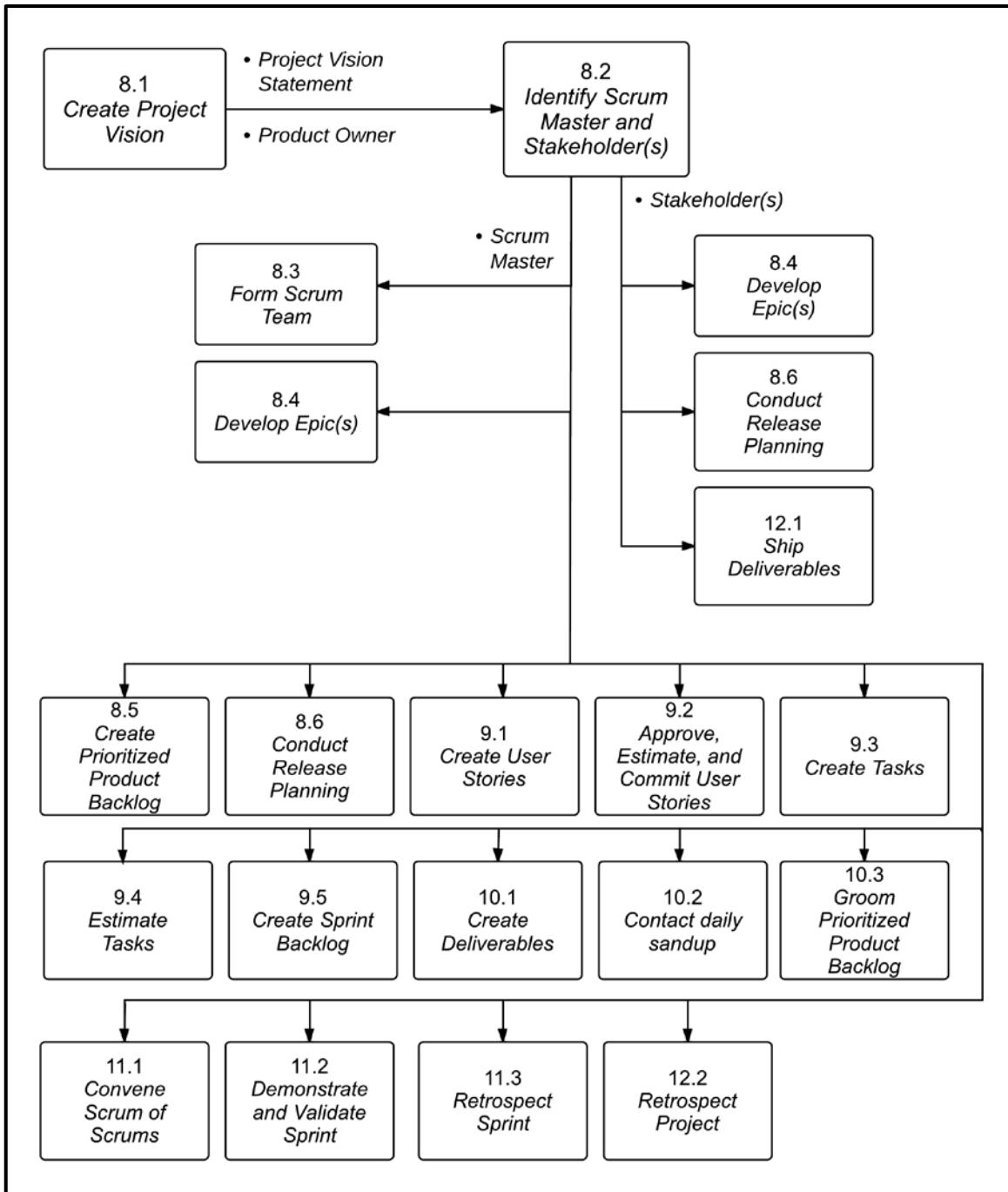


Figura 8-7: Identificar al Scrum Master y al socio(s)—Diagrama de flujo de datos

8.2.1 Entradas

8.2.1.1 *Propietario del producto**

Descrito en la sección 8.1.3.1.

8.2.1.2 *Reunión de la Visión del Proyecto **

Descrito en la sección 8.1.3.2.

8.2.1.3 *Program Propietario del producto*

Descrito en la sección 8.1.1.2.

8

8.2.1.4 *Program Scrum Master*

Descrito en la sección 8.1.1.3.

8.2.1.5 *Jefe Propietario del producto*

Descrito en la sección 8.1.1.5.

8.2.1.6 *Jefe Scrum Master*

Los grandes proyectos requieren que múltiples *Scrum Masters* trabajen en paralelo. Es posible que la información obtenida por un equipo se le deba comunicar adecuadamente a los otros equipos – es el *Jefe Scrum Master* el responsable de esta actividad.

La coordinación entre distintos *Equipos Scrum* que colaboran en un *project* se realiza normalmente a través del *Scrum of Scrums Meeting (SoS)* (véase la sección 3.7.2.1). Esto es análogo al *Daily Standup Meeting* y es facilitado por el *Jefe Scrum Master*. El *Jefe Scrum Master* típicamente es el responsable de atender los *impediments* que más afectan a un *Equipo Scrum*.

8.2.1.7 *Program Stakeholder(s)*

Descrito en la sección 8.1.1.4.

8.2.1.8 *People Requirements*

People Requirements es uno de los pasos iniciales en la selección del *Scrum Master* y *Stakeholder(s)*. Es importante documentar las funciones y responsabilidades de todos los que se verán involucrados en la realización de las tareas del *project*. Esto incluye a todas las personas involucradas en el *project*, en cualquier calidad, independientemente de que su papel sea básico o no.

Por lo general, el *Propietario del producto* o el *Scrum Master* colaboran con el *Human Resource Department* de la empresa para determinar y concluir los *People Requirements*.

8.2.1.9 Disponibilidad y compromiso de las personas

Antes de seleccionar al *Scrum Master* y al/ a los *stakeholder(s)*, se debe confirmar su disponibilidad. Sólo los miembros del equipo que estarán disponibles y que puedan comprometerse plenamente con el *project* deben ser seleccionados. En *People Availability and Commitment* es representado comúnmente en forma de calendario que muestra cuando los recursos humanos estarán disponibles para trabajar durante toda la duración del *project*.

Para ser eficaz, los *Equipos Scrum* idealmente deberían tener de seis a diez miembros; y la sustitución de las personas o el cambio de los miembros del equipo no son aconsejables en un *Scrum Core Team*. Por lo tanto, es importante contar con personas en un *Scrum Core Team* que estén disponibles y plenamente comprometidas con el *project*.

8.2.1.10 *Organizational Resource Matrix*

Organizational Resource Matrix es una representación jerárquica de una combinación de una estructura de organización funcional y una estructura organizativa proyectizada. Las organizaciones matriciales reúnen a miembros de varios equipos del *project* de diferentes departamentos funcionales, tales como alguien de la tecnología, las finanzas, marketing, ventas, manufactura, y otros departamentos – para así crear equipos multifuncionales.

Los miembros del equipo en una organización matricial cumplen dos objetivos - funcionales y de *project*. Los miembros del equipo son dirigidos por el/los *Propietario del producto(s)* con respecto a las actividades de *projects*, mientras que los directores funcionales realizan actividades de gestión relacionadas con sus departamentos, como las evaluaciones de desempeño y la aprobación de ausencia.

8.2.1.11 Skills Requirement Matrix

Skills Requirement Matrix también conocido como un marco de competencias, se utiliza para evaluar las carencias de habilidades y los requisitos de formación para los miembros del equipo. Una matriz de habilidades asigna las habilidades, las capacidades y el nivel de interés de los miembros del equipo en el uso de esas habilidades y capacidades en un *project*. Al utilizar esta matriz, la organización puede evaluar los vacíos de habilidades en los miembros del equipo e identificar a los empleados que van a necesitar más formación en un área o competencia particular.

8.2.1.12 Cuerpo de asesoramiento de Scrum Recommendations

Descrito en la sección 8.1.1.12.

8.2.2 Herramientas

8

8.2.2.1 Criterios de selección*

La selección adecuada de *Scrum Master(s)* y la identificación del/de *stakeholder(s)* pertinente(s) es crucial para el éxito de cualquier *project*. En algunos *projects*, puede ser que haya habido pre-condiciones que estipulen determinados miembros del equipo y sus roles.

Cuando hay flexibilidad en la elección del/de los *Scrum Master(s)*, se deben considerar los siguientes *Selection Criteria*:

1. *Habilidades para la resolución de problemas*—Es uno de los principales criterios a considerar al seleccionar al/a los *Scrum Master(s)*. El/los *Scrum Master(s)* debe(n) tener las habilidades y experiencia necesarias para ayudar a eliminar cualquier *impediment* que encare el *Equipo Scrum*.
2. *Disponibilidad*—El *Scrum Master* debe estar disponible para programar, supervisar y facilitar varias reuniones, incluyendo *Release Planning Meeting*, *Daily Standup Meeting*, y otras reuniones relacionadas al *Sprint*.
3. *Compromiso*—El *Scrum Master* se debe comprometer a que el *Equipo Scrum* esté dotado de un ambiente de trabajo propicio para asegurar la entrega exitosa de los *products* *Scrum*.
4. *Servant Leadership Style*—Para más detalles, consulte la sección 3.10.4.1

En la identificación del/de los *stakeholder(s)*, es importante recordar que el/los *socios* incluye(n) a todos los *customers*, los usuarios y patrocinadores, quienes a menudo interactúan con el *Propietario del producto*, *Scrum Master* y *Equipo Scrum* para proveer entradas y facilitar la creación de los *products* del *project*. Los *socios* influyen el *project* a lo largo de su ciclo de vida.

8.2.2.2 *Expert Advice from HR*

Expert Advice from HR puede ser útil en la identificación del *Scrum Master* y de los *Stakeholder(s)*. El departamento de Recursos Humanos posee un conocimiento específico sobre los empleados de una organización y las diversas técnicas que pueden ayudar a identificar a un *Scrum Master* como a un *Stakeholder*.

8.2.2.3 Formación y costos de capacitación

Scrum es un marco radicalmente diferente a los métodos tradicionales de gestión de *projects*. Los miembros del equipo no siempre pueden poseer los conocimientos o habilidades necesarias para trabajar en el entorno de Scrum. El *Propietario del producto* debería evaluar las necesidades de capacitación de los miembros potenciales del equipo y facilitar la formación para eliminar tal carencia. El *Propietario del producto* es normalmente responsable de la evaluación y la selección de los miembros del equipo, pero a menudo lo hace en consulta con el *Scrum Master* que puede tener un conocimiento adicional de los recursos de los que trabajan con ellos en otros *projects*.

Una formación adecuada se le debe proporcionar a los miembros del *Equipo Scrum*, tanto antes del inicio de los trabajos y también mientras están trabajando en sus *projects*. Los miembros del *Equipo Scrum* también deben estar dispuestos a aprender de los demás, y de quienes tienen más experiencia en el equipo.

8.2.2.4 Costo de los recursos

Una de las principales consideraciones en la selección de las personas tiene que ver con las ventajas y desventajas relacionadas con el nivel de experiencia en comparación al salario. Hay otros factores relacionados a las personas que influyen en el costo que también se deben considerar. Idealmente, el/los *Scrum Master(s)*, los miembros del equipo, y el/los *stakeholder(s)* deben ser *colocated*, para que puedan comunicarse con frecuencia y facilidad. Si *colocation* no es posible y hay equipos distribuidos, se debe tener recursos adicionales para facilitar la comunicación, el entendimiento de las diferencias culturales, sincronizar el trabajo y fomentar el intercambio de conocimientos.

8.2.3 Salidas

8.2.3.1 *Identified Scrum Master**

Un *Scrum Master* es un facilitador y *servant leader* que se asegura de que el *Equipo Scrum* esté dotado de un ambiente propicio para completar el *project* con éxito. El *Scrum Master* guía, facilita y les enseña prácticas de *Scrum* a todos los involucrados en el *project*; borra *impediments* que encara el equipo; y asegura que se estén siguiendo los procesos de *Scrum*. Es la responsabilidad del *Propietario del producto* identificar al *Scrum Master* para un *project Scrum*.

El papel de *Scrum Master* se describe con más detalle en la sección 3.4

8.2.3.2 *Identified Stakeholder(s)**

El/Los *Stakeholder(s)*, que es un término colectivo que incluye a los *customers*, los usuarios y los patrocinadores, con frecuencia interactua(n) con el *Scrum Core Team* e influye(n) en el *project* durante todo el proceso de desarrollo de *products*. Es para los *socios* que el *project* produce los beneficios deseados de colaboración.

La función del/de los *stakeholder(s)* se describe en la sección 3.3.2.

8.3 Formación de un equipo Scrum

La figura 8-8 muestra todas las entradas, las herramientas y las salidas para el proceso de *Formación de un equipo Scrum*.



Figura 8-8: *Formación de un equipo Scrum—Entradas, Herramientas y Salidas*

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

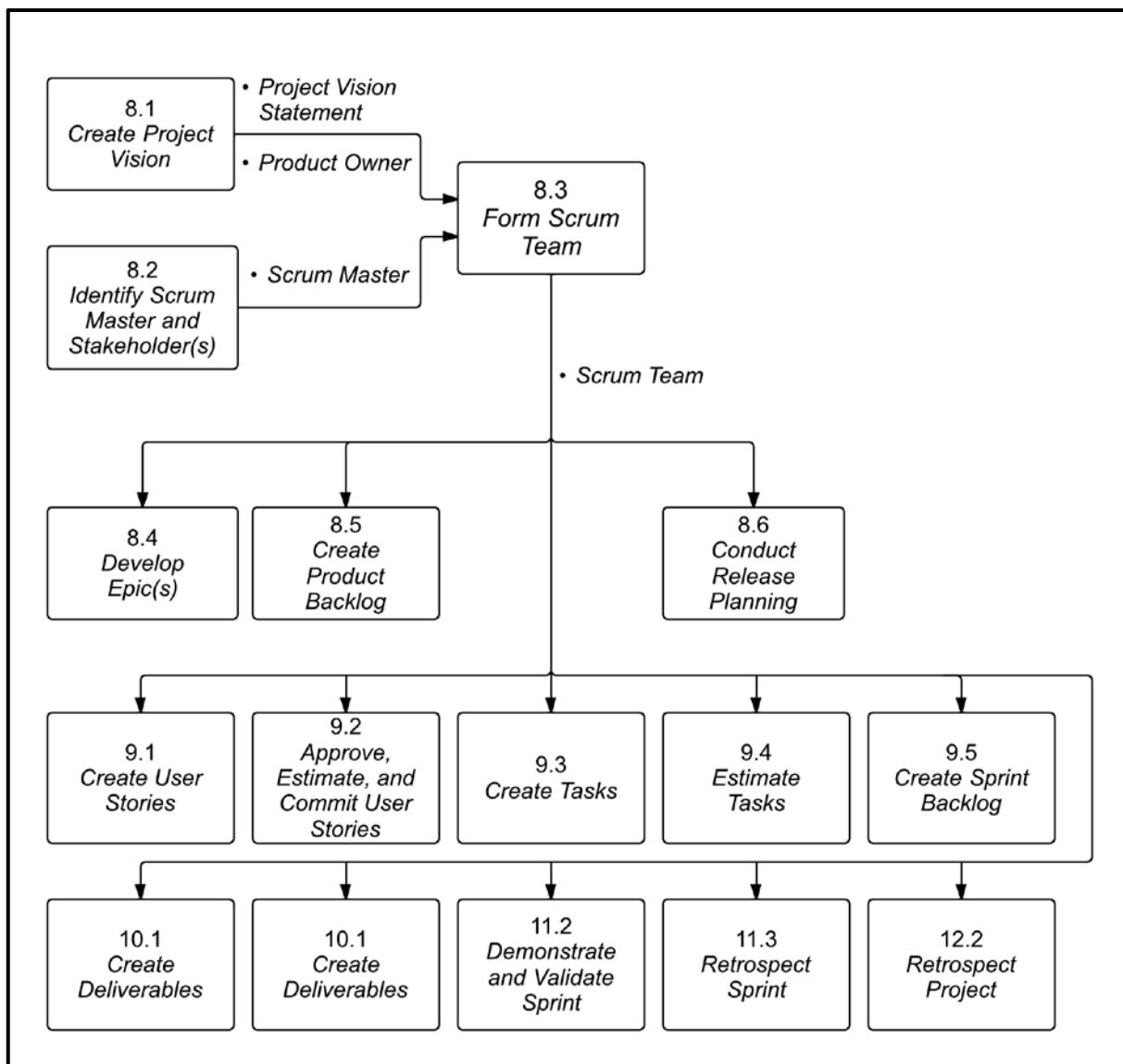


Figura 8-9: Formación de un equipo Scrum—Diagrama de flujo de datos

8.3.1 Entradas

8.3.1.1 *Propietario del producto**

Descrito en la sección 8.1.3.1.

8.3.1.2 *Scrum Master**

Descrito en la sección 8.2.3.1.

8.3.1.3 *Reunión de la Visión del Proyecto **

Descrito en la sección 8.1.3.2.

8.3.1.4 *Jefe Propietario del producto*

Descrito en la sección 8.1.1.5.

8.3.1.5 *People Requirements (Requisitos de personal)*

Descrito en la sección 8.2.1.8.

8.3.1.6 *People Availability and Commitment (Disponibilidad y compromiso de las personas)*

Descrito en la sección 8.2.1.9.

8.3.1.7 *Organizational Resource Matrix*

Descrito en la sección 8.2.1.10.

8.3.1.8 *Skills Requirement Matrix*

Descrito en la sección 8.2.1.11.

8.3.1.9 Requisitos de recursos

Estos requisitos incluyen todos los recursos, sean personas o no, necesarios para que el *Equipo Scrum* funcione con eficacia. Estos recursos incluyen infraestructura de oficinas, espacios de reunión, los equipos de trabajo, *Scrumboards*, etc. En el caso de los equipos virtuales, recursos adicionales, tales como herramientas de *Colaboración*, videoconferencia, repositorios de documentos compartidos, servicios de traducción, etc. tienen que ser considerados.

8.3.1.10 Recomendaciones del *Cuerpo de asesoramiento de Scrum*

Descrito en la sección 8.1.1.12.

8.3.2 Herramientas

8

8.3.2.1 Selección del Equipo Scrum*

El *Equipo Scrum* es la base de cualquier *project* de Scrum y el tener a los miembros adecuados para el equipo es vital para la entrega exitosa de los *projects* Scrum. Los miembros del *Equipo Scrum* son generalistas/ especialistas ya que cuentan con conocimiento de diversos campos y son expertos en al menos uno. Más allá de la experiencia en la materia, son las habilidades interpersonales de cada miembro del equipo que determinará el éxito de los equipos auto-organizados.

Los miembros ideales del *Equipo Scrum* son independientes, auto-motivados, se enfocan en el *customer*, y tienen un sentido alto de la responsabilidad y la colaboración. El equipo debe ser capaz de fomentar un ambiente de reflexión independiente y de tomar decisiones con el fin de extraer los mayores beneficios de la estructura.

8.3.2.2 Asesoramiento experto de Recursos Humanos (*Expert Advice from HR*)

Los gerentes de *Expert Advice from Human Resource (HR)* pueden ser valiosos para la formación de un *Equipo Scrum*. El departamento de Recursos Humanos posee un conocimiento especializado sobre los empleados de una organización y de las numerosas técnicas que pueden ayudar a los *Propietario del producto*, *Scrum Masters*, y a los patrocinadores para identificar a los mejores miembros del equipo.

8.3.2.3 Costo asociado con el personal

Todos los costos asociados con los requisitos de las personas deben ser evaluados, analizados, aprobados y presupuestados.

8.3.2.4 Costos de formación y capacitación

Es posible que los miembros del equipo no cuenten con las habilidades o conocimientos necesarios para llevar a cabo tareas especializadas. En ese caso, el *Propietario del producto* debe evaluar las necesidades de capacitación de los miembros potenciales del equipo y proporcionar capacitación, cuando se ha encontrado una carencia de habilidades o conocimientos.

Para que una implementación de Scrum sea realmente eficaz, tiene que haber un alto nivel de conciencia dentro de la organización de los principios y valores de Scrum. Este conocimiento ayudará a la ejecución exitosa de Scrum. El *Equipo Scrum* tiene que estar capacitado en las prácticas de Scrum, y es el *Scrum Master* quien debe desempeñar el papel de entrenador de este tema para el equipo. Dado a que la planificación de los *Sprints* es un factor muy importante para el éxito, el entrenamiento le ayudará a los miembros del equipo a entender cómo discutir e identificar metas alcanzables en un *Sprint*. El *Scrum Master* tiene que sacar lo mejor del *Equipo Scrum* motivándolos y facilitando el proceso de desarrollo. Con el entrenamiento correcto, el *Scrum Master* puede ayudar al equipo a articular *issues* y desafíos que puedan enfrentar. Normalmente, cualquier *issue* o conflictos en el equipo se resuelven por el mismo equipo con entrenamiento y la asistencia del *Scrum Master*, según sea necesario. El *Scrum Master* debe atender los *issues* como la baja moral o falta de coordinación dentro del equipo. Él/ella es responsable de la eliminación de *impediments* para el equipo. Cuando sea necesario, el *Scrum Master* puede presentarles estos *issues* externos e *impediments* de gestión a la dirección para llegar a una resolución.

Los costos de formación y capacitación son discutidos en el proceso de *Identificar al Scrum Master y al socio(s)*, sección 8.2.2.3.

8.3.2.5 Costo de los recursos

Los costos asociados con todos los requisitos no relacionados con las personas deben ser evaluados, analizados, aprobados y presupuestados. Un recurso en el entorno del *project* es identificado como aquello que se usa para realizar una tarea o actividad, incluyendo-pero no limitado a equipos, materiales, servicios externos, y el entorno físico.

8.3.3 Salidas

8.3.3.1 Equipo Scrum identificado*

El *Equipo Scrum*, a veces conocido como *Development Team*, es un grupo o equipo de personas que son responsables de la comprensión de los *business requirements* especificados por el *Propietario del producto*, la estimación de *user stories* y la creación definitiva de los *Entregables* del *project*. Los *Equipos Scrum* son multi-funcionales y auto-organizados. El equipo decide la cantidad de trabajo que se comprometerá en un *Sprint* y determina la mejor manera de realizar el trabajo. El *Equipo Scrum* se compone de miembros multi-

funcionales, que llevan a cabo todo el trabajo involucrado en la creación de Entregables que potencialmente se puedan entregar, incluyendo el desarrollo, la prueba, *quality assurance*, etc.

La identificación del *Equipo Scrum* es la responsabilidad del *Propietario del producto*, a menudo en consulta con el *Scrum Master*.

El papel del *Equipo Scrum* se describe con más detalle en la sección 3.6.

8.3.3.2 Personas de respaldo/Substitutos

Al seleccionar a los equipos, otro aspecto importante es la creación de substitutos para cada miembro del *Equipo Scrum*. Aunque la disponibilidad y compromiso por los miembros del equipo se confirman por adelantado, pueden surgir *issues* tales como una enfermedad, emergencia familiar, o simplemente el hecho de que un miembro se marche de la organización. Los *Equipo Scrum* trabajan en pequeños grupos de seis a diez personas. El tener reemplazantes asegura que no haya una disminución importante en la productividad debido a la falta de un miembro del equipo.

8.3.3.3 Colaboración Plan

Colaboración es un elemento muy importante en Scrum. La planificación de cómo las distintas personas que toman decisiones, los *socios*, y miembros del equipo participan y colaboran entre sí es vital. *Colaboración Plan* es una salida (*output*) opcional que puede ser formal o informal. A veces, puede ser simplemente un entendimiento verbal entre los diversos *socios*, ya en que Scrum se evita toda la documentación innecesaria. Sin embargo, para los *projects* más grandes y complejos, especialmente aquellos con equipos distribuidos, puede ser necesario poner en marcha un acuerdo más formal. El plan puede abordar cómo los miembros de *Scrum Core Team* (*Equipo Principal de Scrum*), *stakeholder(s)* y otras personas involucradas en el *project* Scrum se comunican y colaboran durante todo el *project* y también puede definir las herramientas o técnicas específicas que se utilizarán para este fin. Por ejemplo, en el caso de equipos distribuidos, puede haber una necesidad de un acuerdo sobre cuándo y cómo se llevarán a cabo las reuniones, qué tipo de herramientas de comunicación se utilizará, y quién debe estar involucrado en las diversas reuniones.

8.3.3.4 Team Building Plan

Debido a que un *Equipo Scrum* es multi-funcional, cada miembro debe participar activamente en todos los aspectos del *project*. El *Scrum Master* debe identificar los *issues* con los miembros del equipo y encararlos con el fin de mantener un equipo eficaz.

Para tener cohesión en el equipo, el *Scrum Master* debe asegurarse que las relaciones entre los miembros del equipo sean positivas y que estén unificados en la consecución de la totalidad del *project* y de los objetivos de la organización, lo que conduce a una mayor eficiencia y una mayor productividad.

En este contexto, es importante estudiar la sección 3.10, que habla sobre las teorías populares de recursos humanos y su relevancia para Scrum.

8.4 Desarrollo de épica(s)

La figura 8-10 muestra todas las entradas, herramientas y salidas para el proceso de *Desarrollo de épica(s)*.

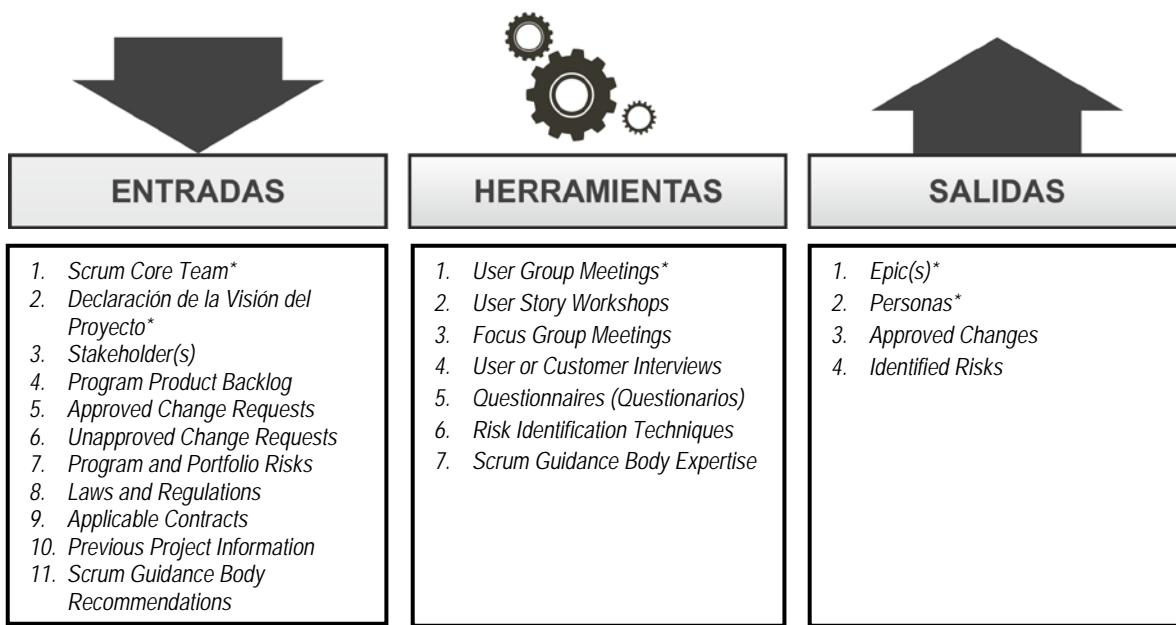


Figura 8-10: *Desarrollo de épica(s)*—Entradas, Herramientas y Salidas

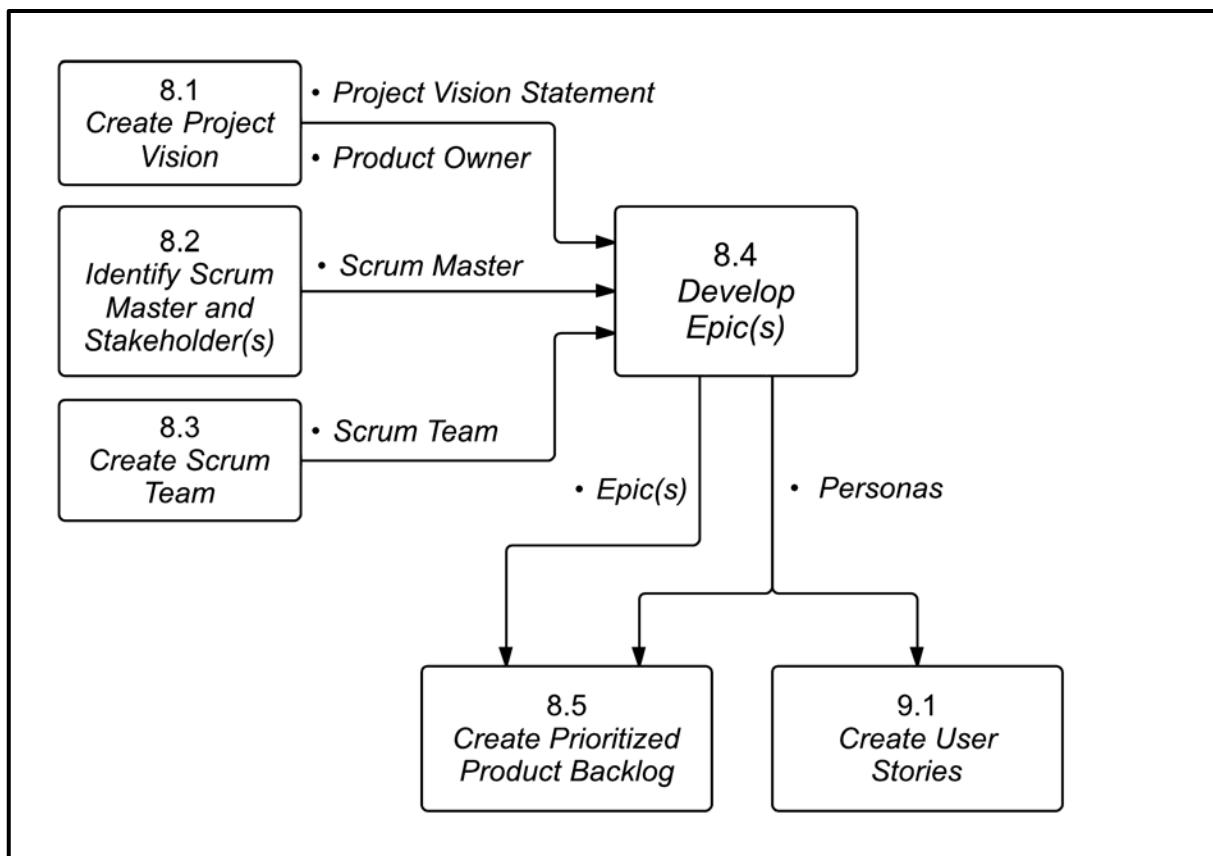


Figura 8-11: Desarrollo de épica(s)—Diagrama de Flujo

8.4.1 Entradas

8.4.1.1 *Scrum Core Team**

El Equipo Principal/Central de Scrum consiste en el *Equipo Scrum*, el *Scrum Master* y el *Propietario del producto* como se describe en la sección 3.3.1.

8.4.1.2 *Reunión de la Visión del Proyecto **

Descrito en la sección 8.1.3.2.

8.4.1.3 *Stakeholder(s)*

Descrito en la sección 8.2.3.2.

8.4.1.4 *Program Product Backlog*

Descrito en la sección 8.1.1.6.

8.4.1.5 *Approved Change Requests*

Approved changed requests que vienen del *program* o *portfolio* son entradas que se añadirán a la lista de cambios de *project* aprobados para su ejecución en *Sprints* futuros. Cada cambio puede requerir su propio *Epic* o *User Story* y podría convertirse en un aporte al proceso de *Desarrollo de épica(s)*. *Approved Change Requests* en este proceso también podría ser el resultado de otros procesos de Scrum.

Change Requests y *Approved Change Requests* se discuten en las secciones 6.3.1, 6.4.2.1 y 6.6.

8.4.1.6 *Unapproved Change Requests*

Los pedidos de cambios se presentan por lo general como *Change Requests* y permanecer en un estado de “*no aprobado*” hasta que sean formalmente aprobados. *Unapproved Change Requests* para el proceso *Desarrollo de épica(s)* podría provenir de *Crear entregables*, *Llevar a cabo el Standup diario* y otros procesos.

Change requests y *Unapproved Change Requests* se discuten en las secciones 6.3.1, 6.4.2.1 y 6.6.

8.4.1.7 Program and Portfolio Risks

Risks relacionados con un *portfolio* o el programa también tendrán un impacto en los *projects* que forman parte del respectivo *portfolio* o *program*. Durante *risk assessment* de *portfolios* y *programs*, si se determina que el riesgo puede afectar a un *project* individual, la información relevante sobre *risk* debe ser comunicada al *Propietario del producto* y *Equipo Scrum*. *Program and Portfolio Risks* podrían ser entradas para el proceso *Desarrollo de épica(s)* y pueden tener un impacto global sobre cómo se lleva a cabo este proceso.

Program y Portfolio Risks se describen en la sección 7.5.1.

8.4.1.8 Laws and Regulations (Leyes y regulaciones)

Dependiendo del *project*, hay *Laws and Regulations* impuestos por un consejo de administración que afectan la planificación y ejecución. Las leyes son externas a la organización e impuesta por una entidad gubernamental. Las regulaciones pueden ser o bien interna o externa. Las regulaciones internas son aquellas que son aplicables dentro de la empresa, por lo general basadas en la política de la empresa. Estas regulaciones pueden estar relacionadas con los sistemas de *quality management*, regulaciones financieras, reglamentos de personal, etc. Las regulaciones externas son las relativas a los criterios, las normas y los requisitos gubernamentales establecidos.

Laws and Regulations deben ser considerados al desarrollar *Epics*. Los *Epics* se basan en *business requirements*. Para cumplir con estos requisitos el equipo del proyecto tiene que cumplir con ambas leyes y reglamentos internos y externos.

A veces, algunas de las leyes y reglamentos que afectan a múltiples *projects* Scrum pueden incluirse como parte de los *Cuerpo de asesoramiento de Scrum Recommendations*, como se discutió en la sección 8.1.1.12.

8.4.1.9 Los contratos aplicables

Si todo el *project* o partes del mismo se cumple(n) en virtud de un contrato, el contrato ha de definir el alcance del trabajo y las condiciones específicas del mismo. El tipo de contrato utilizado tendrá influencia en el riesgo del *project*.

Algunos de los tipos más comunes de contratos utilizados en *projects* Scrum son los siguientes:

Incremental Delivery Contract—Este contrato incluye puntos de *inspección* en intervalos regulares. Ayuda a que el *customer* o *socios* tome(n) decisiones sobre el desarrollo de los *products* periódicamente a lo largo del *project* en cada punto de inspección. El *customer* puede aceptar el desarrollo del *product*, decidir parar el desarrollo del *product*, o solicitar modificaciones del *product*.

Joint Venture Contract—Este contrato se utiliza generalmente cuando dos o más socios llevan a cabo el trabajo de un proyecto. Las partes involucradas en el proyecto lograr algún *Return on Investment*, porque los ingresos o beneficios generados serán compartidos entre todos.

Development in Phases Contract—Este contrato permite que los fondos estén disponibles cada mes o cada trimestre después de que un *release* se complete con éxito. De esta forma, se les incentiva al *customer* y a los proveedores y se asegura de que el riesgo monetario para el *customer* se limite a ese período de tiempo determinado, ya que no son financiados los lanzamientos sin éxito.

Incentive and Penalty Contract—Estos contratos se basan en el acuerdo de que el proveedor se verá recompensado con un incentivo financiero si los *products* del *project* se entregan a tiempo, y acumulará sanciones financieras si la entrega se retrasa.

Otros tipos de contratos más populares incluyen el pago por las características del contrato, el tiempo y los materiales del contrato, precio fijo y contrato de alcance fijo, y el contrato de beneficio fijo.

Los *Epics* deberían desarrollarse teniendo en cuenta los términos y condiciones del tipo de contrato que se utilice.

8.4.1.10 Previous Project Information (Información previa del proyecto)

Información y conocimientos adquiridos en *projects* anteriores y similares dentro de la organización son insumos valiosos para el desarrollo de *Epics* y de la evaluación de *risks*. *Previous Project Information* podría incluir las notas del director del *project*, registros de *projects* y los comentarios por el *stakeholder*.

Algunos detalles y las mejores prácticas relacionadas con información de *projects* anteriores pueden estar disponibles a través de *Cuerpo de asesoramiento de Scrum Recommendations*.

8.4.1.11 Cuerpo de asesoramiento de Scrum Recommendations

Discutidos en la sección 8.1.1.12

Cuerpo de asesoramiento de Scrum Recommendations pueden incluir información sobre las normas, los reglamentos, las reglas y las mejores prácticas para el desarrollo de *Epics*.

8.4.2 Herramientas

8.4.2.1 User Group Meetings*

User Group Meetings implica a relevantes *socios* (principalmente usuarios o *customers* del *product*). Ellos le proporcionan al Equipo Principal de Scrum con información de primera mano acerca de las expectativas del usuario. Esto ayuda en la formulación de los *Acceptance Criteria* para el *product* y proporciona información valiosa para el desarrollo de *Epics*. *User Group Meetings* son de vital importancia en la prevención del trabajo costoso que puedan deberse a la falta de claridad con respecto a las expectativas y exigencias. Estas reuniones también promueven que se crea en el *project* y crea un entendimiento común entre el Equipo Principal de Scrum y el/los *Stakeholder(s)* pertinente(s).

8.4.2.2 User Story Workshops

User Story Workshops se llevan a cabo como parte del proceso de *Desarrollo de épica(s)*. El *Scrum Master* facilita estas sesiones en las que todo el *Scrum Core Team* interviene y, en ocasiones, es deseable incluir a otro(s) *Stakeholder(s)*. Estos talleres ayudan al *Propietario del producto* a dar prioridad a los requisitos y permitir que el *Scrum Core Team* tenga una perspectiva compartida de los *Acceptance Criteria*. Se aseguran de que los *Epics* y *User Stories* describan la funcionalidad desde el punto de vista de los usuarios, sean fáciles de entender, y que puedan ser calculados de forma fiable. *User Story workshops* son útiles en la comprensión de las expectativas del usuario para los entregables y son excelentes para la formación de equipos. También facilitan la preparación para la planificación del próximo *Sprint*. Un Taller de historia de *User Story* es una buena plataforma para discutir y aclarar todos los elementos de un *product* y a menudo profundizar en los detalles más pequeños para garantizar la claridad.

8

8.4.2.3 Focus Group Meetings

En los *Focus Group Meetings* se reúnen las personas en una sesión guiada para proporcionar sus opiniones, percepciones o valoraciones de un *product*, servicio o resultado deseado. Los miembros del grupo de enfoque tienen la libertad de hacerse preguntas el uno al otro y de obtener aclaraciones sobre temas o conceptos específicos. A través de cuestionamiento, la crítica constructiva y la retroalimentación, los grupos focales conducen a un *product* de mejor calidad y con ello contribuyen a la satisfacción de las expectativas de los usuarios. En estas reuniones, los miembros del grupo de enfoque a veces llegan a un consenso en ciertas áreas, mientras que en otras áreas las opiniones pueden ser diferentes. Cuando los miembros del grupo tienen diferentes opiniones o puntos de vista, no se escatiman esfuerzos para resolver las diferencias con el fin de llegar a un consenso.

Las sesiones de grupos focales pueden ayudar a los equipos a crear ideas innovadoras, resolver problemas, y dar sugerencias para mejorar. Estas reuniones generan ideas y retroalimentación de los

usuarios potenciales y desarrolladores de *products*. Estas reuniones se realizan normalmente para la planificación, la evaluación, o la mejora de un *product* o servicio. Perspectivas obtenidas a partir de estas reuniones también pueden ayudar con *Develop Epics*, y con los *User Stories*. A veces, los *focus group meetings* se llevan a cabo para resolver los *issues* que puedan surgir durante el desarrollo de los *Epics*.

8.4.2.4 Entrevistas con usuario o *customers*

El incorporar a los *socios*, incluyendo al patrocinador, los usuarios, y los *customers* del *product* es importante para ganar el contexto y la visión necesaria para desarrollar *Epics*. Este tiempo compartido para entrevistar a los usuarios y *customers* resultará en un mejor alineamiento de los requisitos de *Epics* con la visión general de *projects*, ofreciendo de esta forma un mayor valor.

Estas entrevistas ayudan a:

- Identificar y entender las necesidades y expectativas del *stakeholder*
- Reunir opiniones y hechos
- Comprender la perspectiva del *stakeholder* sobre el producto final
- Recopilar la retroalimentación sobre el producto iterado o parcialmente desarrollado

8.4.2.5 Questionario

Una forma económica de obtener una perspectiva estadística cuantitativa y cualitativa de un gran número de usuarios o *customers* es el uso de encuestas o cuestionarios. Un cuestionario es un instrumento de investigación con el fin de recopilar información sobre un asunto o tema específico. Los cuestionarios pueden ser auto-administrados o administrados por un entrevistador.

Gran cuidado debe ser ejercido en el diseño de cuestionarios, la selección del público debe ser adecuada, y la determinación de un método apropiado de implementación de encuestas es también de gran importancia para evitar errores y prejuicios.

Durante el desarrollo de *Epics*, el *Propietario del producto* o el *Scrum Master* podría llevar a cabo una encuesta para recopilar la información pertinente de *socios* o el *Equipo Scrum*.

8.4.2.6 Técnicas de *Risk Identification*

Descrito en la sección 7.4.1.1

8.4.2.7 Cuerpo de asesoramiento de Scrum Expertise

Descrito en la sección 3.3.2

Al crear *Epics*, el *Cuerpo de asesoramiento de Scrum Expertise* podría relatar las normas y reglamentos documentados; o los estándares y mejores prácticas para la creación de *Epics*. También puede haber un equipo de expertos en la materia que pueda ayudar al *Propietario del producto* a crear *Epics*. Este equipo podría incluir analistas de negocios, arquitectos, desarrolladores, expertos de Scrum u otras personas con experiencia. Generalmente este grupo de expertos no es el mismo equipo que se quedará y trabajará en un *project* en particular, ya que tienen que pasar de un *project* a otro durante la "fase de venta" o "fase cero" con *customers* o usuarios.

8.4.3 Salidas

8.4.3.1 Epic(s)*

8

Los Epics se escriben en las etapas iniciales del proyecto, cuando la mayoría de los *User Stories* son funcionalidades de alto nivel o descripciones de *products* que están ampliamente definidas. *Epics* son *user stories* grandes sin refinar en el *Prioritized Product Backlog*.

Una vez que estos *Epics* aparecen en el *Prioritized Product Backlog* para ser terminados en el próximo *Sprint*, se convierten en *User Stories* más pequeños. Estos *User Stories* más pequeñas son generalmente funcionalidades simples, cortas y fáciles de implementar, o bloques de tareas que deben completarse en un *Sprint*.

8.4.3.2 Personas*

Personas son personajes de ficción altamente detallados. Estos representan a la mayoría de los usuarios y otros *socios* que puede ser no utilicen directamente el producto final. *Personas* se creó para identificar las necesidades de los usuarios. La creación de *Personas* específicas puede ayudar al equipo a entender mejor a los usuarios y sus necesidades y metas. Basado en una *Persona*, el *Propietario del producto* puede priorizar de manera más efectiva las funciones para crear el *Prioritized Product Backlog*.

Creación de una Persona: Esto implica la asignación al personaje de un nombre ficticio y preferentemente una foto, como una foto de banco (*stock photo*). A la *Persona* se le incluirá atributos muy específicos como su edad, género, nivel de educación, entorno, intereses y metas. Una cita que ilustra las necesidades de la persona puede ser incluida también. A continuación se muestra un ejemplo de una *Persona* para un sitio web de viajes.

Ejemplo:

Vanessa tiene 39 años de edad y es residente de San Francisco. Ella es apasionada por viajar y después de haber tenido una exitosa carrera como abogada, ha decidido disfrutar de su pasión por el viaje. Le gusta tener opciones al seleccionar sus viajes aéreos y servicios de alojamiento para que ella pueda elegir el mejor, a un precio accesible. Ella se siente frustrada con los sitios web lentos y desordenados.

8.4.3.3 *Approved Changes*

Unapproved Change Requests pueden ser aprobados por el *Propietario del producto* durante el proceso de *Desarrollo de épica(s)*, a veces con sugerencias proporcionadas por los *socios relevantes*. Tales cambios se clasifican como las modificaciones aprobadas (*approved changes*) y pueden ser priorizadas e implementadas en los futuros *Sprints*.

Change Requests y *Approved Change Requests* se discuten en las secciones 6.3.1, 6.4.2.1 y 6.6.

8.4.3.4 *Risks identificados*

Al crear *Epics*, nuevos *risks* pueden ser identificados y estos *risks* identificados constituyen una salida importante de esta etapa. Estos *risks* contribuyen al desarrollo del *Prioritized Product Backlog* (también se conoce como *Risk Adjusted Product Backlog*).

Risk identification se describe en la sección 7.4.1.

8.5 Creación de la lista priorizada de pendientes del producto

La figura 8-12 muestra todas las entradas, las herramientas y las salidas para el proceso *Creación de la lista priorizada de pendientes del producto*.

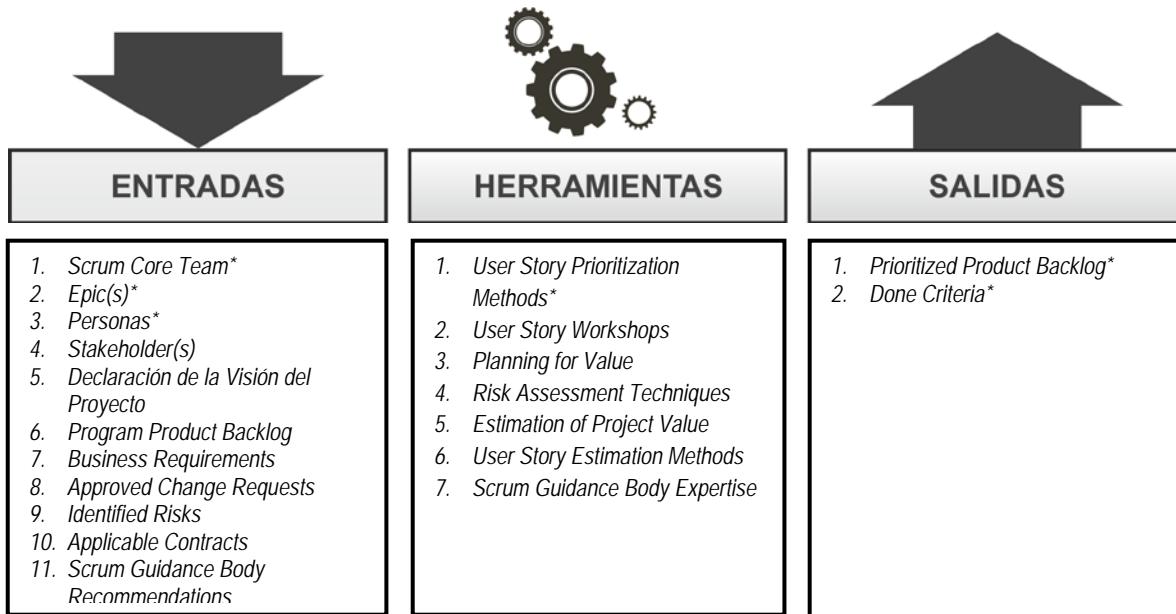


Figura 8-12: *Creación de la lista priorizada de pendientes del producto*—Entradas, Herramientas y Salidas

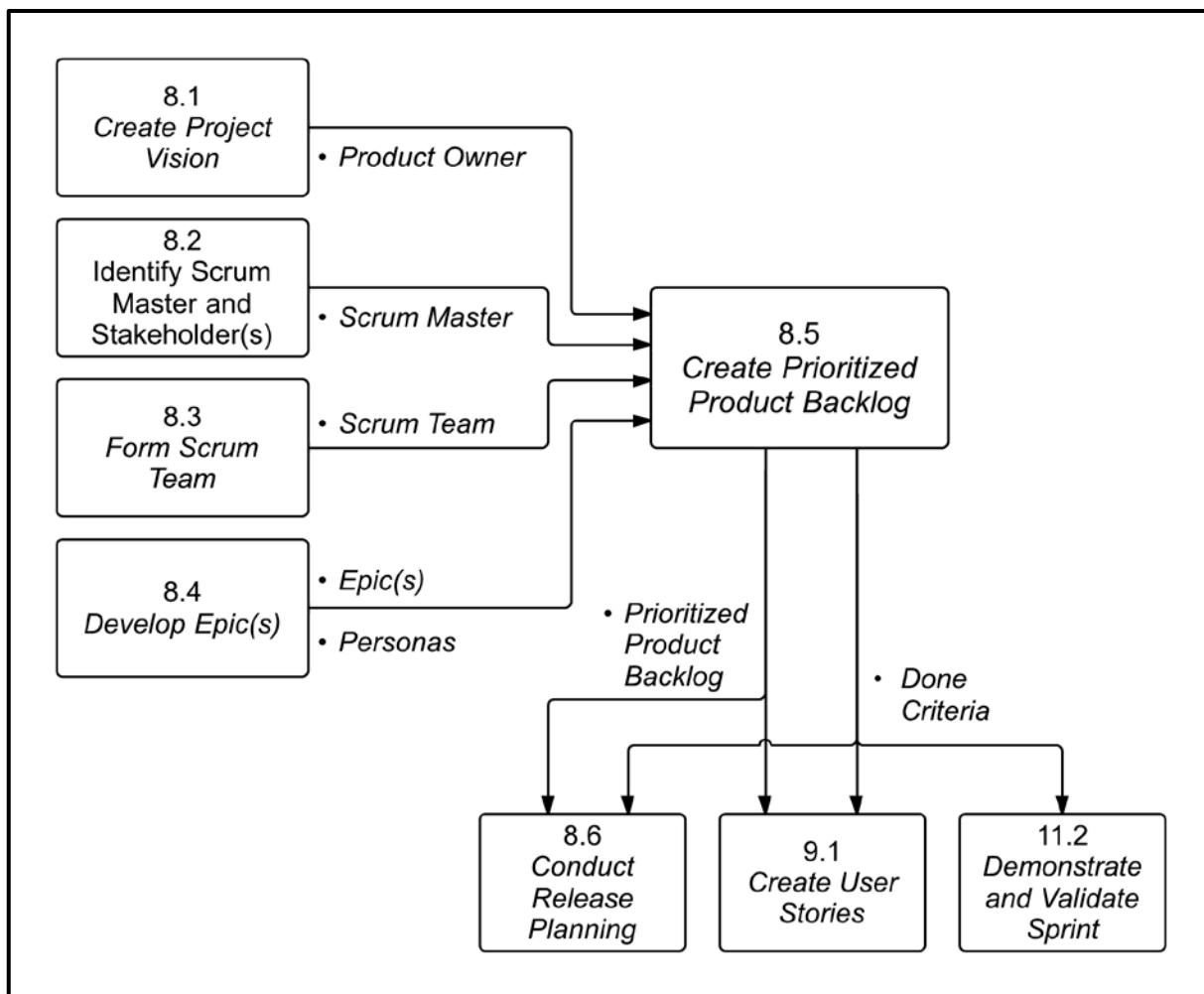


Figura 8-13: Creación de la lista priorizada de pendientes del producto— Diagrama de flujo de datos

8.5.1 Entradas

8.5.1.1 *Scrum Core Team**

Descrito en la sección 8.4.1.1.

8.5.1.2 *Epic(s)**

Descrito en la sección 8.4.3.1.

8.5.1.3 *Personas**

Descrito en la sección 8.4.3.2.

8

8.5.1.4 *Stakeholder(s)*

Descrito en la sección 8.2.3.2.

8.5.1.5 *Reunión de la Visión del Proyecto*

Descrito en la sección 8.1.3.2.

8.5.1.6 *Program Product Backlog*

Descrito en la sección 8.1.1.6.

8.5.1.7 *Business Requirements*

La suma de todos los conocimientos adquiridos a través de diversas herramientas como entrevistas a los *customers* o usuario o, Cuestionarios, JAD Sessions, Gap Analysis, SWOT Analysis, y otras reuniones, ayuda a desarrollar una mejor perspectiva sobre los *business requirements* y ayuda en la creación del *Prioritized Product Backlog*.

8.5.1.8 *Approved Change Requests*

Descrito en la sección 8.4.3.3.

8.5.1.9 *Risks identificados*

Descrito en la sección 8.4.3.4.

8.5.1.10 Los contratos aplicables

Descrito en la sección 8.4.1.9.

8.5.1.11 *Cuerpo de asesoramiento de Scrum Recommendations*

Descrito en la sección 8.1.1.12.

Durante la creación del *Prioritized Product Backlog*, *Cuerpo de asesoramiento de Scrum* puede incluir información sobre las normas, los reglamentos, los procedimientos y las mejores prácticas para el desarrollo del *Prioritized Product Backlog*.

8.5.2 Herramientas

8.5.2.1 *User Story Prioritization Methods** (Métodos para establecer prioridad de los *User Stories*)

A continuación se presentan algunas de las técnicas que se utilizan para dar prioridad a los *User Stories* o requisitos en el *Prioritized Product Backlog* sobre la base del valor de negocio:

- ***MoSCoW Prioritization scheme***—El *MoSCoW prioritization scheme* deriva su nombre de las primeras letras de las frases "must have" (debe tener), "should have" (debería tener), "could have" (podría tener), y "will not have" (no tendrá). Este método de *priorization* es generalmente más efectivo que un *simple scheme*. Las etiquetas están en orden decreciente de prioridad. "Debe tener" *user stories* son aquellos sin los que el *product* no tendrá valor y "no tendrá" *User Stories* son aquellos que, a pesar de que sería bueno tener, no son necesarios para ser incluidos.
- ***Paired Comparison***—esta técnica se prepara una lista de todas las *user stories* en el *Prioritized Product Backlog*. A continuación, cada *User Story* se toma de forma individual y se compara con

las otros *User stories* en la lista, uno a la vez. Cada vez que dos *User stories* se comparan, se toma una decisión en cuanto a cuál de los dos es más importante. A través de este proceso, una lista de prioridades de los *User stories* se puede generar.

- **100-Point Method**—El *100-Point Method* (*Método de 100 puntos*) fue desarrollado por Dean Leffingwell y Don Widrig (2003). Se trata de darle al *customer* 100 puntos que pueden usar para votar por los *User Stories* que son más importantes. El objetivo es dar más peso a las *User Stories* que son de mayor prioridad en comparación con las otras *User Stories* disponibles. Cada miembro del grupo asigna puntos a las diversas *User Stories*, dando más puntos a los que se sienten son más importantes. Al finalizar el proceso de votación, la *prioritization* se determina calculando el total de puntos asignados a cada *User Story*.
- **Kano Analysis**

Descrito en la sección 4.5.2

8.5.2.2 *User Story Workshops*

8

Descrito en la sección 8.4.2.2.

8.5.2.3 *Planning for Value*

Descrito en la sección 4.5.2

8.5.2.4 Técnicas de *Risk Assessment*

Descrito en la sección 7.4.2.1.

8.5.2.5 Estimación de Valor del Proyecto

Descrito en la sección 4.5.1.

8.5.2.6 Métodos de estimación del *User Story*

Todas las herramientas estimadas usadas para los procesos *Approve*, *Estimate and Commit User Stories* (como se describe en la sección 9.2.2) se pueden utilizar para crear estimaciones de alto nivel para *Epic(s)* cuando creamos el *Prioritized Product Backlog*. Algunas herramientas importantes son:

1. *User Group Meetings*
2. *Planning Poker*
3. *Fist of Five*
4. Puntos para la Estimación de Costos
5. Otras técnicas de estimación

8.5.2.7 Cuerpo de asesoramiento de *Scrum Expertise*

Descrito en la sección 8.4.2.7

Durante la creación del *Prioritized Product Backlog*, el *Cuerpo de asesoramiento de Scrum Expertise* podría relacionarse con las normas y reglamentos, y las mejores prácticas documentadas para la creación de *Epics*. También puede haber un equipo de expertos en la materia que podría ayudar al *Propietario del producto* en la opción *Creación de la lista priorizada de pendientes del producto*. Este equipo podría incluir analistas de negocios, arquitectos, desarrolladores, Expertos de Scrum, y otras personas con experiencia. Generalmente este grupo de expertos no es el mismo equipo que se quedará trabajando en este *project*, ya que tienden a pasar de un *project* a otro durante la "fase de venta" o "fase cero" con los *customers* o usuarios.

8.5.3 Salidas

8.5.3.1 *Prioritized Product Backlog**

El *Propietario del producto* desarrolla una *Prioritized Product Backlog*, que contiene una lista priorizada de los requerimientos del negocio y de los *projects* escritos en forma de *Epic(s)*, que son de altos niveles de *User Stories*. El *Prioritized Product Backlog* se basa en tres factores principales: el valor, riesgo o incertidumbre, y dependencias. También se le conoce como *Risk Adjustment Product Backlog* dado a que incluye *risks* identificados y evaluados relacionados con el *project*. También incluye cambios aprobados que pueden ser priorizados adecuadamente en el *Prioritized Product Backlog* (tal como se describe en la sección 6.3.1).

- **Valor**—Es la responsabilidad del *Propietario del producto* asegurar en primer lugar la entrega de los productos que ofrezcan el mayor valor. Incluso un *product* de gran valor no puede ser parte del

primer *release* si hay otros *products* de mayor valor que son suficientes para un primer lanzamiento.

- **Riesgo e Incertidumbre**—Cuanta más incertidumbre existe, más riesgoso es el *project*. Por lo tanto, es importante que los *products* de mayor riesgo en el *Prioritized Product Backlog* se les de mayor prioridad. Los *products* que llevan un mayor nivel de *risk* también requerirán acciones de *risk mitigation*. Cuando estas acciones de *risk mitigation* se priorizan contra el atraso, el resultado es un *Risk Adjusted Product Backlog*. Tratar con *risks* al principio del *project* no garantiza que el *project* será un éxito, pero sí mejorará la capacidad del equipo para hacer frente a los *risks*. Esto se describe en la sección 7.4.3.
- **Dependencias**—Por lo general, no es posible crear un *Prioritized Product Backlog* en la que no existen dependencias entre los *User Stories*. Los requisitos funcionales a menudo dependen de otros requisitos funcionales e incluso no funcionales. Estas dependencias pueden afectar cómo se priorizan los *user stories* en el *Prioritized Product Backlog*. Dos de las formas más comunes para resolver dependencias son, o bien dividir una sola historia en varias partes, o combinar historias interdependientes.
- **Estimaciones**—Las estimaciones de alto nivel para *Epic(s)* también están disponibles en el *Prioritized Product Backlog*.

8.5.3.2 *Done Criteria**

Done Criteria es un conjunto de reglas que se aplican a todos los *User Stories*. Una definición clara de *Done* es crítica, ya que elimina la ambigüedad de los requisitos y ayuda a que el equipo se adhiera a las normas de calidad obligatorias. Esta clara definición se utiliza para crear los *Done Criteria*, que son un resultado del proceso de *Creación de la lista priorizada de pendientes del producto*. Un *User Story* se considera *Done* cuando se le demuestra y se aprueba por el *Propietario del producto* que juzga sobre la base de los *Done Criteria* y los *User Story Acceptance Criteria*.

Ejemplo de Done Criteria:

Proyecto: Diseñar las nuevas variantes de un coche deportivo popular en LRA Ltd.

Done Criteria:

- El diseño es aprobado por la división de Excelencia Técnica.
- El prototipo pasa todas las pruebas de túnel de viento mandado por la División de Aerodinámica.
- El diseño se aprueba para la producción por la división de Propiedad Intelectual.
- Las expectativas de diseño de seguridad son corroboradas por el informe de Diseño de Seguridad de la División de Seguridad.
- El informe de estimación de costos para el diseño es aprobado por la división de Finanzas.

8.6 Realizar el plan de lanzamiento

La figura 8-14 muestra todas las entradas, herramientas y salidas para el proceso de *Realizar el plan de lanzamiento*.

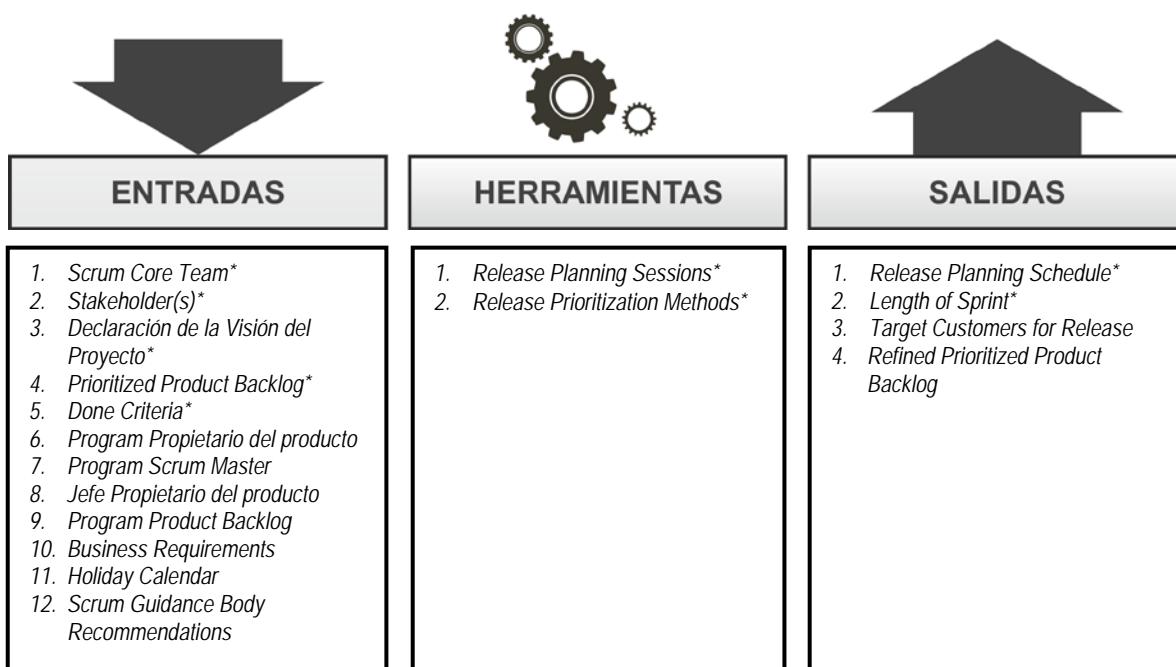
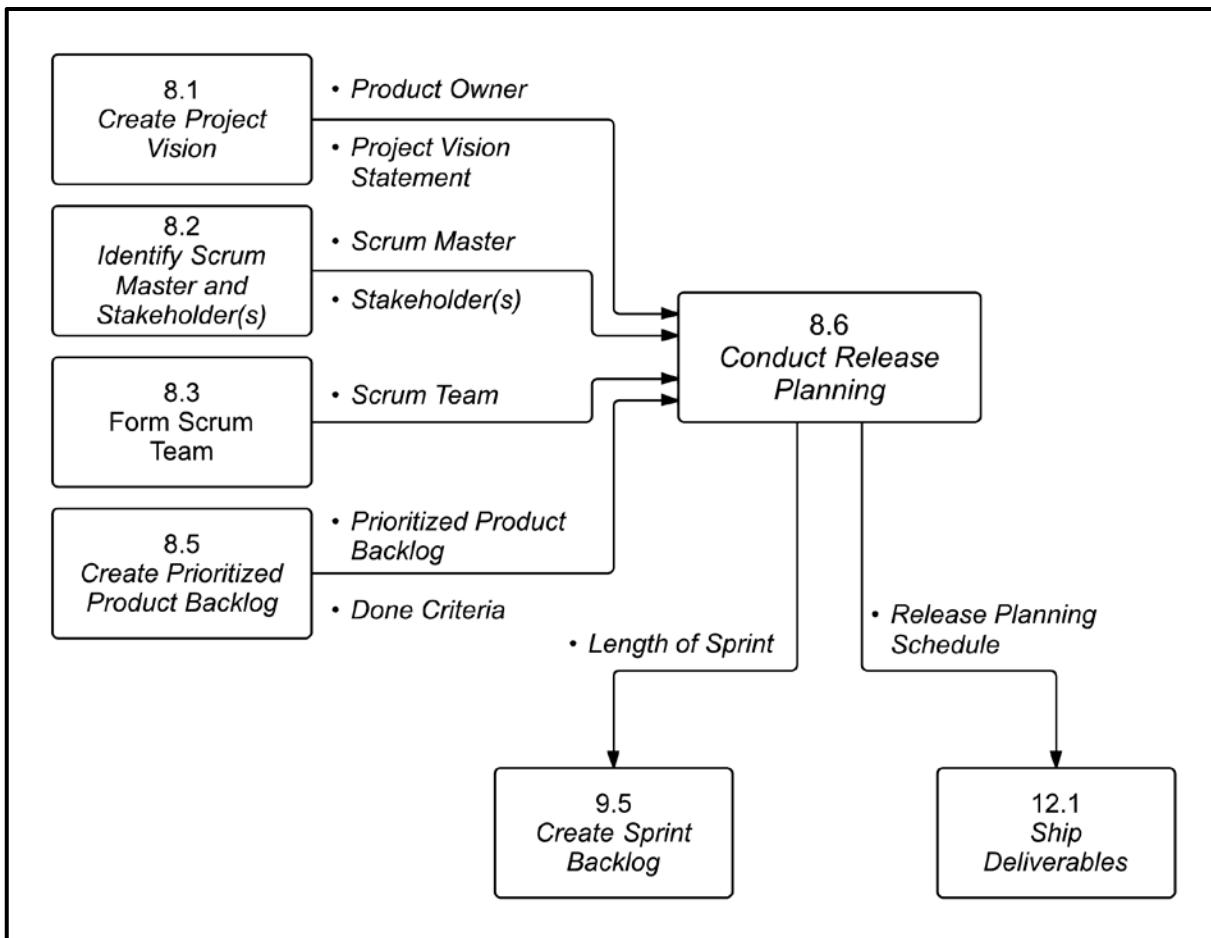


Figura 8-14: *Realizar el plan de lanzamiento*—Entrada, Herramientas y Salidas

Figura 8-15: Realizar el *plan de lanzamiento*—Diagrama de flujo de datos

8.6.1 Entradas

8.6.1.1 *Scrum Core Team**

Descrito en la sección 8.4.1.1.

8.6.1.2 *Stakeholder(s)**

Descrito en la sección 8.2.3.2.

8.6.1.3 *Reunión de la Visión del Proyecto **

Descrito en la sección 8.1.3.2.

8.6.1.4 *Prioritized Product Backlog**

Descrito en la sección 8.5.3.1.

8.6.1.5 *Done Criteria**

Descrito en la sección 8.5.3.2.

8.6.1.6 *Program Propietario del producto*

Descrito en la sección 8.1.1.2.

8.6.1.7 *Program Scrum Master*

Descrito en la sección 8.1.1.3.

8.6.1.8 *Jefe Propietario del producto*

Descrito en la sección 8.1.1.5.

8.6.1.9 *Program Product Backlog*

Descrito en la sección 8.1.1.6.

8.6.1.10 *Business Requirements*

Descrito en la sección 8.5.1.7.

8.6.1.11 *Calendario de vacaciones*

Es importante para el *Equipo Scrum* estar al tanto de las fechas claves y la disponibilidad de todos los miembros del equipo. Esto se puede lograr mediante el uso de un calendario compartido que proporciona información sobre los días feriados, licencias, faltas al trabajo, los planes de viaje, eventos, etc. Este calendario ayudará al equipo en la planificación y ejecución de *Sprints*.

8

8.6.1.12 *Cuerpo de asesoramiento de Scrum Recommendations*

Descrito en la sección 8.1.1.12

En el proceso de *Realizar el plan de lanzamiento*, las recomendaciones de *Cuerpo de asesoramiento de Scrum* pueden relacionarse con las normas, reglamentos, estándares y mejores prácticas para el desarrollo del Plan de Lanzamiento. El Cuerpo de Orientación puede ser la mejor autoridad para definir las directrices relacionadas con el valor del negocio, las expectativas de liberación, estrategias de implementación, la calidad y la seguridad.

8.6.2 *Herramientas*

8.6.2.1 *Release Planning Sessions**

Release Planning Sessions se llevan a cabo para desarrollar un Plan de Lanzamiento. El plan define cuando varios conjuntos de funcionalidad o productos utilizables serán entregados al *customer*. En Scrum, el objetivo principal de una reunión de planificación de lanzamiento es hacer que el *Equipo Scrum* tenga una visión general de las emisiones y el calendario de entrega del *product* que están desarrollando para que puedan alinearse con las expectativas del *Propietario del producto* y los relevantes *socios* (principalmente el patrocinador del *proyecto*).

Muchas organizaciones tienen una estrategia en relación con la liberación de los productos. Algunas organizaciones prefieren despliegue continuo, donde se produce una liberación/un lanzamiento después de la creación de la funcionalidad útil especificada. Otras organizaciones prefieren despliegue por etapas, donde las liberaciones se hacen en intervalos predefinidos. Dependiendo de la estrategia de la organización, *Release Planning Session* en los *projects* puede ser impulsado por la funcionalidad, en el que el objetivo es hacer un lanzamiento una vez que un conjunto predeterminado de funcionalidad se ha desarrollado; o la planificación puede ser impulsada por la *fecha*, en la que ocurre la liberación en una fecha predefinida.

Dado a que el marco de Scrum promueve la planificación basada en información y la práctica de decisiones iterativas, sobre la planificación detallada por adelantado como en la gestión de proyectos tradicional estilo Cascada, no es necesario que *Release Planning Session* elabore un plan de lanzamiento detallado de todo el *project*. En Scrum, el Plan de Lanzamiento (*Release Plan*) se puede actualizar continuamente a medida que la información relevante está disponible.

8.6.2.2 *Release Prioritization Methods**

Release Prioritization Methods se utilizan para desarrollar un *Release Plan*. Estos métodos son específicos a la industria y organización y generalmente son determinados por la alta dirección de la organización.

8.6.3 Salidas

8.6.3.1 *Release Planning Schedule**

Un *Release Planning Schedule* es uno de los resultados más importantes del proceso llamado *Conduct Planning Schedule*. Un *Release Planning Schedule* indica que entregas van a ser despachadas a los *customers*, junto con intervalos planificados y fechas para los *releases*. Puede que no haya un lanzamiento previsto a finales de cada iteración *Sprint*. A veces, un lanzamiento puede ser planificado después que un grupo de iteraciones *Sprint* se ha completado. Dependiendo de la estrategia de la organización, *Release Planning Sessions* de los *projects* pueden ser impulsado por la funcionalidad en el que el objetivo es la entrega una vez que un conjunto predeterminado de funcionalidad que se ha desarrollado, o la planificación puede ser impulsada por la *fecha*, en la que ocurre la liberación en una fecha predefinida. La entrega debe ser *released (liberada)* cuando ofrece valor empresarial suficiente para el *customer*.

8.6.3.2 *Length of Sprint**

Basado en las diversas entradas que incluyen *Business Requirements* y *Release Planning Schedule*, el *Propietario del producto* y el *Equipo Scrum* deciden sobre el *Length of Sprint* para el proyecto. Una vez determinado, el *Length of Sprint* a menudo sigue siendo el mismo durante todo el proyecto.

Sin embargo, el *Length of Sprint* puede ser cambiado si y como el *Propietario del producto* y el *Equipo Scrum* lo consideren apropiado. Puede ser que temprano en el *project* todavía estén experimentando para encontrar la mejor longitud del *Sprint*. Si más adelante en el *project* hay un cambio en *Length of Sprint*, típicamente este cambio sería una reducción del *Sprint* debido a las mejoras en el entorno del *project*.

Un *Sprint* podría ser *Time-boxed* de 1 a 6 semanas. Sin embargo, para obtener los máximos beneficios de un proyecto Scrum, siempre se recomienda mantener el *Sprint Time-boxed* a 4 semanas, a menos que existan proyectos con requisitos muy estables, donde los *Sprints* pueden extenderse hasta 6 semanas.

El impacto de un Cambio Esperado (Expected Change) en el *Length of Sprint* se describe en la sección 6.5.1

8.6.3.3 *Target Customers for Release*

No todos los lanzamientos se dirigirán a todos los *socios* o usuarios. El/los stakeholder(s) puede(n) optar por limitar ciertos comunicados a un subconjunto de usuarios. El Release Plan (El Plan de Lanzamiento) debe especificar los *customers* en quienes se va a enfocar el lanzamiento (reléase).

8.6.3.4 *Refined Prioritized Product Backlog*

El *Prioritized Product Backlog*, desarrollado en el proceso de *Creación de la lista priorizada de pendientes del producto*, se puede refinar en este proceso. Es posible que haya más claridad sobre los *User Stories* en el *Prioritized Product Backlog* después de que el Equipo Principal de Scrum lleve a cabo *Release Planning Sessions* con el/los *Stakeholder(s)*.

8.7 Fase *diagrama de flujo de datos*

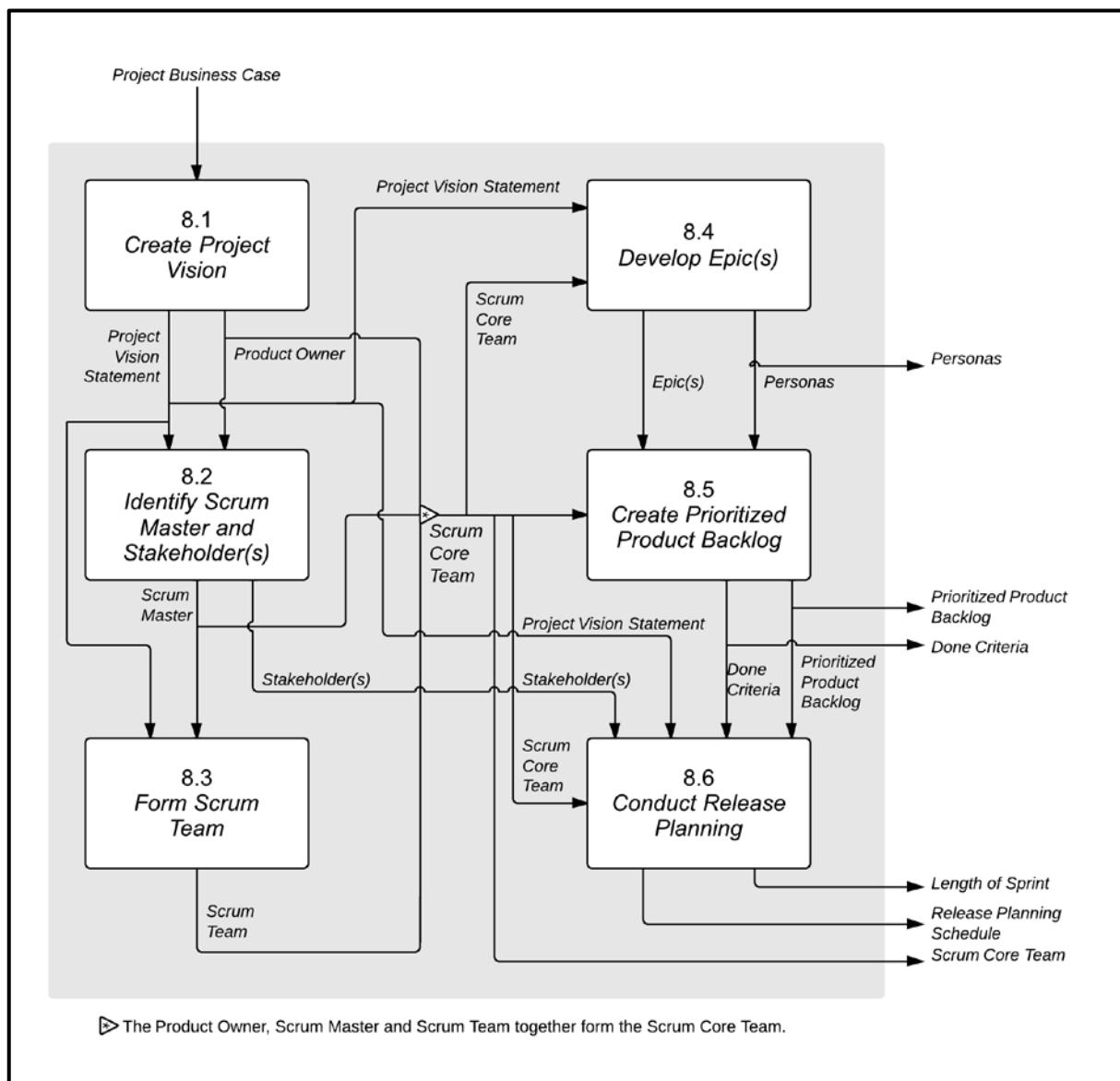


Figura 8-16: *Initiate Phase—Diagrama de flujo de datos*

9. PLANEAR Y ESTIMAR

La fase de *Plan and Estimate* consiste en procesos relacionados con la planificación y las tareas de estimación, que incluyen *Elaborar historias de usuario*, *Aprobar, estimar y asignar historias de usuarios*, *Elaboración de tareas*, *Estimar tareas*, y *Elaboración de la lista de pendientes del Sprint*.

Plan and Estimate, tal como se define en *Una guía para el conocimiento de Scrum (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolios, programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se les entregará a los socios
- *Projects* de cualquier tamaño y complejidad

El término "producto" en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria - desde pequeños proyectos o equipos con tan sólo seis miembros por equipo, hasta proyectos grandes y complejos que cuentan con cientos de miembros por equipo.

A fin de facilitar la mejor aplicación del marco de Scrum, en este capítulo se identifican las entradas, herramientas y salidas de cada proceso, ya sea como "obligatorio" u "opcional". Las entradas, herramientas y salidas indicadas por asteriscos (*) son obligatorias, mientras que las que no tienen asteriscos son opcionales.

Se recomienda que el *Equipo Scrum* y aquellas personas que recién comienzan a aprender sobre el marco y los procesos de Scrum, se centran principalmente en las aportaciones obligatorias, las herramientas y los productos; mientras que los *Propietario del producto*, *Scrum Masters*, y otros practicantes con experiencia de Scrum se deberían de esforzar por alcanzar un conocimiento más profundo de la información ofrecida en este capítulo. También es importante darse cuenta de que, aunque todos los procesos se definen de forma única en la *Guía SBOK™*, no necesariamente se llevan a cabo de forma secuencial o por separado. A veces, puede ser más apropiado combinar algunos procesos, dependiendo de los requisitos específicos de cada proyecto.

Este capítulo está escrito desde la perspectiva de un *Equipo Scrum* trabajando en un *Sprint* para producir entregables (*deliverables*) como parte de un proyecto más amplio. Sin embargo, la información que se describe es igualmente aplicable a proyectos enteros, programas y *portfolios*. Información adicional relacionada con el uso de Scrum para proyectos, programas y *portfolios* está disponible en los capítulos 2 al 7, que cubren los principios y aspectos de Scrum.

Figura 9-1 Proporciona una descripción general de los procesos de *Plan and Estimate Phase*, que son los siguientes:

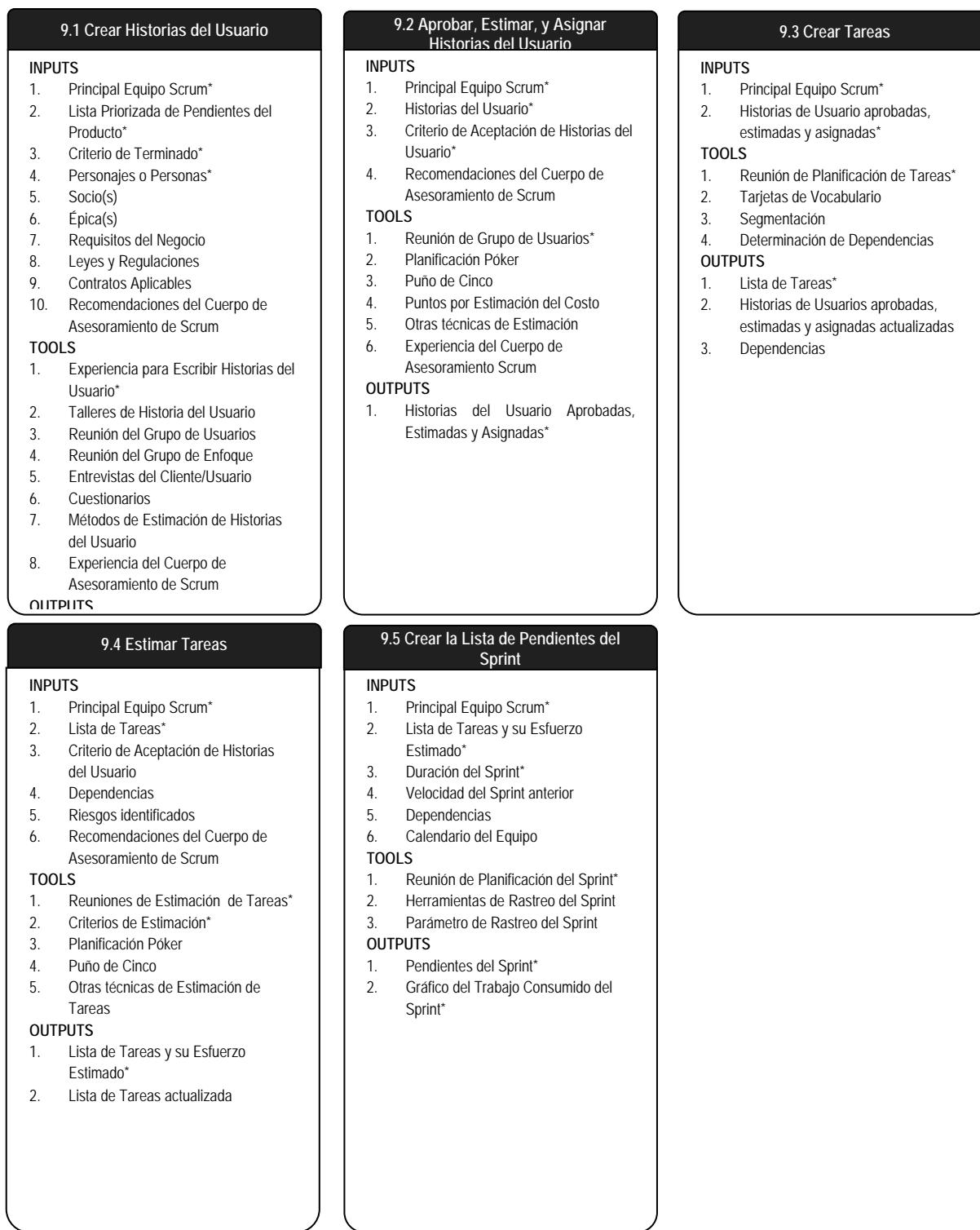
9.1 Elaborar historias de usuario—En este proceso, *User Stories* y sus afines *User Story Acceptance Criteria* se crean. Los *User Stories* son generalmente escritos por el *Propietario del producto* y están diseñados para asegurar que los requisitos del *Customer* estén claramente representados, y que puedan ser plenamente comprendidos por todos los *socios*. *User Story Writing Workshops* se pueden llevar a cabo lo cual implica que los miembros del *Equipo Scrum* creen *User Stories*. Estos *User Stories* se incorporan en el *Prioritized Product Backlog*.

9.2 Aprobar, estimar y asignar historias de usuarios—En este proceso, el *Propietario del producto* aprueba los *User Stories* para un *Sprint*. Luego, el *Scrum Master* y el *Equipo Scrum* estiman el esfuerzo necesario para desarrollar la funcionalidad descrita en cada *User Story*. Por último, el *Equipo Scrum* se compromete a entregar los requisitos del *Customer* mediante *Aprobar, estimar y asignar historias de usuarios*.

9.3 Elaboración de tareas—En este proceso, los *Approved, Estimated, and Committed User Stories* se dividen en tareas específicas y se compilan en un *Task List*. A menudo, un *Task Planning Meeting* se lleva a cabo con este fin.

9.4 Estimar tareas—En este proceso, el *Scrum Core Team*, en las reuniones de *Task Estimation*, estima el esfuerzo necesario para realizar cada tarea del *Task List*. El resultado de este proceso es un *Effort Estimated Task List*.

9.5 Elaboración de la lista de pendientes del Sprint—En este proceso, el *Equipo Scrum* tiene un *Sprint Planning Meeting* donde el grupo crea un *Sprint Backlog* que contiene todas las tareas que deben completarse en el *Sprint*.

Figura 9-1: *Plan and Estimate General*

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida "obligatoria" para el proceso correspondiente.

La figura 9-2 muestra las entradas obligatorias, las herramientas y las salidas de los procesos en la fase *Plan and Estimate*.

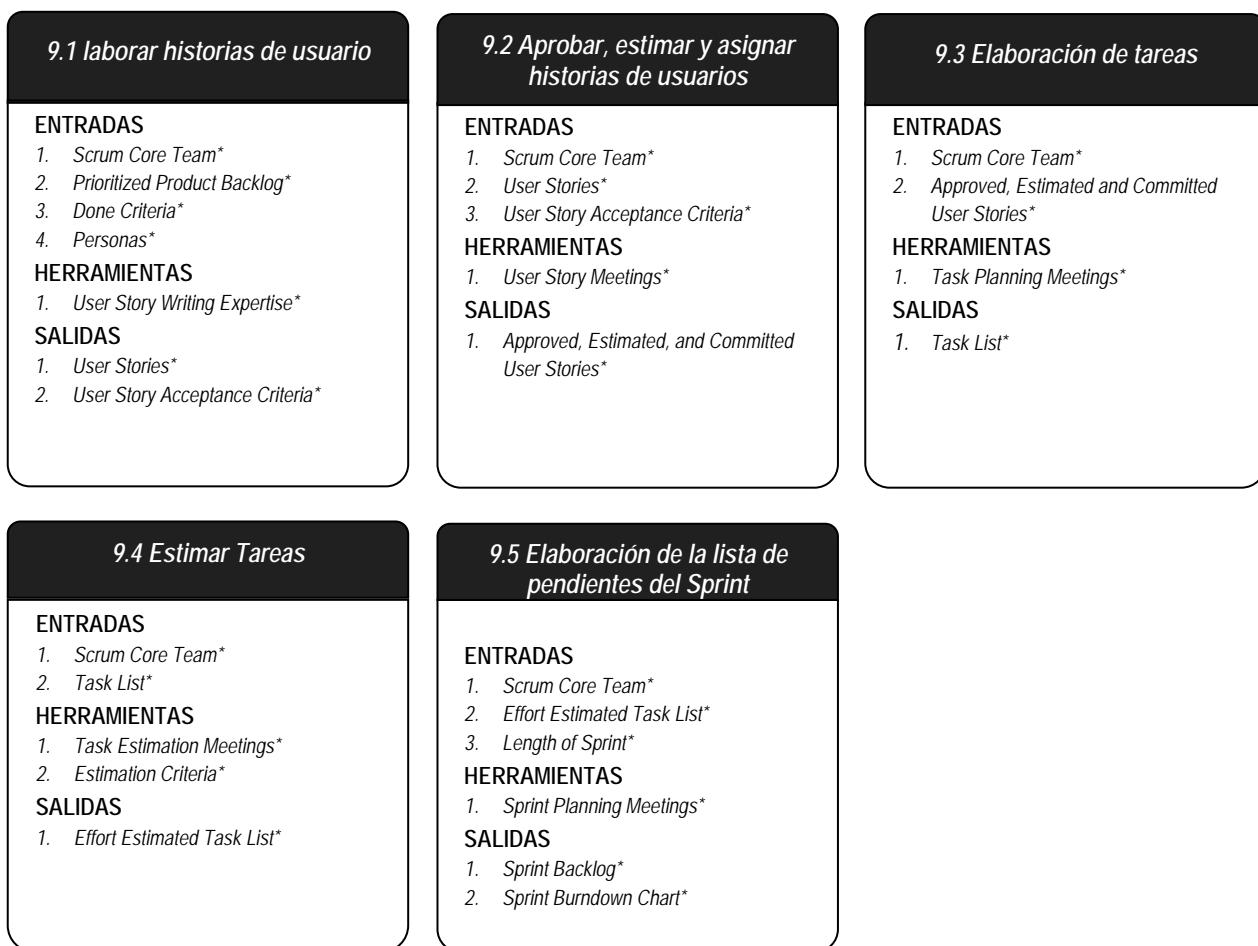


Figura 9-2: *Plan and Estimate* General (Esenciales)

9.1 *Elaborar historias de usuario*

La figura 9-3 muestra todas las entradas, las herramientas y las salidas para *Elaborar historias de usuario*.

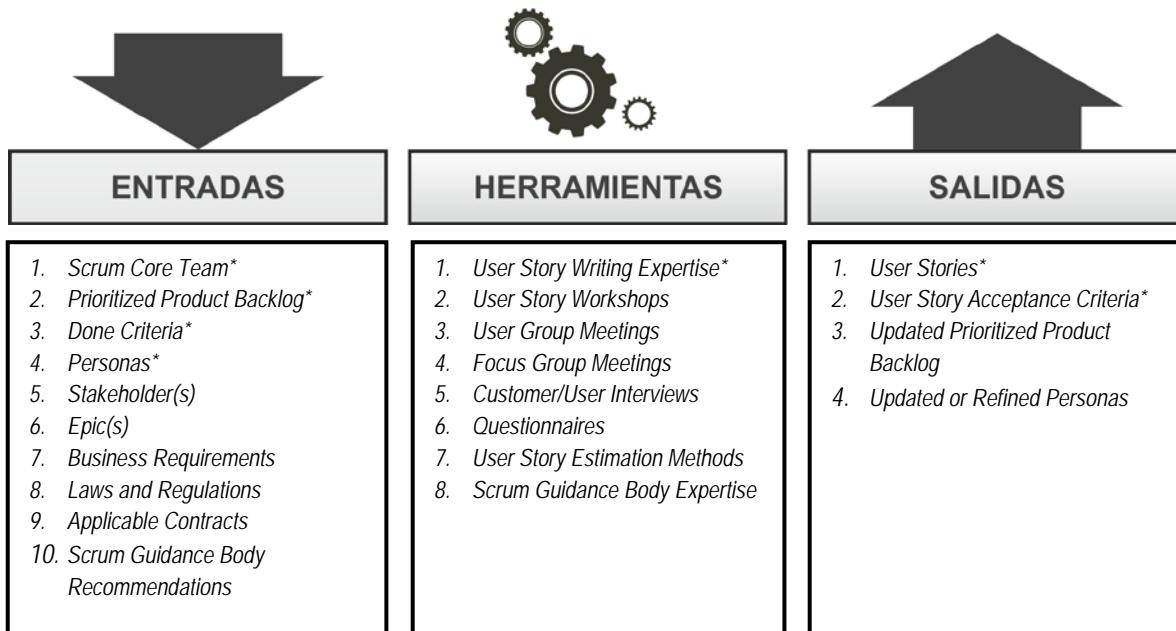
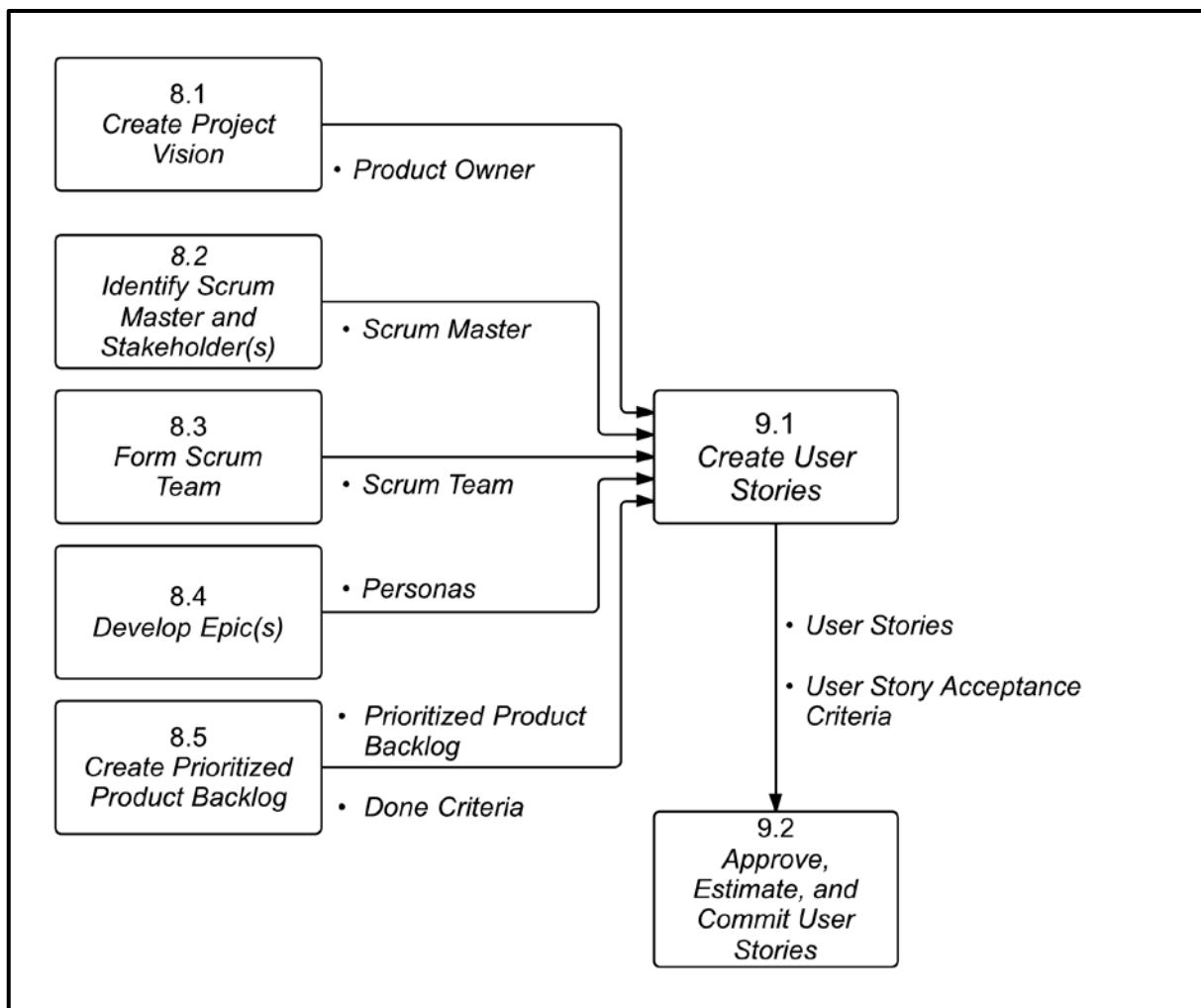


Figura 9-3: *Elaborar historias de usuario*—Entradas, Herramientas, y Salidas

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

Figura 9-4: *Elaborar historias de usuario—Diagrama de Flujo de Datos*

9.1.1 Entradas

9.1.1.1 *Scrum Core Team**

Describo en la sección 8.4.1.1.

9.1.1.2 *Prioritized Product Backlog**

Describo en la sección 8.5.3.1.

9.1.1.3 *Done Criteria**

Describo en la sección 8.5.3.2.

9.1.1.4 *Personas**

Describo en la sección 8.4.3.2.

9

9.1.1.5 *Stakeholder(s)*

Describo en la sección 8.2.3.2.

9.1.1.6 *Epic(s)*

Describo en la sección 8.4.3.1.

9.1.1.7 *Business Requirements*

Describo en la sección 8.5.1.7.

9.1.1.8 Leyes y Reglamentos

Describo en la sección 8.4.1.8.

9.1.1.9 Los contratos aplicables

Descrito en la sección 8.4.1.9.

9.1.1.10 Recomendaciones de *Cuerpo de asesoramiento de Scrum*

Descrito en la sección 8.1.1.12.

En el proceso *Elaborar historias de usuario* las recomendaciones pueden incluir información sobre las normas, los reglamentos y las mejores prácticas que se requieren para crear *User Stories* eficaces.

9.1.2 Herramientas

9.1.2.1 *User Story Writing Expertise**

El *Propietario del producto*, basado en su interacción con los *socios*, conocimiento del negocio, experiencia y las aportaciones del equipo, desarrolla los *User Stories* que formarán el *Prioritized Product Backlog* inicial para el proyecto. El *Prioritized Product Backlog* representa la suma total de lo que debe ser completado para el proyecto. El objetivo de este ejercicio es crear *User Stories* elaborados y refinados que sean aprobados y calculados, y a los cuales el *Equipo Scrum* se pueda comprometer. A veces, el *Propietario del producto* puede traer un analista de negocios para ayudar con la escritura de *User Stories*.

Aunque el *Propietario del producto* tiene la responsabilidad primordial de la escritura de *User Stories*, y a menudo lleva a cabo este ejercicio por sí solo, un *User Story Writing Workshop* puede ser considerado si se desea.

9.1.2.2 *User Story Workshops*

Descrito en la sección 8.4.2.2.

9.1.2.3 *User Group Meetings*

Descrito en la sección 8.4.2.1.

9.1.2.4 *Focus Group Meetings*

Focus Group Meetings es una técnica cualitativa para medir y entender las necesidades de los usuarios y las expectativas acerca de un producto propuesto. Un pequeño grupo de usuarios es seleccionado para formar el grupo de enfoque. Este grupo puede ser seleccionado al azar de un conjunto grande de usuarios, o se puede seleccionar específicamente para representar a todos las *Personas* en quienes se quieren enfocar. *Focus Group Meetings* normalmente se adhieren a un determinado formato en el que al grupo se le hacen preguntas que luego discuten entre ellos. Cada reunión del grupo de enfoque puede tener sus propias reglas de discusión a lo decidido por los organizadores. Estas reuniones se llevan a cabo generalmente en presencia de un moderador.

9.1.2.5 Entrevistas con el *Customer* o *Usuario*

Descrito en la sección 8.4.2.4.

9.1.2.6 Questionarios

Descrito en la sección 8.4.2.5.

9

9.1.2.7 *User Story Estimation Methods*

Todas las herramientas que se utilizan para el proceso de *Aprobar, estimar y asignar historias de usuarios* (como se describe en la sección 9.2.2) se pueden usar para crear estimaciones de alto nivel de los *Epic(s)* cuando creamos el *Prioritized Product Backlog*. Algunas herramientas importantes son:

1. *User Group Meetings*
2. *Planning Poker*
3. *Fist of Five*
4. *Points for Cost Estimation*
5. Otras técnicas de estimación

9.1.2.8 *Cuerpo de asesoramiento de Scrum Expertise*

Descrito en la sección 8.4.2.7.

Al crear *User Stories*, el *Cuerpo de asesoramiento de Scrum Expertise* podría referirse a las normas y reglamentos documentados; o los estándares y mejores prácticas para la creación de *User Stories*.

También puede haber un equipo de expertos en la materia que pueda ayudar al *Propietario del producto* o proporcionar orientación sobre como crear *User Stories*. Este equipo podría incluir analistas de negocios, arquitectos, desarrolladores, expertos en Scrum, al igual que otras personas con experiencia. Generalmente, este grupo de expertos no es el mismo equipo que estará envuelto y trabajará en este proyecto ya que tiende a pasar de grupo en grupo para orientar a los *Scrum Teams*.

9.1.3 Salidas

9.1.3.1 *User Stories**

Los *User Stories* se adhieren a una estructura específica y predefinida, y es una manera simplista de la documentación de los requisitos y de la funcionalidad deseada del usuario final. Un *User Story* indica tres cosas acerca de la exigencia: ¿Quién, qué y por qué? Los requisitos expresados en *User Stories* son declaraciones breves, simples y fáciles de entender. Los resultados de formato estándar predefinidos resultan en una mejor comunicación entre los *socios* y mejores cálculos determinados por el equipo. Algunos *User Stories* pueden ser demasiado grandes para manejar dentro de un sólo *Sprint*. Estos *User Stories* grandes a menudo se llaman *Epics*. Una vez que los *Epics* aparecen en el *Prioritized Product Backlog* para ser completados en el próximo *Sprint*, se hacen más pequeños y se convierten en *User Stories*.

El *Prioritized Product Backlog* es una lista dinámica que se actualiza de forma continua debido a *reprioritization* y a *User Stories* nuevos, actualizados, refinados, y a veces, borrados. Estos cambios son normalmente los resultados del cambio de *business requirements*.

Consulte también la sección 8.5.3.1 para saber más sobre el *Prioritized Product Backlog*.

Formato de User Story:

Como <rol/ persona>, yo debería <requisito> así <beneficio>.

Ejemplo de User Story:

Como administrador de banco de datos, debería revertir una cantidad específica de actualizaciones de base de datos para que la versión deseada de ésta se restaure.

9.1.3.2 *User Story Acceptance Criteria**

Cada *User Story* está asociado con un *User Story Acceptance Criteria*. Los *User Stories* son subjetivos, por lo que los *Acceptance Criteria* proporcionan la objetividad requerida para que el *User Story* sea considerado como *Done*, o no, durante el *Sprint Review*. Los *Acceptance Criteria* le proporcionan claridad al equipo sobre el *User Story*, eliminan la ambigüedad de los requisitos y ayudan en la alineación de expectativas. El

Product Owner define y le comunica los *Acceptance Criteria* al *Equipo Scrum*. En el *Prioritized Product Backlog*, los *Acceptance Criteria* proporcionan el contexto para que el *Propietario del producto* decida si un *User Story* se ha completado satisfactoriamente. Es importante, y la responsabilidad del *Scrum Master*, asegurar que el *Propietario del producto* no cambie los *Acceptance Criteria* de un *User Story* que ya está comprometido a un *Sprint*.

9.1.3.3 Prioritized Product Backlog actualizado

El *Prioritized Product Backlog* creado en el proceso *Creación de la lista priorizada de pendientes del producto* se actualiza con información sobre el *User Story*, *Epic(s)*, las estimaciones de *User Stories* y *User Story Acceptance Criteria*.

Prioritized Product Backlog se describe en el apartado 8.5.3.1.

9.1.3.4 Personas actualizadas y refinadas

Las *Personas* se crean inicialmente en el proceso llamado *Desarrollo de épica(s)*. Mientras se escribe un *User Stories*, el *Equipo Scrum* puede llegar a una decisión colectiva de que algunas de esas *Personas* iniciales son insuficientes y necesitan refinamiento. Si se requiere la refinación de *Personas*, esto normalmente se realiza casi al final del proceso *Elaborar historias de usuario*.

Personas se describe en el apartado 8.4.3.2.

9.2 Aprobar, estimar y asignar historias de usuarios

La figura 9-5 muestra todas las entradas, las herramientas y las salidas para el proceso *Aprobar, estimar y asignar historias de usuarios*.



Figura 9-5: *Aprobar, estimar y asignar historias de usuarios*—Entrada, Herramientas y Salidas

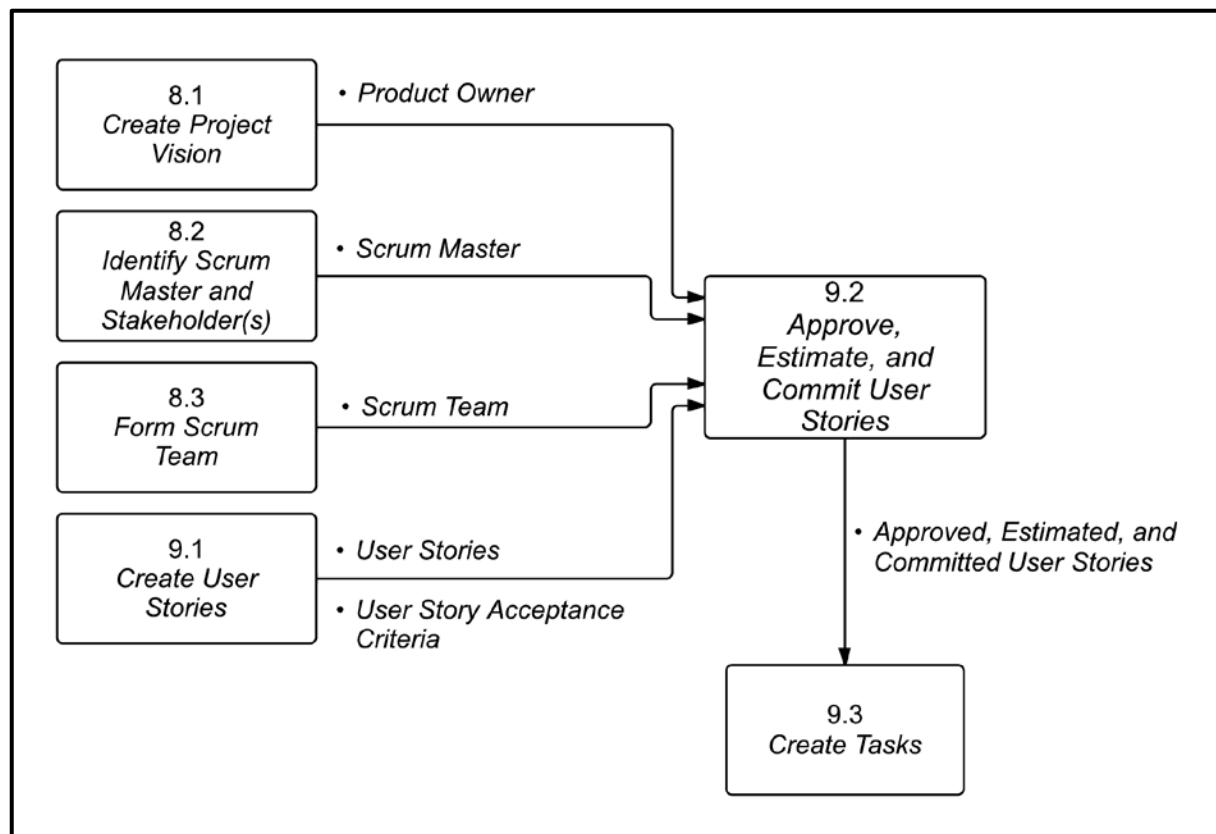


Figura 9-6: *Aprobar, estimar y asignar historias de usuarios*— Diagrama de flujo de datos

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

9.2.1 Entradas

9.2.1.1 *Scrum Core Team**

Descrito en la sección 8.4.1.1.

9.2.1.2 *User Stories**

Descrito en la sección 9.1.3.1.

Los *User Stories* tienen estimaciones de alto nivel de los procesos *Creación de la lista priorizada de pendientes del producto* y *Elaborar historias de usuario*. Estas estimaciones serían utilizadas por el *Propietario del producto* para crear una lista de *User Stories* aprobados que se estiman con mayor precisión por el *ScrumTeam*. Tales *User Stories* son entonces comprometidos a ser completados por el *Equipo Scrum* en el *Sprint*.

9.2.1.3 *User Story Acceptance Criteria**

9

Descrito en la sección 9.1.3.2.

9.2.1.4 Recomendaciones del Cuerpo de asesoramiento de Scrum

Descrito en la sección 8.1.1.12.

En el proceso *Aprobar, estimar y asignar historias de usuarios*, el *Cuerpo de asesoramiento de Scrum Recommendations* pueden incluir información sobre las normas, los reglamentos, las reglas y las mejores prácticas necesarias para *Aprobar, estimar y asignar historias de usuarios* de forma efectiva.

9.2.2 Herramientas

9.2.2.1 *User Group Meetings**

Descrito en la sección 8.4.2.1.

9.2.2.2 *Planning Poker*

Planning Poker, también llamada *Estimation Poker*, es una técnica de estimación o cálculo que utiliza el consenso para estimar los tamaños relativos de los *User Stories* o el esfuerzo requerido para crearlos.

En *Planning Poker*, a cada miembro del equipo se le asigna una baraja de cartas. Cada tarjeta tiene un número en una secuencia y los números representan la complejidad del problema, en términos de tiempo y esfuerzo, según lo estimado por el miembro del equipo. El *Propietario del producto* elige un *User Story* en el *Prioritized Product Backlog* y lo presenta al equipo. Los miembros del *Equipo Scrum* evalúan el *User Story* y tratan de entenderlo mejor antes de proporcionar su estimación para desarrollarlo. Luego, cada miembro toma una carta de la baraja que representa su estimación para el *User Story*. Si la mayoría o todos los miembros del equipo seleccionan la misma tarjeta, entonces la estimación indicada por esa tarjeta será la estimación del *User Story*. Si no hay consenso, los miembros del equipo discuten las razones para seleccionar diferentes tarjetas o estimaciones. Después de esta discusión se agarran cartas de nuevo. Esta secuencia continúa hasta que se entiendan todos los supuestos, se resuelvan los malentendidos, y se llegue a un consenso o acuerdo.

Planning Poker aboga por una mayor interacción y una mejor comunicación entre los participantes. Facilita el pensamiento independiente por los participantes, evitando así el fenómeno de pensamiento grupal.

9.2.2.3 *Fist of Five*

Fist of Five es un mecanismo sencillo y rápido para llegar a un consenso en un grupo e iniciar una conversación. Tras el debate inicial sobre una propuesta o una decisión pendiente, se les pide a los miembros del *Equipo Scrum* que voten en una escala de 1 a 5 usando sus dedos. El valor en el uso de esta técnica no es sólo la creación de consenso, sino también la reflexión y charla, ya que cada miembro del equipo se le pide que explique el motivo de su clasificación. También se les da la oportunidad de expresar cualquier *issues* o preocupación. Una vez que el equipo ha discutido, se tomará una decisión colectiva.

El número de dedos que se utiliza para la votación indica el nivel de acuerdo y el deseo para el debate:

1. Un dedo: no estoy de acuerdo con la conclusión del grupo y tienen grandes preocupaciones.
2. Dos dedos: no estoy de acuerdo con la conclusión del grupo y me gustaría hablar de algunos *issues* menores.
3. Tres dedos: no estoy seguro y me gustaría asumir la conclusión de consenso del grupo.
4. Cuatro dedos: Estoy de acuerdo con la conclusión del grupo y me gustaría discutir algunos *issues* menores.
5. Cinco dedos: Estoy totalmente de acuerdo con la conclusión del grupo.

9.2.2.4 *Points for Cost Estimation (Puntos para la estimación del costo)*

La estimación de costos se puede lograr mediante el uso de unidades relativas (por ejemplo, las estimaciones de esfuerzo) en lugar de unidades absolutas (es decir, los costos reales incurridos). Con el fin de estimar el costo de implementar un *User Story*, el *Equipo Scrum* puede utilizar *story points*. Cuando se hace esto, el costo estimado para cada tarea será en forma de *story points*, en lugar de unidades monetarias. Con el fin de hacer esto con éxito, el *Equipo Scrum* debe identificar un *User Story* de base del cual todos los miembros del equipo pueden basarse. Una vez que esta línea de base se identifica, todas las

estimaciones de costos para los *User Stories* se deben hacer en comparación con la línea base. Estas estimaciones se mantienen fijas a lo largo de un *Sprint* porque los equipos no deben cambiar durante un *Sprint*.

9.2.2.5 Otras técnicas de estimación

9.2.2.5.1 *Wideband Delphi*

Wideband Delphi es una técnica de estimación basada en grupo para la determinación de la cantidad de trabajo que está involucrado y el tiempo que tardará en completarse. Los individuos de un equipo de forma anónima proporcionan estimaciones para cada función y las estimaciones iniciales se trazan en una gráfica. Posteriormente, el equipo analiza los factores que influyeron en sus estimaciones y proceden a una segunda ronda de estimación. Este proceso se repite hasta que las estimaciones de los individuos sean similares y se llegue a un consenso para la estimación final.

Planning Poker (tal como se describe en la sección 9.2.2.2) es un ejemplo de una *Wideband Delphi Technique*. También es importante tener en cuenta que es la entrada (*input*) individual obtenida por un mecanismo que evita el pensamiento de grupo. Luego, las entradas individuales se utilizan para una decisión de grupo.

9

9.2.2.5.2 *Relative Sizing/Story Points*

Además de ser utilizado para la estimación de costos, *story points* también se puede utilizar para estimar el tamaño total de un *User Story*. Este procedimiento consiste en atribuir un valor en *story points* basado en una evaluación general del tamaño de un *User Story* considerando el riesgo, la cantidad de esfuerzo que se requiere, y el nivel de complejidad. Esta evaluación se llevará a cabo por el *Equipo Scrum* y un valor de *story point* se le asignará. Una vez que la evaluación se realiza en un *User Story* en el *Prioritized Product Backlog*, el Equipo Scrum puede entonces evaluar otros *User Stories* en relación a la primera historia. Por ejemplo, una característica con un valor de historia de 2-puntos debe ser dos veces tan difícil de completar como una característica con una historia de 1 punto; una historia de 3 puntos debe ser tres veces más difíciles de completar como una historia de 1 punto.

9.2.2.5.3 *Affinity Estimation*

Affinity Estimation es una técnica utilizada para estimar rápidamente un gran número de *User Stories*. Usando notas adhesivas o *Index Cards* y cinta, el equipo coloca *User Stories* en la pared u otro lugar, en orden de menor a mayor. Para hacer esto, cada miembro del equipo comienza con un subconjunto de *User Stories* del *Prioritized Product Backlog* para colocar basado en el tamaño. Esta colocación inicial se hace en silencio. Una vez que todos han puesto sus *User Stories* en la pared, el equipo revisa todas las colocaciones y puede mover los *User Stories* si piensan es apropiado. Esta segunda parte del ejercicio consiste en la discusión. Por último, el *Propietario del producto* indicará algunas categorías de tamaño observadas en la pared. Estas categorías pueden ser pequeñas, medianas o grandes, o pueden ser

numeradas usando los valores de *story points* para indicar el tamaño relativo. El equipo desplazará los *User Stories* en estas categorías como el paso final en el proceso. Algunos de los beneficios claves de este enfoque es que el proceso es muy transparente, visible para todo el mundo, y es fácil de llevar a cabo.

9.2.2.5.4 *Estimate Range*

Las estimaciones para los projectss deben ser presentadas en rangos. Las cifras exactas pueden dar la impresión de ser muy precisas cuando en realidad puede que no lo sean. De hecho, las estimaciones, por definición, se entienden que no son precisamente exactas. *Estimate ranges* deben basarse en el nivel de confianza que el equipo tiene en cada estimación. El rango puede ser estrecho, cuando el equipo está confiado y amplio cuando el equipo se siente menos seguro.

9.2.2.6 *Cuerpo de asesoramiento de Scrum Expertise*

Descrito en la sección 8.4.2.7.

Los conflictos con respecto a las estimaciones para completar ciertos *User Stories* pueden surgir durante este proceso porque las perspectivas de los miembros del equipo pueden ser diferentes y porque el equipo aún puede no tener suficiente experiencia para estimar *Sprints*. En estas situaciones, la experiencia y los conocimientos especializados del *Cuerpo de asesoramiento de Scrum Expertise* pueden ayudar a resolver conflictos.

9.2.3 Salidas

9.2.3.1 *Approved, Estimated, and Committed User Stories**

Los *User Stories*, que son aportes a este proceso tienen estimaciones de alto nivel de los procesos *Creación de la lista priorizada de pendientes del producto* y *Elaborar historias de usuario*. Estas estimaciones son utilizadas por el *Propietario del producto* para aprobar *User Stories* para el *Sprint*.

Debe tenerse en cuenta que es la responsabilidad del *Propietario del producto* asegurar que aprobaron *User Stories* que entregan valor y satisfacen las necesidades y requerimientos de los proyectos de los socios. Una vez aprobados, los *User Stories* se estiman por el equipo utilizando las distintas técnicas de estimación tratadas en esta sección. Después de la estimación, el equipo se compromete en un subconjunto de *User Stories* aprobados y estimados que creen que pueden completar en el próximo *Sprint*. Estos *User Stories* son *Approved, Estimated, and Committed User Stories* que se convertirán en parte del *Sprint Backlog*.

Aunque el *Propietario del producto* aprueba los *User Stories* iniciales para un *Spirnt*, la decisión final sobre qué *User Stories* específicos (entre los aprobados por el *Propietario del producto*) deben ser elegidos para

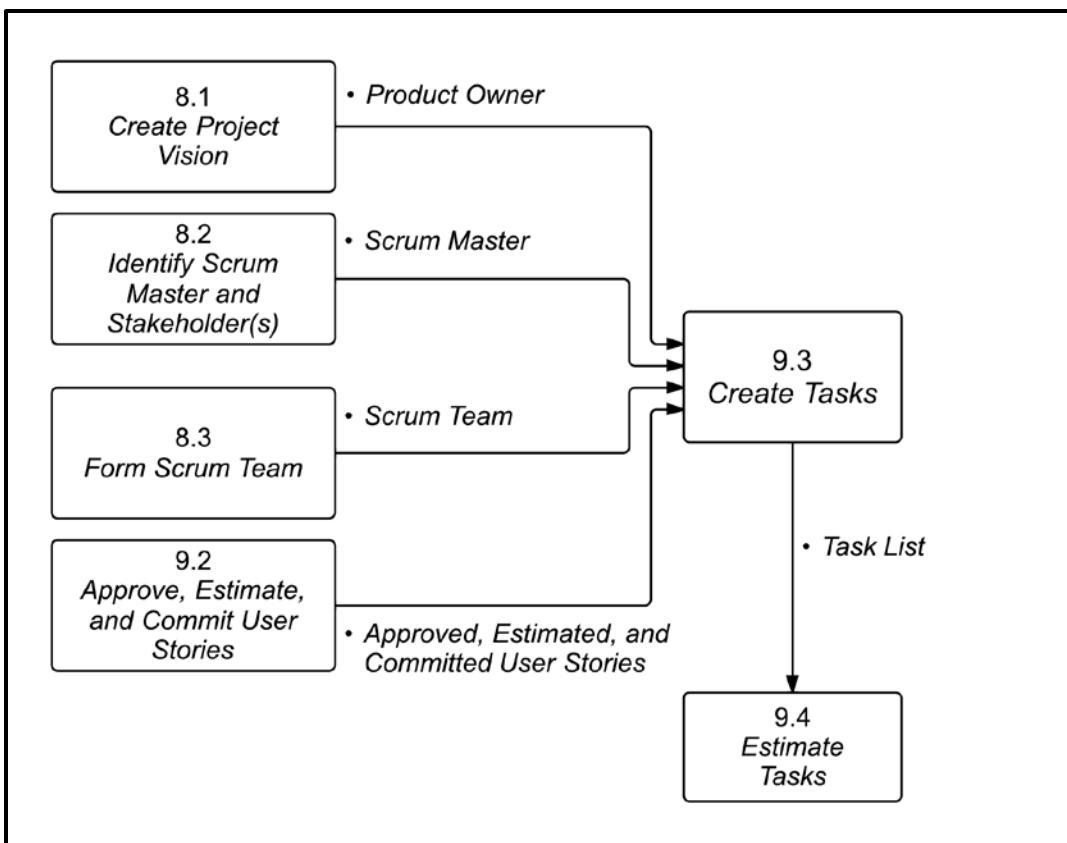
el *Sprint* recae en el *Equipo Scrum*. El *Equipo Scrum* (en consulta del *Propietario del producto*, si es necesario) finaliza los *User Stories* en los que van a trabajar durante el *Sprint*.

9.3 *Elaboración de tareas*

La figura 9-7 muestra todas las entradas, las herramientas y las salidas del proceso *Elaboración de tareas*.



Figura 9-7: *Elaboración de tareas—Entradas, Herramientas y Salidas*

Figura 9-8: *Elaboración de tareas—Diagrama de Flujo de Dato*

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

9.3.1 Entradas

9.3.1.1 Scrum Core Team*

Descritas en las secciones 8.4.1.1.

9.3.1.2 Approved, Estimated, and Committed User Stories*

Describo en la sección 9.2.3.1.

9.3.2 Herramientas

9.3.2.1 Task Planning Meetings*

En *Task Planning Meetings*, el *Equipo Scrum* se reúne para planificar el trabajo que se hará en el *Sprint*. El equipo revisa los *User Stories* ya comprometidos en la parte superior del *Prioritized Product Backlog*. El *Propietario del producto* está presente en esta reunión en caso de que se requiera aclaración en relación a los *User Stories* en el *Prioritized Product Backlog*, y para ayudar al equipo a tomar decisiones de diseño. Para ayudar a asegurar que el grupo se concentre en el tema, esta reunión será *time-boxed*, con la longitud estándar limitada a dos horas por semana durante el *Sprint*. Esto ayuda a evitar la tendencia a desviarse en discusiones que deben ocurrir realmente en otras reuniones, como las reuniones de *Release Planning* o *Sprint Review Meetings*. Al final de la reunión, el *Equipo Scrum* se ha comprometido plenamente a proporcionar un subconjunto de los *User Stories* en el *Task Planning Meeting* en el *Sprint*.

Task Planning Meeting se divide normalmente en dos secciones, con un fin determinado y amplia agenda para cada uno (ver Figura 9-9)

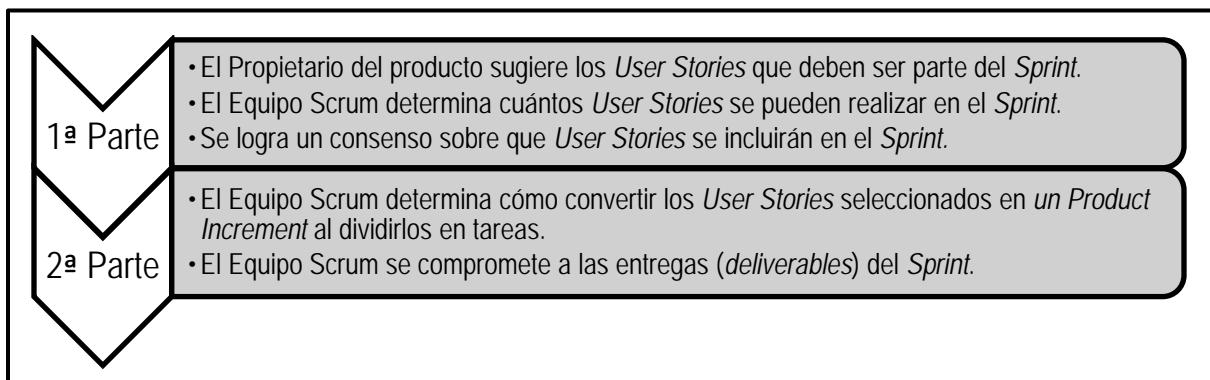


Figura 9-9: *Task Planning Meetings*

Task Planning Meetings a veces también se conoce como *Sprint Planning Meetings* - esas reuniones también se pueden combinar con los *Task Estimation Meetings* como se describe en el apartado 9.4.2.1

9.3.2.2 Index Cards

En Scrum, los *User Stories* se escriben en pequeños *Index Cards*. Sólo los detalles esenciales están documentados en las tarjetas, que pueden ser utilizados por el *Equipo Scrum* para colaborar y discutir. Estos *Index Cards*, a menudo descrito como *story cards*, aumentan la visibilidad y la transparencia, y facilitan la detección temprana de cualquier problema que pueda surgir.

9.3.2.3 Descomposición

Descomposición es una herramienta mediante la cual las tareas de alto nivel se dividen en tareas detalladas con niveles más bajos. Los *User Stories* se reducen en tareas por los miembros del *Equipo Scrum*. Los *Prioritized Product Backlog User Stories* se deben reducir suficientemente a un nivel que le proporcione al *Equipo Scrum* información adecuada para *Crear entregables* de las tareas mencionadas en el *Task List*.

9.3.2.4 Dependency Determination

Una vez que el *Equipo Scrum* ha seleccionado *User Stories* para un determinado *Sprint*, debería entonces considerar las dependencias, incluyendo las relacionadas con la disponibilidad de las personas, así como las dependencias técnicas. El documentar adecuadamente las dependencias ayuda a los *Equipos Scrum* a determinar el orden relativo en el que las tareas deben ejecutarse para crear el *Sprint Deliverables*. Las dependencias también destacan la relación e interacción entre las tareas tanto en el *Equipo Scrum* que trabajan en un determinado *Sprint* con otros *Equipos Scrum* en el proyecto.

Existen numerosos tipos de dependencias: obligatorias y discretionales, internas y externas, o alguna combinación de estas dependencias. Por ejemplo, una dependencia puede ser tanto obligatoria y externa.

- **Mandatory dependencies**—Dependencias que son ya sea inherente a la naturaleza del trabajo, como una limitación física, o puedan deberse a las obligaciones contractuales o los requisitos legales. Por ejemplo, el trabajo en el primer piso no puede comenzar hasta que el cimiento de la construcción se haya completado. *Mandatory Dependencies* también se describen comúnmente como *hard logic*.
- **Discretionary dependencies**—Dependencias que se colocan en el flujo de trabajo por decisión propia. Normalmente, *Discretionary Dependencies* son determinados por el *Equipo Scrum*, basadas en las experiencias del pasado o las mejores prácticas en un campo o dominio en particular. Por ejemplo, el equipo puede decidir completar una tarea antes de trabajar en otra, ya que es una buena práctica, pero no es obligatorio. Por ejemplo, el equipo puede optar por construir los marcos de puertas y ventanas antes de que la estructura completa de la pared esté en su lugar.
- **External dependencies**—*External dependencies* son las las tareas, actividades o productos que están fuera del alcance del *Equipo Scrum*, pero son necesarios para completar una tarea de proyecto o crear una prestación del proyecto. *External dependencies* están por lo general fuera del control del *Equipo Scrum*. Por ejemplo, si el *Equipo Scrum* no es responsable de la adquisición de los materiales necesarios para la construcción de las paredes, entonces, los materiales y las tareas relacionadas con su adquisición se consideran *external dependencies*.
- **Internal dependencies**—*Internal Dependencies* son las dependencias de tareas, productos o actividades que están bajo el control del *Equipo Scrum*. Por ejemplo, la instalación de paneles de yeso debe ser completada antes de pintar la pared. Este es un ejemplo de una dependencia interna debido a que ambas tareas son parte del proyecto. En este caso, también es obligatorio, ya que se basa en una limitación física. No es posible pintar la pared antes qu la pared esté seca.

9.3.3 Salidas

9.3.3.1 Task List*

Esta es una lista completa con todas las tareas a la que el *Equipo Scrum* se ha comprometido para el *Sprint* corriente. Contiene descripciones de cada tarea junto con las estimaciones derivadas durante el proceso de *Elaboración de tareas*. *Task List* debe incluir cualquier esfuerzo de prueba y de integración de manera que el *Product Increment* del *Sprint* se pueda integrar con éxito en las entregas anteriores de *Sprints*.

A pesar de que las tareas son a menudo basadas en actividades, el nivel de detalle al que las tareas se descomponen se decide por el *Equipo Scrum*.

9.3.3.2 *Approved, Estimated, and Committed User Stories* actualizados

Los *User Stories* se actualizan durante este proceso. Las actualizaciones pueden incluir revisiones de las estimaciones de *User Stories* originales, basadas en creaciones de tareas y complejidad discutidas durante el *Sprint Planning Meeting*. *Approved, Estimated, and Committed User Stories* se describen en la sección 9.2.3.1.

9.3.3.3 Dependencias

Las *Dependencias* describen la relación e interacción entre diferentes tareas de un proyecto y pueden ser clasificadas como obligatorias o discretionales; o interna o externa; como se discutió en la sección 9.3.2.4.

Hay numerosas maneras de identificar, definir y presentar las tareas y sus dependencias. Dos métodos comunes involucran el uso de diagramas de flujo del producto y diagramas de Gantt.

9.4 Estimar tareas

La figura 9-10 muestra todas las entradas, las herramientas y las salidas para el proceso de *Estimate Task*.



Figura 9-10: *Estimar tareas*—Entradas, Herramientas y Salidas

9

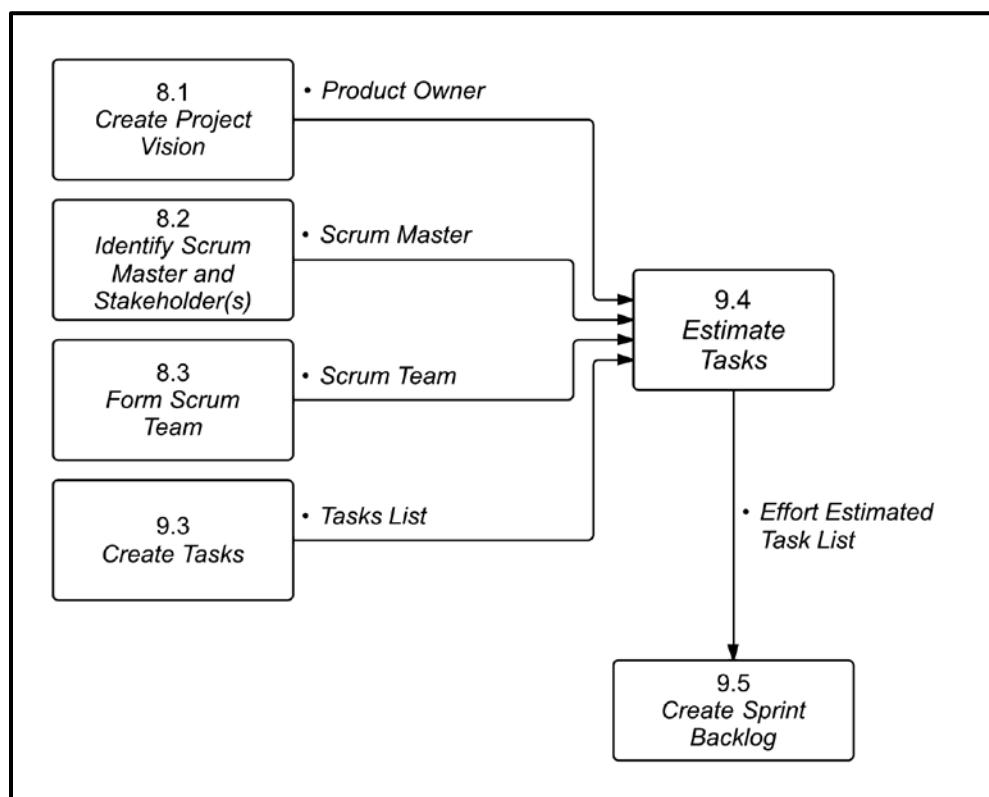


Figura 9-11: *Estimar tareas*—Diagrama de Flujo de Datos

Nota: Los asteriscos () denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.*

9.4.1 Entradas

9.4.1.1 *Scrum Core Team**

Descrito en la sección 8.4.1.1.

9.4.1.2 *Task List**

Descrito en la sección 9.3.3.1.

9.4.1.3 *User Story Acceptance Criteria*

Descrito en la sección 9.1.3.2.

El *Equipo Scrum* debe asegurarse de que los *Acceptance Criteria* definidos sean apropiados para los *User Stories* y proporcionen claridad en cuanto a los requisitos para el *Equipo Scrum*. Las pruebas de aceptación se refieren a la evaluación de la capacidad de la entrega completa para satisfacer los *Acceptance Criteria*. Esto proporciona información para el *Propietario del producto* para ayudar a tomar una decisión acerca de la aprobación o el rechazo de la entrega.

En el desarrollo de *User Story Acceptance Criteria*, lo siguiente debe ser considerado:

- *Acceptance Criteria* no deben ser vagos, ambiguos o demasiado generalizados.
- *Defined Acceptance Criteria* aseguran de que el equipo sea capaz de verificar que los resultados estén alineados con las metas y objetivos de la organización patrocinadora.

9.4.1.4 Dependencias

Descrito en la sección 9.3.3.3

9.4.1.5 *Risks identificados*

Descrito en la sección 8.4.3.4.

9.4.1.6 Recomendaciones del *Cuerpo de asesoramiento de Scrum*

Descrito en la sección 8.1.1.12.

En el proceso de *Estimar tareas*, *Cuerpo de asesoramiento de Scrum Recommendations* pueden incluir información sobre las normas, los reglamentos, las reglas y las mejores prácticas que se requieren para estimar con eficacia las tareas en el *Task Lists*.

9.4.2 Herramientas

9.4.2.1 *Task Estimation Meetings** (*Reuniones de estimación de trabajo*)

Task Estimation Meetings le permiten al *Equipo Scrum* estimar el esfuerzo necesario para completar una tarea o conjunto de tareas y estimar el esfuerzo de las personas y otros recursos necesarios para llevar a cabo los trabajos dentro de un *Sprint* determinado. En *Task Estimation Meetings*, los miembros del *Equipo Scrum* utilizan *Task List* para estimar la duración y el esfuerzo de los *User Stories* que se completarán en el *Sprint*.

9

Una de las principales ventajas de esta técnica es que permite que el equipo tenga una perspectiva compartida de los *User Stories* y los requisitos para que puedan estimar con fiabilidad el esfuerzo requerido. La información desarrollada en *Task Estimation Meetings* se incluye en el *Effort Estimated Task List* y se utiliza para determinar la velocidad del *Sprint*.

En este taller, el *Equipo Scrum* puede utilizar diversas técnicas como segmentación (*decomposition*), la opinión de expertos, estimación análoga, y la estimación paramétrica.

Task Estimation Meetings también se conoce como *Sprint Planning Meetings* - esas reuniones también se pueden combinar con *Task Planning Meetings*, como se describe en la sección 9.3.2.1.

9.4.2.2 *Estimation Criteria**

El objetivo principal de utilizar *Estimation Criteria* es mantener los tamaños de estimación relativos y minimizar la necesidad de re-estimación. *Estimation Criteria* se pueden expresar de muchas maneras, dos ejemplos comunes son los *story points* e *ideal time*. Por ejemplo, un momento ideal normalmente describe el número de horas que un miembro del *Equipo Scrum* trabaja exclusivamente en el desarrollo de los entregables del proyecto, sin incluir el tiempo dedicado a otras actividades o trabajos que están fuera del proyecto. *Estimation Criteria* hace que sea más fácil para el *Equipo Scrum* estimar el esfuerzo y les permita evaluar las ineficiencias de dirección cuando sea necesario.

9.4.2.3 *Planning Poker*

Descrito en la sección 9.2.2.2.

9.4.2.4 *Fist of Five*

Descrito en la sección 9.2.2.3.

9.4.2.5 Otras técnicas de estimación de tareas

Descrito en la sección 9.2.2.5.

9.4.3 Salidas

9.4.3.1 *Effort Estimated Task List**

Effort Estimated Task List es una lista de las tareas asociadas con los *User Stories* incluidos en un *Sprint*. Normalmente, la precisión de las estimaciones varía dependiendo de las habilidades de equipo. El esfuerzo estimado se expresa en términos de los *Estimation Criteria* acordados por el equipo. El *Effort Estimated Task List* es usado por el *Equipo Scrum* durante el *Sprint Planning Meetings* para crear el *Sprint Backlog* y *Sprint Burndown Chart*. También se utiliza para determinar cuando el equipo necesita reducir su compromiso, o cuando puede asumir *User Stories* adicionales durante el *Sprint Planning*.

9.4.3.2 *Updated Task List (Task List actualizados)*

Updated Task List, desarrollado como parte del proceso de *Elaboración de tareas*, incluye las estimaciones iniciales de *User Stories* que necesitan ser revisadas en base a las actividades de estimación más detalladas realizadas en el *Estimar tareas process*. También puede haber re-estimaciones resultantes de una revisión de los principios de *Sprints*, o un cambio en la comprensión colectiva del *Equipo Scrum* relacionado a los *User Stories* y los requisitos.

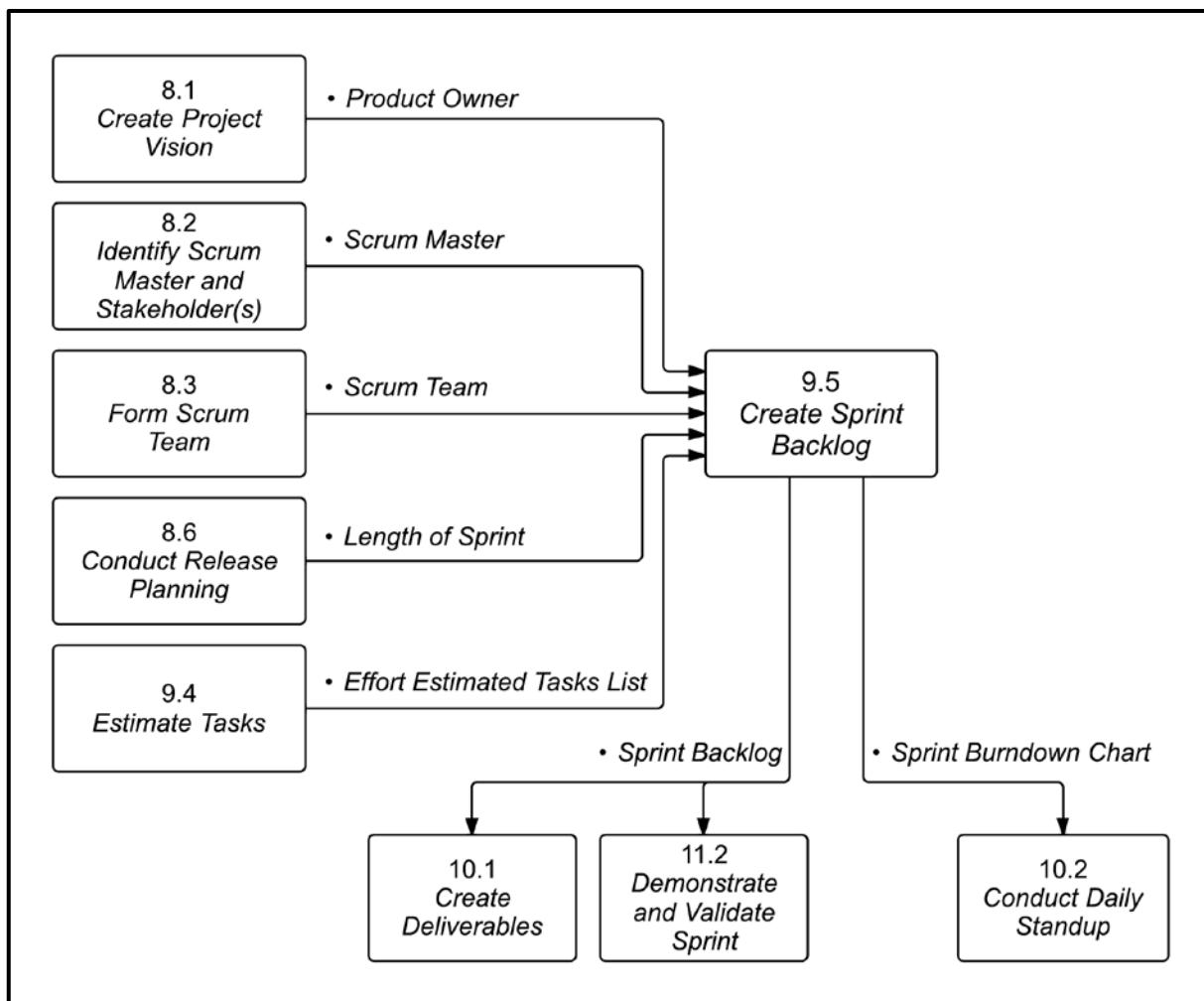
9.5 *Elaboración de la lista de pendientes del Sprint*

La figura 9-12 muestra todas las entradas, las herramientas y las salidas para el proceso *Elaboración de la lista de pendientes del Sprint*.



Figura 9-12: *Elaboración de la lista de pendientes del Sprint*—Entradas, Herramientas, y Salidas

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

Figura 9-13: *Elaboración de la lista de pendientes del Sprint—Diagrama de Flujo de Dato*

9.5.1 Entradas

9.5.1.1 *Scrum Core Team** (Equipo Principal de Scrum)

Descrito en la sección 8.4.1.1.

9.5.1.2 *Effort Estimated Task List**

Descrito en la sección 9.4.3.1.

9.5.1.3 *Length of Sprint**

Descrito en la sección 8.6.3.2.

9.5.1.4 *Sprint Velocity* previo

Sprint Velocity es la velocidad en la que el equipo pueda completar el trabajo en un *Sprint*. Por lo general se expresa en las mismas unidades que las utilizadas para la estimación, normalmente *story points* o *ideal time*. Se mantiene un registro del *Sprint Velocity* del equipo para cada *Sprint* y se utiliza como referencia en los siguientes *Sprints*. *Previous Sprint Velocity* se convierte en el factor más importante para determinar la cantidad de trabajo que el equipo podría lograr en un *Sprint* posterior. Cualquier cambio en la situación o las condiciones desde el último *Sprint* se tienen en cuenta para asegurar una estimación precisa del *Sprint Velocity* para el próximo *Sprint*.

9

9.5.1.5 Dependencias

Descrito en la sección 9.3.3.3.

9.5.1.6 Team Calendar

Team Calendar contiene información sobre la disponibilidad de los miembros del equipo, incluyendo la información correspondiente a las vacaciones de los empleados, fechas de licencia, acontecimientos importantes, y los días festivos.

Uno de los principales objetivos de la utilización de un *Team Calendar* es un seguimiento de lo que cada miembro del equipo está trabajando en todo el proyecto. Ayuda al equipo no sólo en la planificación y ejecución de *Sprints* eficientes, sino también en la alineación de *Sprints* con las fechas de lanzamiento.

9.5.2 Herramientas

9.5.2.1 *Sprint Planning Meetings**

Durante *Sprint Planning Meetings*, los *User Stories*, los cuales son aprobados, calculados, y con los cuales ya hay un compromiso, serán examinados por el *Equipo Scrum* durante los procesos de *Aprobar*, *estimar* y *asignar historias de usuarios*. Cada miembro del *Equipo Scrum* también utiliza *Effort Estimated Task List* para seleccionar las tareas en que planean trabajar en el *Sprint*, en base a sus habilidades y experiencia. El *Equipo Scrum* también crea el *Sprint Backlog* y *Sprint Burndown Chart* con los *User Stories* y el *Effort Estimated Task List* durante los *Sprint Planning Meetings*.

9.5.2.2 *Sprint Tracking Tools*

Es importante hacer un seguimiento del progreso de un *Sprint* para saber dónde está el *Equipo Scrum* en relación a las tareas del *Sprint Backlog*. Una variedad de herramientas se puede utilizar para realizar el seguimiento del trabajo en un *Sprint*, pero uno de los más comunes es un *Scrumboard*, también conocido como *task board* o *progress chart*. El *Scrumboard* se divide en secciones: *To Do* (a veces conocido como *Work not Started*), *Work in Progress*, y *Completed Work*. Se colocan notas adhesivas que representan a cada tarea o *User Story* en la categoría apropiada para reflejar el estado del trabajo. Estas notas adhesivas se mueven hacia adelante a la siguiente categoría a medida que progresa el trabajo.

9.5.2.3 *Sprint Tracking Métricas*

Las métricas utilizadas en proyectos Scrum incluyen *velocity*, *business value delivered* y *number of stories*.

Velocity—representa el número de *User Stories* o número de funcionalidades entregadas en un sólo *Sprint*.

Business value delivered—mide el valor de los *User Stories* entregados desde el punto de vista comercial.

Number of stories—se refiere a la cantidad de *User Stories* que se entregan como parte de un *Sprint*. Se puede expresar en términos de *simple count* o *weighted count*.

9.5.3 Salidas

9.5.3.1 Sprint Backlog*

La lista de las tareas a ser ejecutadas por el *Equipo Scrum* en el próximo *Sprint* se llama *Sprint Backlog*.

Es una práctica común que el *Sprint Backlog* se represente en un *Scrumboard* o tablero de tarea, que proporciona una representación visible constantemente de la situación de los *User Stories* en el backlog. También se incluyen en el *Sprint Backlog* algunos *risks* asociados con las diversas tareas. Todas las actividades de mitigación para hacer frente a los *risks* identificados también serían incluidos como tareas en el *Sprint Backlog*.

Una vez que el *Sprint Backlog* está finalizado y el *Equipo Scrum* se comprometió al mismo, no deben añadirse nuevos *User Stories*; sin embargo, puede ser necesario añadir las tareas que pueden haberse pasado por alto o ignorado. Si surgen nuevas necesidades durante un *Sprint*, se añadirán al *Prioritized Product Backlog* y se incluirán en un futuro *Sprint*.

9.5.3.2 Sprint Burndown Chart*

9

Sprint Burndown Chart es un gráfico que muestra la cantidad de trabajo que queda en el *Sprint* actual. El *Sprint Burndown Chart* inicial es acompañado por una *burndown* planeado. El *Sprint Burndown Chart* debe actualizarse al final de cada día al completar el trabajo. Este gráfico muestra el progreso que se ha hecho por el *Equipo Scrum* y también permite la detección de cálculos que pudieron haber sido incorrectos. Si el *Sprint Burndown Chart* muestra que el *Equipo Scrum* no está en camino de terminar las tareas en el *Sprint* a tiempo, el *Scrum Master* debe identificar los obstáculos o *impediments* a la finalización con éxito y tratar de eliminarlos.

Un gráfico relacionado es un *Sprint Burnup Chart*. A diferencia del *Sprint Burndown Chart*, que muestra la cantidad de trabajo restante, el *Sprint Burnup Chart* muestra el trabajo realizado como parte del *Sprint*.

9.6 Fase Diagrama de flujo de datos

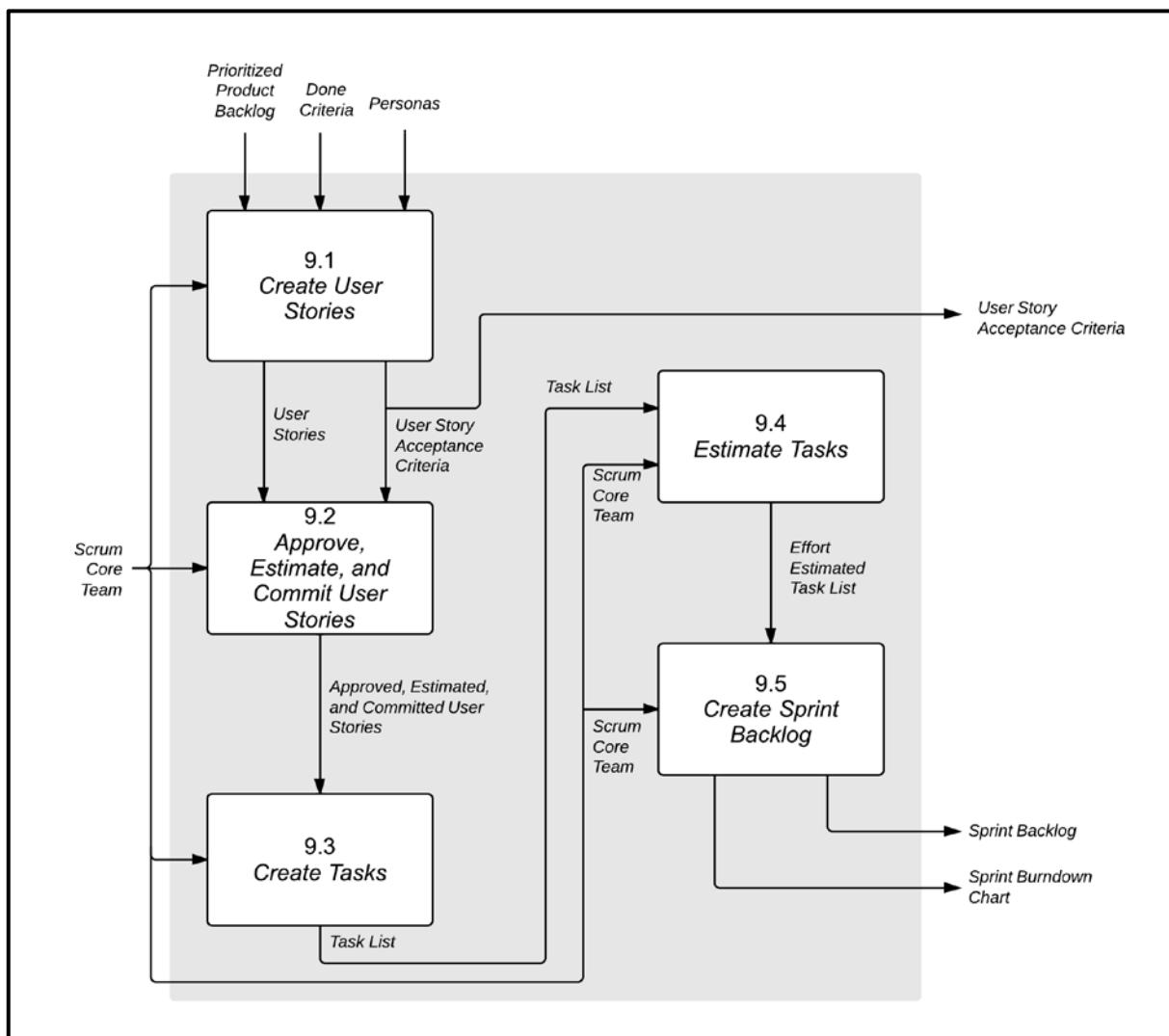


Figura 9-14: *Plan and Estimate* Fase—Diagrama de Flujo de Datos

10. IMPLEMENTAR

La fase de *Implement*, se relaciona con la ejecución de las tareas y actividades para crear el *product* de un proyecto. Estas actividades incluyen la creación de varias entregas, la realización de *Daily Standup Meetings*, y el mantenimiento (es decir, revisiones, ajustes, y actualización periódica) del *Product Backlog* en intervalos regulares.

Implement, tal como se define en *Una guía para el conocimiento de Scrum (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolios, programs y/o projects de cualquier sector*
- *Products, servicios o cualquier otro resultado que se les entregará a los socios*
- *Projects de cualquier tamaño y complejidad*

El término "*product*" en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria - desde pequeños proyectos o equipos con tan sólo seis miembros por equipo, hasta proyectos grandes y complejos que cuentan con cientos de miembros por equipo.

A fin de facilitar la mejor aplicación del marco de Scrum, en este capítulo se identifican las entradas, herramientas y salidas de cada proceso, ya sea como "obligatoria" o "facultativa". Las entradas, herramientas y salidas indicadas por asteriscos (*) son obligatorias, mientras que las que no tienen asteriscos son opcionales.

Se recomienda que el *Equipo Scrum* y aquellas personas que recién se familiarizan con el marco y los procesos de Scrum se centren principalmente en las aportaciones obligatorias, las herramientas y los productos; mientras que los *Propietario del producto*, *Scrum Masters* y otros profesionales con más experiencia de Scrum se deben esforzar por alcanzar un conocimiento más profundo de la información de todo este capítulo. También es importante darse cuenta de que, aunque todos los procesos se definen de forma única en la *Guía SBOK™*, no necesariamente se llevan a cabo de forma secuencial o por separado. A veces, puede ser más apropiado combinar algunos procesos, dependiendo de los requisitos específicos de cada proyecto.

Este capítulo está escrito desde la perspectiva de un *Equipo Scrum* quien trabaja en un *Sprint* para producir entregables (*Deliverables*) como parte de un proyecto más grande. Sin embargo, la información que se describe es igualmente aplicable a proyectos completos, programas y portfolios. Información adicional relacionada con el uso de Scrum para proyectos, programas y *portfolios* está disponible en los capítulos 2 al 7, que cubren los principios y aspectos de Scrum.

Figura 10-1 proporciona una visión general de los procesos en la fase de *Implement*, que son los siguientes:

10.1 Crear entregables—En este proceso, el *Equipo Scrum* trabaja en las tareas del *Sprint Backlog* para crear *Sprint Deliverables*. A menudo se utiliza un *Scrumboard* para realizar el seguimiento del trabajo y actividades que se llevan a cabo. Los *issues* o problemas que enfrenta el *Equipo Scrum* podrían actualizarse en un *Impediment Log*.

10.2 Llevar a cabo el Standup diario—En este proceso, todos los días se lleva a cabo una reunión *Timeboxed* altamente concentrada llamada *Daily Standup Meeting*. Este es el foro donde los miembros del *Equipo Scrum* comparten sus progresos y los obstáculos que puedan enfrentar.

10.3 Mantenimiento de la lista priorizada de pendientes del producto—En este proceso, el *Prioritized Product Backlog* se actualiza y mantiene continuamente. Un *Prioritized Product Backlog Review Meeting* se puede llevar a cabo, en el cual cambios o actualizaciones al *backlog* se discuten y se incorporan al *Prioritized Product Backlog* de forma debida.

10.1 Crear Entregables	10.2 Realizar el Standup Diario	10.3 Mantener la Lista Priorizada de Pendientes del Producto
<p>INPUTS</p> <ol style="list-style-type: none"> 1. Principal Equipo Scrum* 2. Lista de Pendientes del Sprint* 3. Tablero del Scrum* 4. Registro de Impedimentos* 5. Cronograma de Planificación del Lanzamiento 6. Dependencias 7. Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> 1. Experiencia del equipo* 2. Software 3. Otras herramientas de desarrollo 4. Experiencia del Cuerpo de Asesoramiento de Scrum <p>OUTPUTS</p> <ol style="list-style-type: none"> 1. Entregables del Sprint* 2. Tablero Scrum actualizado* 3. Registro de Impedimentos actualizado* 4. Solicitud de cambios no aprobados 5. Riesgos identificados 6. Riesgos mitigados 7. Dependencias actualizadas 	<p>INPUTS</p> <ol style="list-style-type: none"> 1. Equipo Scrum* 2. Scrum Master* 3. Tabla del Trabajo Consumido del Sprint* 4. Registro de Impedimentos* 5. Propietario del Producto 6. Experiencia del día anterior de trabajo 7. Tablero del Scrum 8. Dependencias <p>TOOLS</p> <ol style="list-style-type: none"> 1. Reunión del Standup Diario* 2. Tres preguntas diarias* 3. Sala de guerra 4. Videoconferencia <p>OUTPUTS</p> <ol style="list-style-type: none"> 2. Tabla del Trabajo Consumido del Sprint* 3. Registro de Impedimentos actualizado* 4. Equipo Scrum motivado 5. Tablero Scrum actualizado 6. Solicitud de cambios no aprobados 7. Riesgos identificados 8. Riesgos mitigados 9. Dependencias actualizadas 	<p>INPUTS</p> <ol style="list-style-type: none"> 1. Principal Equipo Scrum* 2. Lista Priorizada de Pendientes del Producto* 3. Entregables rechazados 4. Solicitudes de cambios aprobados 5. Solicitud de cambios rechazados 6. Riesgos identificados 7. Lista de Pendientes del Producto del Programa actualizada 8. Registro de la Retrospectiva del Sprint 9. Dependencias 10. Cronograma de Planificación del Lanzamiento 11. Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> 1. Reunión de Repaso de Priorización de la Lista del Producto* 2. Técnicas de comunicación 3. Otras técnicas de mantenimiento de la Lista Priorizada de Pendientes del Producto <p>OUTPUTS</p> <ol style="list-style-type: none"> 1. Lista Priorizada de Pendientes del Producto actualizada* 2. Cronograma de Planificación del Lanzamiento actualizado

Figura 10-1: Implement Resumen

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida "obligatoria" para el proceso correspondiente.

La figura 10-2 muestra las entradas, herramientas y salidas obligatorias para los procesos en la fase *Implement*.



Figura 10-2: Repaso de *Implement* (Esenciales)

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

10.1 Crear entregables

La figura 10-3 muestra todas las entradas, las herramientas y salidas para el proceso *Crear entregables*.

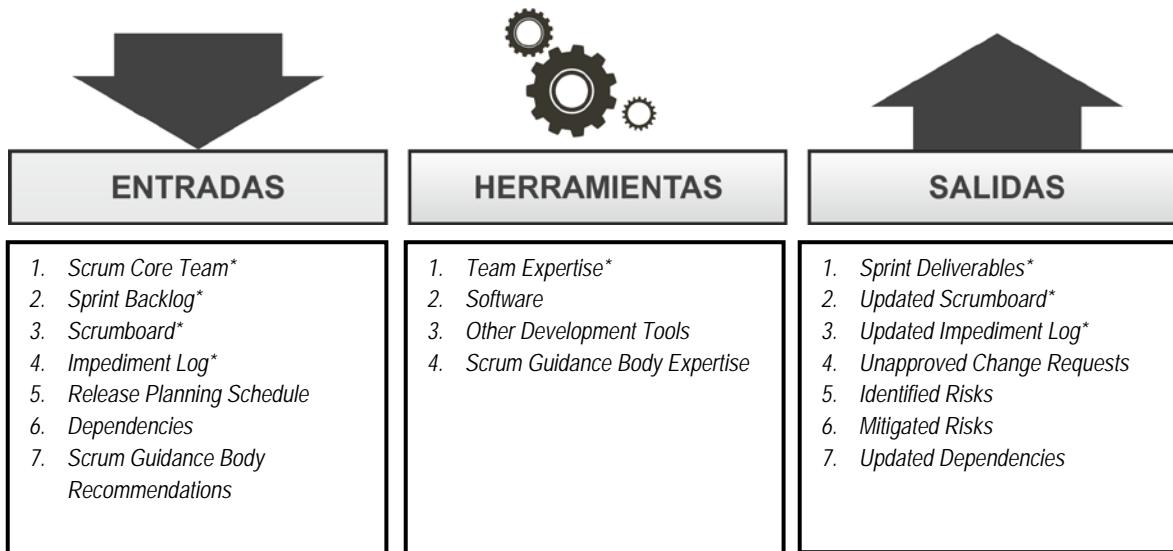
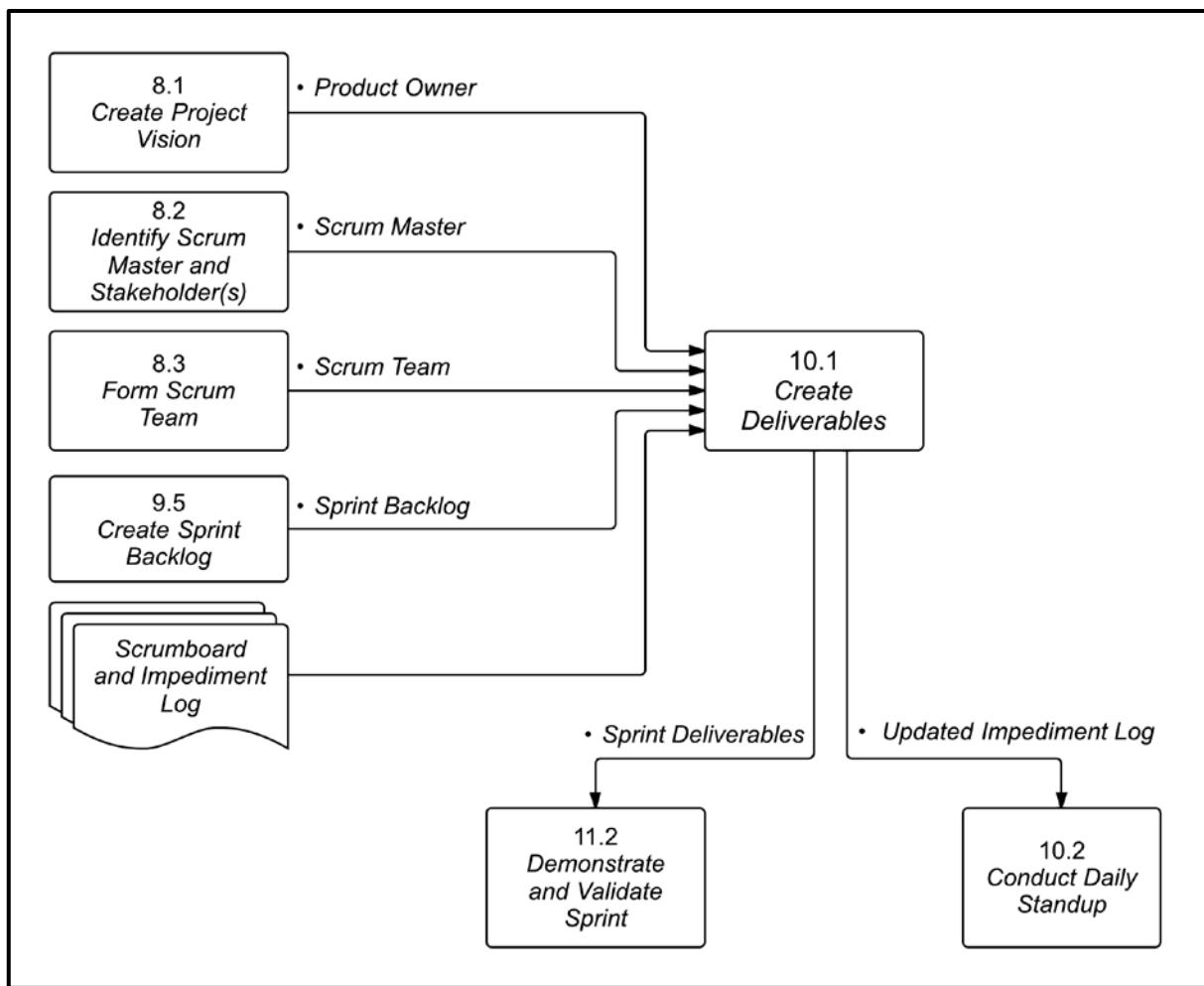


Figura 10-3: Crear entregables—Entradas, Herramientas y Salidas

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida "obligatoria" para el proceso correspondiente.

Figura 10-4: *Crear entregables*—Diagrama de Flujo de Datos

10.1.1 Entradas

10.1.1.1 *Scrum Core Team**

Descrito en la sección 8.4.1.1.

10.1.1.2 *Sprint Backlog**

Descrito en la sección 9.5.3.1.

10.1.1.3 *Scrumboard**

La transparencia de Scrum proviene de las herramientas de información abiertamente visibles como el *Scrumboard*, que muestra el progreso del equipo. El equipo utiliza un *Scrumboard* para planificar y realizar un seguimiento de los progresos realizados durante cada *Sprint*. El *Scrumboard* contiene cuatro columnas para indicar el progreso de las tareas estimadas para el *Sprint*: una columna "*To Do*" (Para Hacer), para las tareas aún no iniciadas, "*In Progress*" (En curso) para las tareas iniciadas pero que aún no se han completado, una columna llamada "*Testing*" (A Prueba) para las tareas completadas pero en el proceso donde se hacen las pruebas necesarias, y la columna "*Done*" (Hecho) para las tareas que se han completado con éxito y han sido aprobadas. Al comienzo de un *Sprint*, todas las tareas para ese *Sprint* se colocan en la columna de "*To Do*" y, posteriormente, se mueven hacia adelante en función de su progreso.

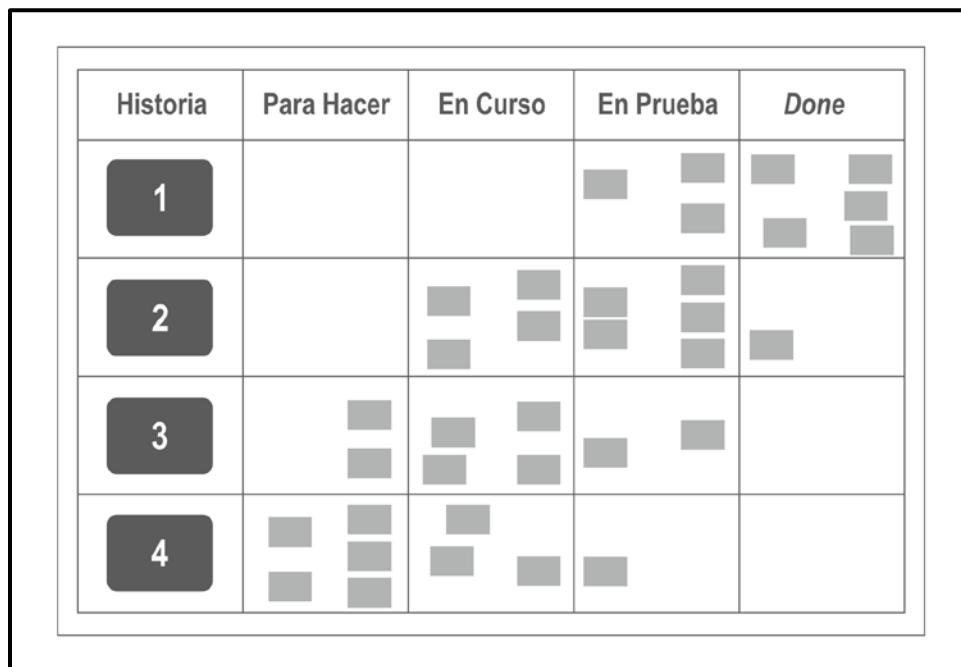


Figura 10-5: Scrumboard

El *Scrumboard* debe mantenerse, preferentemente, manualmente en papel o en una pizarra blanca, pero también se puede mantener electrónicamente en una hoja de cálculo.

El *Equipo Scrum* debe cambiar o agregarle información al *Scrumboard* según sea necesario para que el *Scrumboard* proporcione información visual y control sobre el trabajo que se realiza conforme a lo acordado y comprometido por el equipo.

10.1.1.4 *Impediment Log** (Registro de impedimento)

Impediment es cualquier obstáculo o barrera que reduce la productividad del *Equipo Scrum*. Los *impediments* deben ser identificados, resueltos y eliminados si el equipo ha de seguir trabajando con eficacia. Los *impediments* pueden ser internos del equipo, tales como flujo de trabajo ineficiente o falta de comunicación, o pueden ser externos. Algunos ejemplos de impedimentos externos pueden incluir problemas de licencia de software o los requisitos de documentación innecesarios. El marco de Scrum, con su transparencia inherente, facilita la rápida y fácil identificación de los obstáculos. El no identificar o hacerle frente a los obstáculos puede ser muy costoso. Los impedimentos deben ser registrados formalmente por el *Scrum Master* en un *impediment log*, y pueden ser discutidos durante los *Daily Standup Meetings* y *Sprint Review Meetings*.

10.1.1.5 *Release Planning Schedule*

Descrito en la sección 8.6.3.1.

10.1.1.6 *Dependencies* (Dependencias)

Descrito en la sección 9.3.3.3.

10.1.1.7 *Cuerpo de asesoramiento de Scrum Recommendations* (Recomendaciones del Cuerpo de asesoramiento de Scrum)

Descrito en la sección 8.1.1.12.

En el proceso de *Crear entregables*, *Cuerpo de asesoramiento de Scrum Recommendations* pueden incluir las mejores prácticas para crear los entregables, incluyendo los métodos preferidos para llevar a cabo revisiones, pruebas, documentaciones, etc.

10.1.2 Herramientas

10.1.2.1 *Team Expertise**

Esto se refiere a la experiencia colectiva de los miembros del *Equipo Scrum* para entender los *User Stories* y tareas en el *Sprint Backlog* con el fin de crear los entregables finales. La experiencia del equipo se utiliza para evaluar las entradas necesarias para ejecutar el trabajo previsto del proyecto. El juicio y la experiencia de cada miembro se aplica a todos los aspectos técnicos y de gestión del proyecto durante el proceso de *Crear entregables*. Los miembros del *Equipo Scrum* tienen la autoridad y la responsabilidad de determinar los mejores medios para convertir los elementos *Prioritized Product Backlog* en productos terminados, sin necesidad de participación de todos los *socios* del equipo. Experiencia adicional está disponible por parte del *Cuerpo de asesoramiento de Scrum*, según sea necesario.

10.1.2.2 Software

Las Herramientas de software automatizadas pueden ser utilizadas para la programación, la recopilación de información y la distribución. Las herramientas de colaboración virtuales también son esenciales en *projects* en los que el *Equipo Scrum* no está *colocated*. Una variedad de herramientas basadas en software automatizadas está disponible permitiendo el seguimiento del progreso, la recopilación y distribución de datos, y contribuyendo a la aceleración de los procesos.

10.1.2.3 Otras herramientas de desarrollo

Basado en las especificaciones del proyecto y de la industria, otras herramientas de desarrollo se pueden utilizar.

1. *Refactoring*

Refactorizing es una herramienta específica para los proyectos de software. El objetivo de esta técnica es mejorar el mantenimiento del código existente y hacerlo más simple, más conciso y más flexible. *Refactoring* significa mejorar el diseño del código actual, sin cambiar el comportamiento del código. Se trata de lo siguiente:

- La eliminación de código repetitivo y redundante
- Reducir los métodos y las funciones en rutinas más pequeñas
- Definir las variables con claridad y los nombres de los métodos
- Simplificar el diseño del código
- Hacer que el código sea más fácil de entender y modificar

La refactorización regular optimiza el diseño de código de poco a poco, durante un período de tiempo. En última instancia, la refactorización resulta en códigos más fáciles de mantener, mientras se preservan todas las funcionalidades.

2. *Design Patterns*

Design Patterns proporcionan una manera formal de registrar una resolución de un problema de diseño en un campo específico de especialización. Estos patrones registran tanto el proceso que se utiliza, como la misma resolución, que puede ser reusada para mejorar la toma de decisiones y la productividad.

10.1.2.4 *Cuerpo de asesoramiento de Scrum Expertise*

Descrito en la sección 8.4.2.7.

En los procesos *Crear entregables*, *Approve Estimate* y *Commit User Stories*, El *Cuerpo de asesoramiento de Scrum Expertise* podría relacionarse con las normas y reglamentos documentados, directrices de desarrollo; o los estándares y mejores prácticas (por ejemplo, la orientación sobre la forma de llevar a cabo revisiones o pruebas). También puede haber un equipo de expertos en la materia que le pueda servir de orientación al *Equipo Scrum* en la creación de entregables (*deliverables*). Este equipo podría incluir arquitectos, desarrolladores de alto nivel, expertos en seguridad, u otras personas con experiencia.

10.1.3 Salidas

10.1.3.1 *Sprint Deliverables**

Al final de cada *Sprint*, se completa un mínimo de producto o entregable. La entrega debe poseer todas las características y funcionalidades definidas en los *User Stories* incluidas en el *Sprint*, las cuales deben ser probadas con éxito.

10.1.3.2 *Updated Scrumbard**

El *Scrumbard* se actualiza con regularidad a medida que el equipo produce más trabajo. Sin embargo, al final del *Sprint*, el *Scrumbard* se restablecerá o borrará y un nuevo *Scrumbard* se creará para el próximo *Sprint*.

10.1.3.3 Updated Impediment Log (Registro de impedimento actualizado)

Descrito en la sección 10.1.1.4.

10.1.3.4 Unapproved Change Requests

Descrito en la sección 8.4.1.6.

10.1.3.5 Risks identificados

Descrito en la sección 8.4.3.4.

10.1.3.6 Risks mitigados

A medida que el *Equipo Scrum* ejecuta el trabajo de los entregables de acuerdo a los *User Story* en el *Product Backlog*, lleva a cabo las acciones de mitigación que se han definido para abordar cualquier *identified risk*. A lo largo del *proceso* de *Crear entregables*, el equipo documenta los nuevos riesgos identificados y las medidas de mitigación adoptadas. El registro de los riesgos del proyecto es un documento vivo, actualizado de forma continua durante todo el *project* por el equipo para refleja el estado actual de todos los riesgos.

Información adicional sobre *Managing Risks* se describe en la sección 7.4.3

10.1.3.7 Updated Dependencies (Dependencias actualizadas)

Descrito en la sección 9.3.3.3.

10.2 Llevar a cabo el Standup diario

La figura 10-6 muestra todas las entradas, las herramientas y las salidas del proceso *Llevar a cabo el Standup diario*.

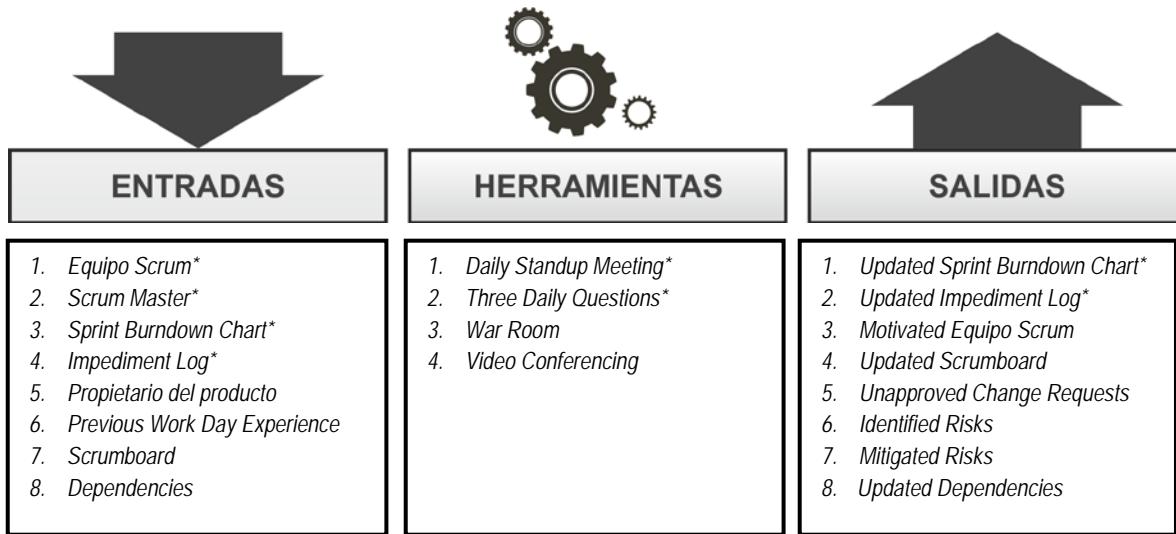


Figura 10-6: *Llevar a cabo el Standup diario*—Entradas, Herramientas y Salidas

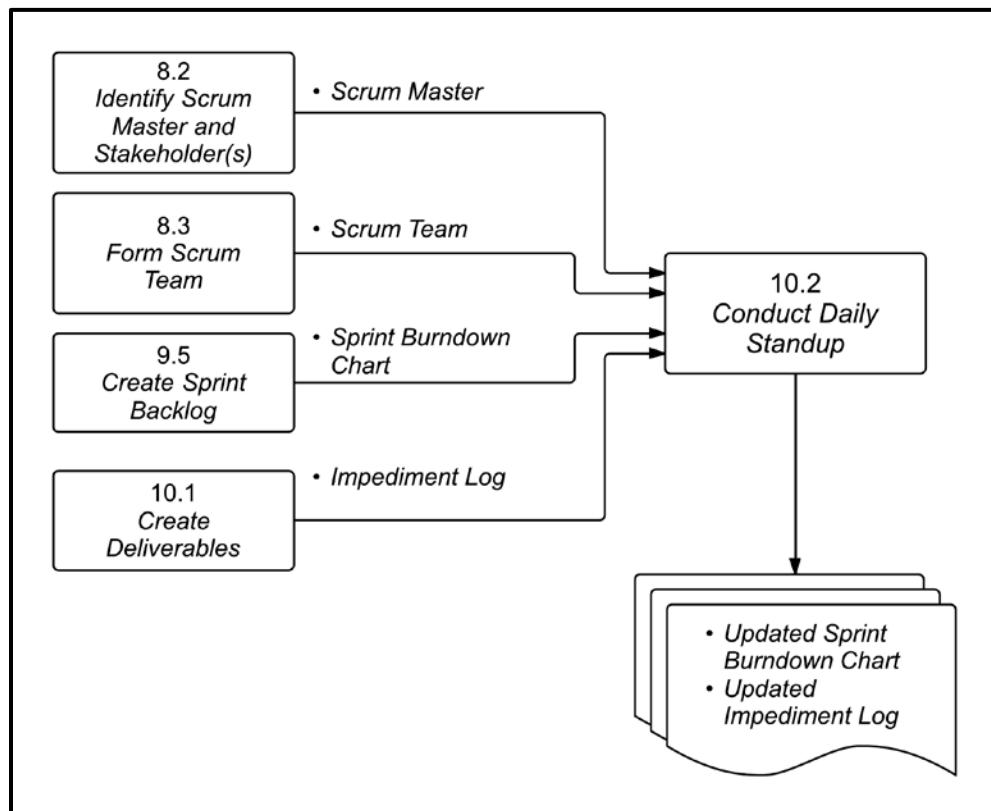


Figura 10-7: *Llevar a cabo el Standup diario*—Diagrama de Flujo de Datos

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

10.2.1 Entradas

10.2.1.1 *Equipo Scrum**

Descrito en la sección 8.3.3.1.

10.2.1.2 *Scrum Master**

Descrito en la sección 8.2.3.1.

10.2.1.3 *Sprint Burndown Chart**

Descrito en la sección 9.5.3.2.

10.2.1.4 *Impediment Log** (Registro de impedimento)

Descrito en la sección 10.1.1.4.

10.2.1.5 *Propietario del producto*

Descrito en la sección 8.1.3.1.

10.2.1.6 *Experiencia del día previo*

Los miembros del *Equipo Scrum* mantienen al tanto a sus compañeros de equipo en el *Daily Standup Meeting*. Esta sesión se denomina *Standup* porque los miembros están de pie durante toda la reunión. Los miembros del equipo discuten logros y la experiencia de trabajo del día anterior. Esta experiencia es una entrada importante para el *Daily Standup Meeting*.

10.2.1.7 *Scrumboard*

Descrito en la sección 10.1.1.3.

10.2.1.8 *Dependencies* (Dependencias)

Descrito en la sección 9.3.3.3.

10.2.2 Herramientas

10.2.2.1 *Daily Standup Meeting**

Daily Standup Meeting es una reunión diaria de corta duración, *Time-boxed* a 15 minutos. Los miembros del equipo se reúnen para informar de sus progresos en el *Sprint* y planificar las actividades del día. La duración de las reuniones es muy corta y se espera que todos los miembros del *Equipo Scrum* asistan. La reunión no se cancela o se retrasa si uno o más miembros no pueden asistir.

En la reunión, cada miembro del *Equipo Scrum* proporciona respuestas a las tres preguntas diarias que se mencionan en la sección 10.2.2.2. Se recomiendan los debates entre el *Scrum Master* y el equipo o entre algunos miembros del *Equipo Scrum*, pero estas discusiones suceden después del *Daily Standup* para asegurar que la reunión sea corta.

10.2.2.2 *Three Daily Questions** (Tres preguntas diarias)

En *Daily Standup Meeting*, facilitado por el *Scrum Master*, cada miembro del *Equipo Scrum* proporciona información en forma de respuestas a tres preguntas específicas:

- ¿Qué terminé ayer?
- ¿Qué voy a terminar hoy?
- ¿Qué impedimentos u obstáculos (si los hay) estoy enfrentando en la actualidad?

Al centrarse en estas tres preguntas, todo el equipo puede tener una comprensión clara de la situación laboral. En ocasiones, otros elementos pueden ser discutidos, pero esto se reduce al mínimo para respetar el aspecto *time-boxed* de la reunión.

Es muy recomendable que las dos primeras preguntas sean respondidas por los miembros del equipo de manera cuantificable, si es posible, en lugar de respuestas largas y cualitativas. Los miembros del equipo pueden organizar reuniones adicionales después del *Daily Standup Meeting* para hacer frente a los artículos que necesitan un análisis adicional.

10.2.2.3 *War Room*

En Scrum, es preferible que el equipo esté *colocated*, en otras palabras, que todos los miembros del equipo trabajen en el mismo lugar. El término comúnmente utilizado para describir este lugar es *War Room*. Normalmente, está diseñado de tal manera que los miembros del equipo pueden moverse libremente, trabajar y comunicarse fácilmente ya que se encuentran cerca el uno del otro. Típicamente, hay tarjetas índices, fichas, notas adhesivas, y otras herramientas de baja tecnología, y de alto contacto, que se encuentran disponibles en el *War Room* para facilitar el flujo de trabajo, la colaboración y la resolución de problemas.

La sala es a veces ruidosa debido a las conversaciones del equipo, pero estas conversaciones contribuyen al progreso del equipo. Un buen *War Room* no tiene cubículos y permite que todo el equipo se siente junto para asegurar la comunicación cara a cara, lo que conduce a la formación de equipos y la apertura. *War Room* es ideal para la realización de *Daily Standup Meetings* también.

El/Los *Stakeholder(s)* y miembros de otros equipos de Scrum también podrían caminar por el *War Room* y discutir temas relevantes.

10.2.2.4 Videoconferencia

En las situaciones de la vida real, no siempre es posible para todo el *Equipos Scrum* estar *colocated*. En estos casos, se hace imperativo el uso de herramientas de videoconferencia para permitir la comunicación cara a cara.

10.2.3 Salidas

10.2.3.1 *Updated Sprint Burndown Chart**

Descrito en la sección 9.5.3.2.

10.2.3.2 *Updated Impediment Log**

Descrito en la sección 10.1.1.4.

10.2.3.3 Equipo Scrum motivado

Daily Standup Meetings propagan la idea de que cada miembro del equipo es valioso y es un importante contribuyente, lo que mejora la moral individual del equipo. Esto, junto con el concepto de equipos de auto-organización, mejora la motivación general, conduce a un mejor rendimiento del equipo y mejora la calidad de los resultados producidos.

Equipo Scrum se describe en la sección 8.3.3.1.

10.2.3.4 Updated Scrumboard

Descrito en la sección 10.1.1.3.

10.2.3.5 Unapproved Change Requests

Descrito en la sección 8.4.1.6.

10.2.3.6 Risks identificados

Descrito en la sección 8.4.3.4.

10.2.3.7 Risks mitigados

Descrito en la sección 10.1.3.6.

10.2.3.8 Dependencies actualizadas

Descrito en la sección 9.3.3.3.

10.3 Mantenimiento de la lista priorizada de pendientes del producto

La figura 10-8 muestra todas las entradas, las herramientas y las salidas para el proceso de *Mantenimiento de la lista priorizada de pendientes del producto*.

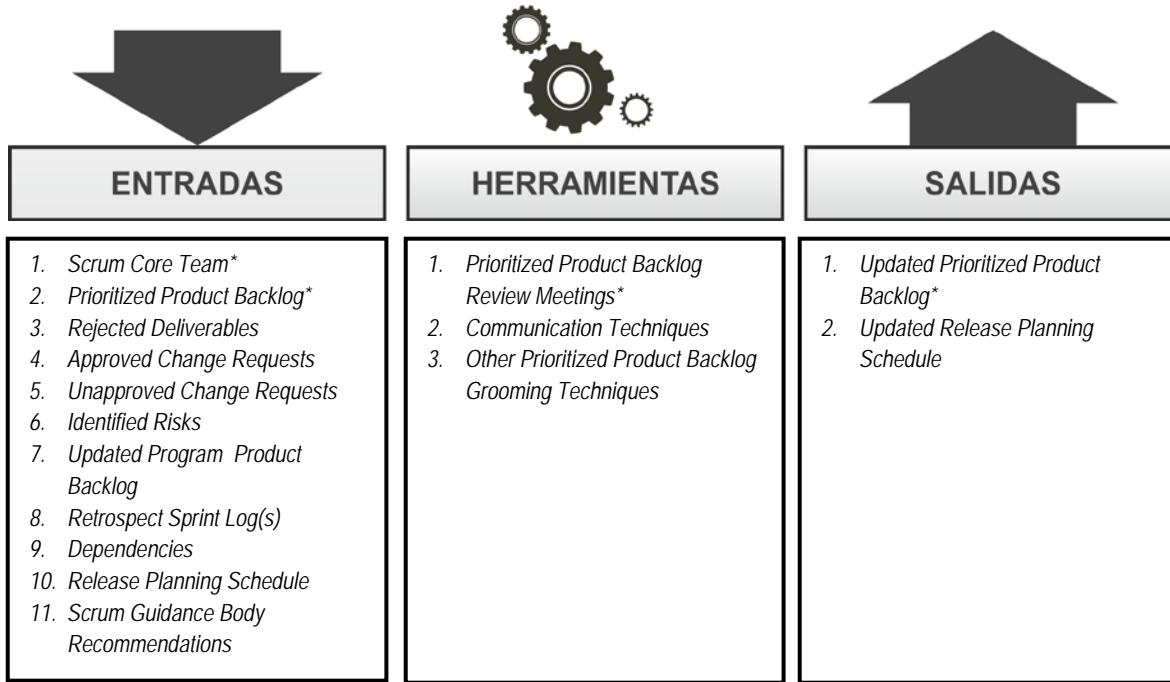


Figura 10-8: *Mantenimiento de la lista priorizada de pendientes del producto—Entradas, Herramientas y Salidas*

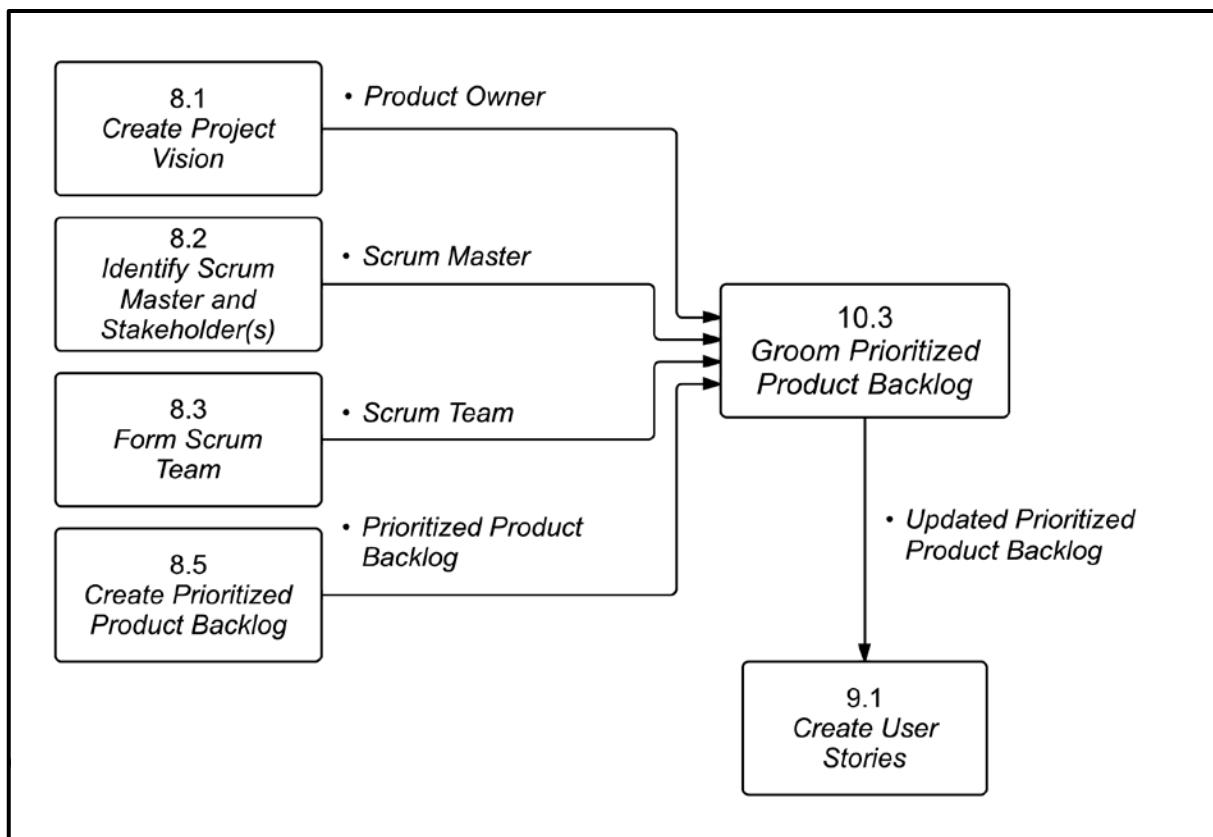


Figura 10-9: Mantenimiento de la lista priorizada de pendientes del producto—Diagrama de Flujo de Datos

10.3.1 Entradas

10.3.1.1 *Scrum Core Team** (Equipo Principal de Scrum)

Descritas en las secciones 8.1.3.1, 8.2.3.1, and 8.3.3.1.

10.3.1.2 *Prioritized Product Backlog**

Descrito en la sección 8.5.3.1.

10.3.1.3 *Rejected Deliverables*

En los casos en que una entrega no cumpla con los criterios de aceptación, esto se considera como *rejected deliverables*. Los *rejected deliverables* normalmente no se mantienen en una lista separada. Simplemente permanecen en el *Prioritized Product Backlog* y no son marcados como *Done* para que puedan ser re-priorizados en el proceso de *Mantenimiento de la lista priorizada de pendientes del producto* y sean considerados para el desarrollo en el próximo *Sprint*.

10.3.1.4 *Approved Change Requests*

Descrito en la sección 8.4.1.5.

10.3.1.5 *Unapproved Change Requests*

Descrito en la sección 8.4.1.6.

10.3.1.6 *Identified Risks* (Riesgos identificados)

Descrito en la sección 8.4.3.4.

10.3.1.7 Updated Program Product Backlog

Al igual que el *Project Product Backlog*, el *Program Product Backlog* también puede someterse a la preparación periódica para incorporar cambios y nuevos requisitos. Los cambios al *Program Product Backlog* pueden ser el resultado de cambios en cualquiera de las condiciones externas o internas. Las condiciones externas pueden incluir el cambio de situaciones de negocio, tendencias de la tecnología, o requisitos de cumplimiento legal. Los factores internos que afectan al *Program Product Backlog* podrían estar relacionados con las modificaciones en la estrategia o las políticas de la organización, riesgos identificados y de otros factores. Los cambios en los requerimientos en el *Program Product Backlog* a menudo impactan el *Project Product Backlogs* de los proyectos subyacentes, por lo que deben tenerse en cuenta durante el proceso de *Mantenimiento de la lista priorizada de pendientes del producto*.

10.3.1.8 Retrospectiva de Sprint Log(s)

Descrito en la sección 11.3.3.4.

10.3.1.9 Dependencias

Descrito en la sección 9.3.3.3.

10.3.1.10 Release Planning Schedule

Descrito en la sección 8.6.3.1.

10.3.1.11 Cuerpo de asesoramiento de Scrum Recommendations

Descrito en la sección 8.1.1.12.

En el proceso de *Mantenimiento de la lista priorizada de pendientes del producto*, *Cuerpo de asesoramiento de Scrum Recommendations* pueden incluir las mejores prácticas sobre la manera de entender sistemáticamente y cotejar los requisitos de los *stakeholder(s)* y *Equipo Scrum* y luego priorizar adecuadamente la pendiente del *product* y comunicar cambios a todas las personas relevantes que intervienen en el *project Scrum*.

10.3.2 Herramientas

10.3.2.1 *Prioritized Product Backlog Review Meetings**

El *Propietario del producto* puede tener reuniones múltiples y separadas con los *socios*, el *Scrum Master* y el *Equipo Scrum* relevante para asegurar que él o ella tenga suficiente información para hacer cambios al *Prioritized Product Backlog* durante el proceso de *Mantenimiento de la lista priorizada de pendientes del producto*.

La intención del *Prioritized Product Backlog Review Meetings* es asegurar que los *User Stories* y los *acceptance criteria* se entiendan y se escriban correctamente por el *Product Owner* de modo que reflejen los requisitos de los *stakeholder* (Customers); que los *User Stories* sean entendidos por todos los miembros de *Equipo Scrum*; y que los *User Stories* de los usuarios de alta prioridad sean muy refinados para que el *Equipo Scrum* pueda estimar correctamente y comprometerse con este tipo de *User Stories*.

Los *Prioritized Product Backlog Review Meetings* también aseguran que los casos irrelevantes de *User Stories* se eliminen y que los *Approved Change Requests* o *Identified Risks* sean incorporados en el *Prioritized Product Backlog*.

10.3.2.2 Técnicas de comunicación

10

Scrum promueve la comunicación precisa y eficaz sobre todo a través de *colocation* del *Equipo Scrum*. Scrum también favorece informales, las interacciones cara a cara sobre las comunicaciones formales por escrito. Cuando un *Equipo Scrum* necesita ser distribuido, el *Scrum Master* debe garantizar que las técnicas de comunicación eficaces estén disponibles para que los equipos puedan auto-organizarse y trabajar con eficacia.

10.3.2.3 *Otras técnicas de mantenimiento de Prioritized Product Backlog*

Otras herramientas de *Prioritized Product Backlog Grooming* incluyen muchas herramientas que se utilizan para los siguientes procesos:

- *Desarrollo de épica(s)*—Descritos en la sección 8.4.2.
- *Creación de la lista priorizada de pendientes del producto*—Descritos en la sección 8.5.2.
- *Realizar el plan de lanzamiento*—Descritos en la sección 8.6.2.
- *Elaborar historias de usuario*—Descritos en la sección 9.1.2.
- *Aprobar, estimar y asignar historias de usuarios*—Descritos en la sección 9.2.2.
- *Elaboración de tareas*—Descritos en la sección 9.3.2.
- *Estimar tareas*—Descritos en la sección 9.4.2.

10.3.3 Salidas

10.3.3.1 *Prioritized Product Backlog* actualizado*

Descrito en la sección 8.5.3.1.

Prioritized Product Backlog puede ser actualizado con nuevos *User Stories*, nuevos *Change Requests*, nuevos *Identified Risks*, *User Stories* actualizados, o la nueva priorización de *User Stories* existentes.

10.3.3.2 *Release Planning Schedule* actualizado

Descrito en la sección 8.6.3.1.

El *Release Planning Schedule* puede ser actualizado para reflejar el impacto de los *User Stories* nuevos o modificados en el *Prioritized Product Backlog*.

10.4 Fase Diagrama de flujo de Datos

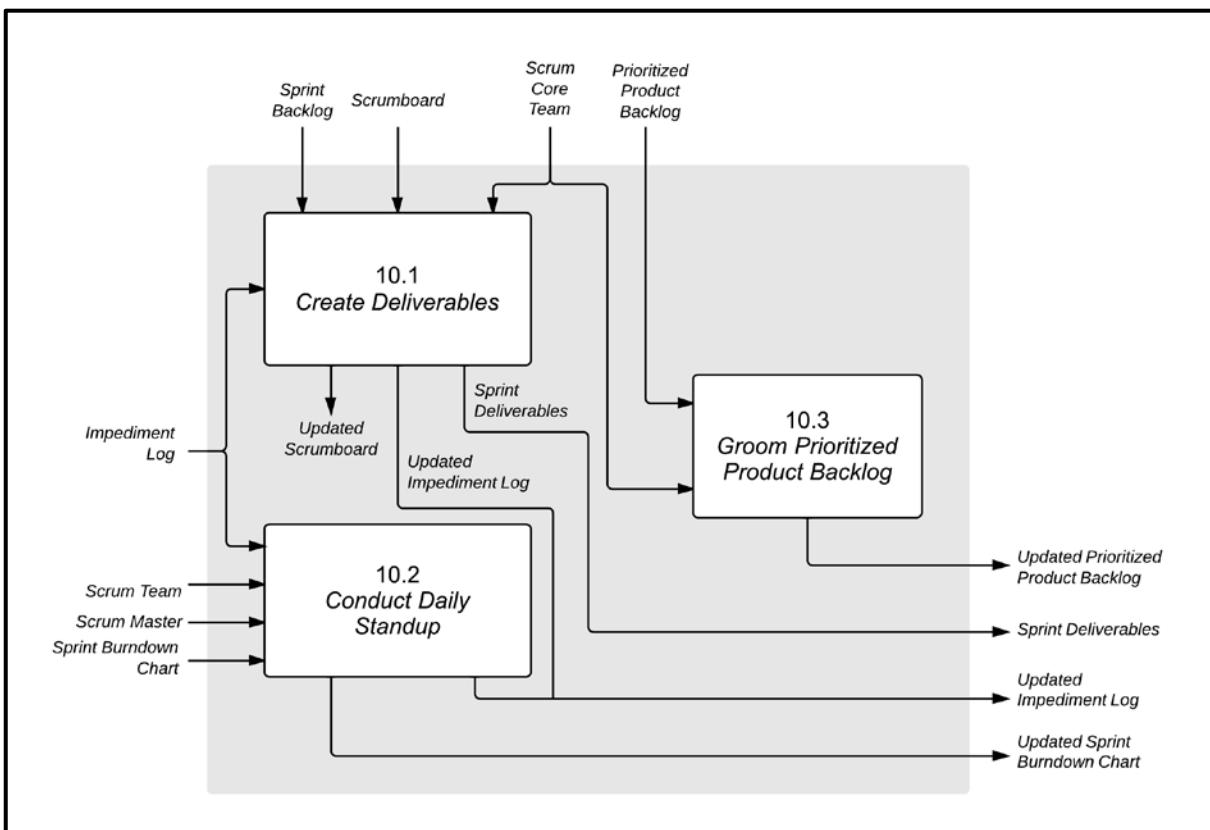


Figura 10-10: Implementar la Fase—Diagrama de Flujo de Datos

11. REVISIÓN Y RETROSPECTIVA

La fase llamada Revisión y Retrospectiva se ocupa de la revisión de los entregables y del trabajo que se ha hecho, y determina las mejores prácticas y métodos utilizados para hacer el trabajo relacionado al proyecto. En las grandes organizaciones, el proceso de Revisión y Retrospectiva puede incluir la convocatoria de *Scrum of Scrums Meetings*.

Revisión y Retrospectiva, tal como se define en *Una guía para el conocimiento de Scrum (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolios, programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se le entregará a los *socios*
- *Projects* de cualquier tamaño y complejidad

El término producto (*product*) en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria - desde proyectos pequeños o equipos con tan sólo seis miembros por equipo, hasta proyectos grandes y complejos que cuentan con cientos de miembros por equipo.

Con el fin de facilitar la mejor aplicación del marco de Scrum, este capítulo identifica las entradas, herramientas y salidas de cada proceso, ya sea como algo "obligatorio" u "opcional". Las entradas, herramientas y salidas indicadas por asteriscos (*) son obligatorias, mientras que las que no tienen asteriscos son opcionales.

Se recomienda que el *Equipo Scrum* y aquellas personas que se están recién familiarizando con el marco y los procesos Scrum se centren principalmente en las aportaciones obligatorias, las herramientas y los *products*; mientras que los *Propietario del producto*, *Scrum Masters*, y aquellos practicantes con más experiencia se esfuerzen por alcanzar un conocimiento más profundo de la información en todo este capítulo. También es importante darse cuenta de que, aunque todos los procesos se definen de forma única en el *Guía SBOK™*, no se lleva a cabo necesariamente de forma secuencial o por separado. A veces, puede ser más apropiado combinar algunos procesos, dependiendo de los requisitos específicos de cada proyecto.

Este capítulo está escrito desde la perspectiva de un *Equipo Scrum* quien trabaja en un *Sprint* para producir entregables como parte de un *project*. Sin embargo, la información que se describe es igualmente aplicable a proyectos completos, programas y *portfolios*. Información adicional relacionada con el uso de Scrum para *projects, programs y portoflios* está disponible en los capítulos 2 al 7, que cubren los principios y aspectos de Scrum.

La figura 11-1 proporciona una visión general de los procesos de la fase *Revisión y Retrospectiva*, los cuales son los siguientes:

11.1 Convocar Scrum de Scrums—En este proceso, los representantes del *Equipo Scrum* convocan una reunión de *Scrum of Scrums (SoS)* en intervalos predeterminados o cuando sea necesario para colaborar y realizar un seguimiento de sus respectivos progresos, *impediments*, y las dependencias entre los equipos. Esto es relevante sólo para grandes proyectos en los que múltiples *Equipos Scrum* están involucrados.

11.2 Demostración y validación del Sprint—En este proceso, el *Equipo Scrum* les demuestra el *Sprint Deliverable* al *Propietario del producto* y a los relevantes *socios* en un *Sprint Review Meeting*. El propósito de esta reunión es asegurar la aprobación y aceptación del *product* o servicio por parte del *Propietario del producto*.

11.3 Retrospectiva de Sprint—En este proceso, el *Scrum Master* y el *Equipo Scrum* se reúnen para discutir las lecciones aprendidas a lo largo del *Sprint*. Esta información se documenta como las lecciones aprendidas que pueden aplicarse a los siguientes *Sprints*. A menudo, como resultado de esta discusión, puede haber un *Agreed Actionable Improvements* o *Updated Cuerpo de asesoramiento de Scrum Recommendations*.

11.1 Convocar el Scrum de Scrums	11.2 Demostrar y Validar el Sprint	11.3 Retrospectiva del Sprint
<p>INPUTS</p> <ol style="list-style-type: none"> Scrum Master o representantes del Equipo* Jefe Scrum Master Jefe Propietario del Producto Agenda de la Reunión Registro de Impedimentos Dependencias Salidas de la Retrospectiva del Sprint <p>TOOLS</p> <ol style="list-style-type: none"> Reunión de Scrum de Scrums* Cuatro preguntas por Equipo* Videoconferencia Sala de Guerra Experiencia del Cuerpo de Asesoramiento de Scrum <p>OUTPUTS</p> <ol style="list-style-type: none"> Mejor coordinación del Equipo* Incidentes resueltos Registro de Impedimentos actualizado Dependencias actualizadas 	<p>INPUTS</p> <ol style="list-style-type: none"> Principal Equipo Scrum* Entregables del Sprint* Pendientes del Sprint* Criterio de Terminado* Criterio de Aceptación de la Historia del Usuario* Socio(s) Cronograma de Planificación del Lanzamiento Riesgos identificados Dependencias Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> Reunión de Revisión del Sprint* Ánalisis del Valor Ganado Experiencia del Cuerpo de Asesoramiento de Scrum <p>OUTPUTS</p> <ol style="list-style-type: none"> Entregables aceptados* Entregables rechazados Riesgos actualizados Resultados del Ánalisis del Valor Ganado Cronograma de Planificación del Lanzamiento actualizado Dependencias actualizadas 	<p>INPUTS</p> <ol style="list-style-type: none"> Scrum Master* Equipo Scrum* Salidas de Demostrar y Validar el Sprint* Propietario del Producto Recomendaciones del Cuerpo de Asesoramiento de Scrum <p>TOOLS</p> <ol style="list-style-type: none"> Reunión de la Retrospectiva del Sprint* ESVP Barco Parámetros y Técnicas de Medición Experiencia del Cuerpo de Asesoramiento de Scrum <p>OUTPUTS</p> <ol style="list-style-type: none"> Mejoras Acordadas Susceptibles a la Acción* Elementos de Acción Asignada y Fechas de Entrega Elementos no funcionales propuestos para la Lista Priorizada de Pendientes del Producto Registro de la Retrospectiva del Sprint Lecciones aprendidas del Equipo de Scrum Recomendaciones actualizadas del Cuerpo de Asesoramiento de Scrum

Figura 11-1: Resumen de Revisión y Retrospectiva

La figura 11-2 muestra las entradas, herramientas y salidas obligatorias para los procesos en la fase *Revisión y Retrospectiva*.

<i>11.1 Convocar Scrum de Scrums</i>	<i>11.2 Demostración y validación del Sprint</i>	<i>11.3 Retrospectiva del Sprint</i>
<p>ENTRADAS</p> <ul style="list-style-type: none"> 1. Scrum Master or Equipo Scrum Representatives* <p>HERRAMIENTAS</p> <ul style="list-style-type: none"> 1. Scrum of Scrums Meeting* 2. Four Questions per Team* <p>SALIDAS</p> <ul style="list-style-type: none"> 1. Better Team Coordination* 	<p>ENTRADAS</p> <ul style="list-style-type: none"> 1. Scrum Core Team* 2. Sprint Deliverables* 3. Sprint Backlog* 4. Done Criteria* 5. User Story Acceptance Criteria* <p>HERRAMIENTAS</p> <ul style="list-style-type: none"> 1. Sprint Review Meetings* <p>SALIDAS</p> <ul style="list-style-type: none"> 1. Accepted Deliverables* 	<p>ENTRADAS</p> <ul style="list-style-type: none"> 1. Scrum Master* 2. Equipo Scrum* 3. Outputs from Demostración y validación del Sprint * <p>HERRAMIENTAS</p> <ul style="list-style-type: none"> 1. Retrospect Sprint Meeting* <p>SALIDAS</p> <ul style="list-style-type: none"> 1. Agreed Actionable Improvements*

Figura 11-2: Resumen de Revisión y Retrospectiva (Esenciales)

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

11.1 Convocar Scrum de Scrums

La figura 11-3 muestra todas las entradas, las herramientas y las salidas para el proceso *Convocar Scrum de Scrums*.

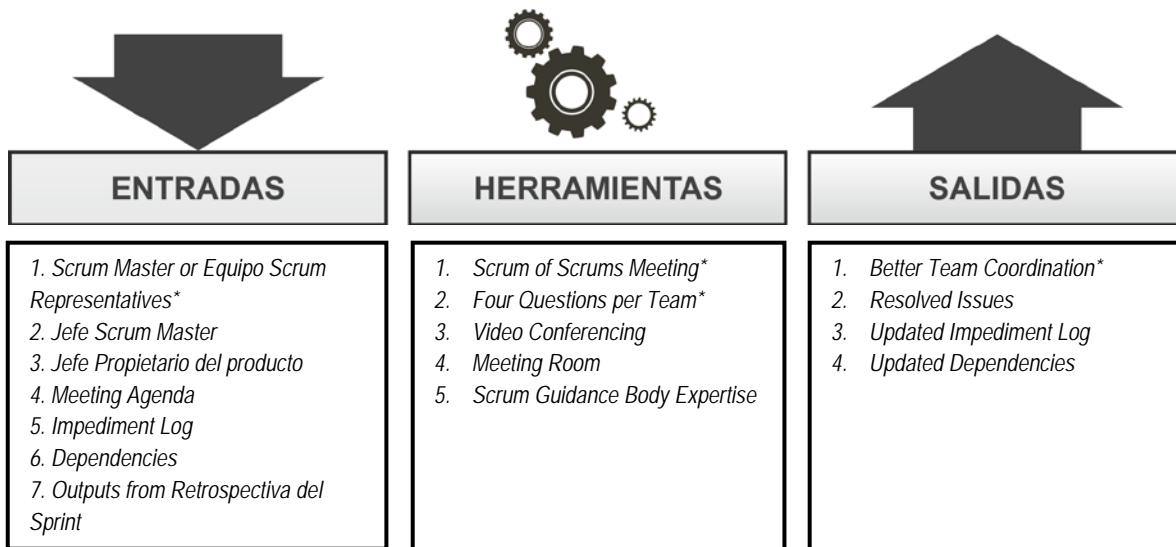


Figura 11-3: Convocar Scrum de Scrums—Entradas, Herramientas y Salidas

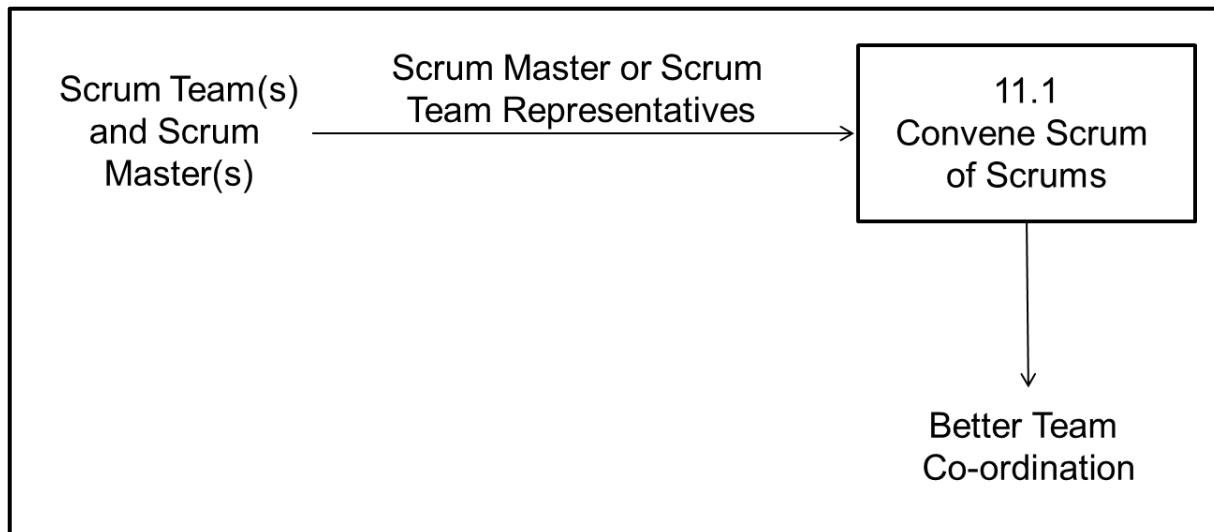


Figura 11-4: Convocar Scrum de Scrums—Diagrama de Flujo de Datos

Nota: Los asteriscos () denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.*

11.1.1 Entradas

11.1.1.1 *Scrum Master o Equipo Scrum Representatives**

Normalmente, un miembro de cada *Equipo Scrum* representará a su equipo en el *Scrum of Scrums (SoS) Meeting*. En la mayoría de los casos, este es el *Scrum Master*, pero a veces alguien más puede representar al equipo. Una sola persona puede ser nominada por el equipo para que los represente en todas las reuniones SoS, o el representante puede cambiar con el tiempo, dependiendo si hay otra persona que pueda mejor cumplir esta función según los *issues* y circunstancias. La persona que participe en la reunión debe tener el conocimiento técnico para identificar los casos en los que los equipos podrían causar *impediments* o retrasos.

11.1.1.2 *Jefe Scrum Master*

Descrito en la sección 8.2.1.6.

11.1.1.3 *Jefe Propietario del producto(s)*

Descrito en la sección 8.1.1.5

11.1.1.4 *Meeting Agenda (Agenda de la reunión)*

El propósito principal del *Scrum of Scrums (SoS) Meeting* es comunicar el progreso entre múltiples equipos. El *Jefe Scrum Master* (o cualquier *Scrum Master* que facilite la reunión SoS) puede anunciar una agenda antes de la reunión. Esto le permite a cada equipo considerar los temas del programa en preparación para la reunión SoS. Cualquier *impediment* que enfrente un equipo que pueda afectar a otros equipos, se debe indicar para ser tratado durante la reunión SoS. Además, si un equipo se da cuenta de un problema de gran escala, un cambio o riesgo que pueda afectar a otros equipos, esto debe ser comunicado en la reunión SoS.

11.1.1.5 *Impediment Log (Registro de impedimentos)*

Descrito en la sección 10.1.1.4.

11.1.1.6 *Dependencies (Dependencias)*

Descrito en la sección 9.3.3.3.

11.1.1.7 Las salidas de *Retrospectiva de Sprint*

Las salidas del proceso *Retrospectiva de Sprint* pueden tener *issues* que podrían afectar múltiples *Equipos Scrum*, y se podrían utilizar como entradas para un *Scrum of Scrums (SoS) Meeting* más efectivo.

11.1.2 Herramientas

11.1.2.1 *Scrum of Scrums Meeting**

Estas son reuniones preferentemente cortas (pero generalmente no *time-boxed* para permitir un mayor intercambio de información entre los equipos), donde los representantes de los *Equipos Scrum* se reúnen para compartir el estado de los equipos respectivos. El *Scrum of Scrums (SoS) Meeting* se llevará a cabo en intervalos predeterminados o cuando sea requerido por los *Equipos Scrum* para facilitar el intercambio de información entre los diferentes *Equipos Scrum*. *Issues*, dependencias y *risks* que afectan a múltiples *Equipos Scrum* pueden ser observados cuidadosamente, lo cual ayudará a los distintos equipos que trabajan en un proyecto grande a coordinar e integrar sus trabajos de mejor forma. Es la responsabilidad del Jefe *Scrum Master* (u otro *Scrum Master* que facilita las reuniones SOS) asegurarse de que todos los representantes tengan un ambiente propicio para compartir sinceramente de la información, incluso ofrecer observaciones a otros representantes. Para proyectos más grandes, con la participación de numerosos equipos, múltiples niveles de estas reuniones pueden ser convocados para así compartir el estado de los equipos respectivos.

La reunión SoS se describe con más detalle en la sección 3.7.2.

11.1.2.2 Cuatro preguntas por equipo*

Cada representante del *Equipo Scrum* proporcionará actualizaciones de su equipo. Estas actualizaciones se proporcionan generalmente en forma de respuestas a cuatro preguntas específicas.

- 1) ¿En qué ha estado trabajando mi equipo desde la última reunión?
- 2) ¿Qué va a hacer mi equipo hasta la próxima reunión?
- 3) ¿Con qué contaban otros equipos que mi equipo hiciera que no se ha hecho?
- 4) ¿Qué piensa hacer nuestro equipo que pueda afectar a otros equipos?

Las respuestas a estas cuatro preguntas proporcionan información que permite a cada equipo entender claramente la situación laboral de todos los demás equipos.

11.1.2.3 Videoconferencia

Descripción en la sección 10.2.2.4

Es muy probable que la reunión *Scrum of Scrums* (SoS) no sea cara a cara. La videoconferencia es generalmente necesaria para grandes proyectos en los que existe una mayor posibilidad de que haya equipos distribuidos.

11.1.2.4 Sala de Conferencias

Se recomienda tener una sala que sea sólo para conferencias que esté a disposición para la reunión SoS, donde todos los *Equipo Scrum Representatives* se sientan cómodos.

11.1.2.5 Cuerpo de asesoramiento de Scrum Expertise

También se describe en la sección 8.4.2.7.

En el proceso *Convocar Scrum de Scrums*, *Cuerpo de asesoramiento de Scrum Expertise* podría estar relacionado con las mejores prácticas documentadas acerca de cómo llevar a cabo las reuniones de *Scrum of Scrums* (SoS) y de cómo incorporar las sugerencias de esas reuniones en los trabajos de proyecto de los *Equipos Scrum*. También puede haber un equipo de expertos en una materia que pueda ayudar al *Jefe Master Scrum* a facilitar la reunión SoS.

11.1.3 Salidas

11.1.3.1 *Better Team Coordination**

La reunión de *Scrum of Scrums (SoS)* facilita la coordinación de trabajo entre varios *Equipos Scrum*. Esto es especialmente importante cuando hay tareas que impliquen dependencias entre equipos. Las incompatibilidades y discrepancias entre el trabajo y los resultados de los diferentes equipos quedan expuestas rápidamente. Este foro también les da a los equipos la oportunidad de mostrar sus logros y de dar retroalimentación a los otros equipos. Mediante el uso de la reunión SoS, las organizaciones tienen más *Colaboración* que las personas que trabajan en equipos cerrados donde esencialmente se preocupan por sus propias responsabilidades.

11.1.3.2 *Resolved Issues*

Scrum of Scrums (SoS) Meeting es un foro donde los miembros del *Equipo Scrum* tienen la oportunidad de debatir de forma transparente los *issues* que influyen en su proyecto. La necesidad de ofrecer cada *Sprint* a tiempo obliga a los equipos a enfrentarse con tales *issues* cuando surgen en vez de posponer la resolución. Esta discusión oportuna y la resolución de *issues* en la reunión SoS mejora en gran medida la coordinación entre los diferentes *Equipos Scrum*, y también reduce la necesidad de crear un nuevo trabajo y diseño. En esta reunión, *risks* relacionados con las dependencias y horarios de entrega son también mitigados.

SoS Meetings se describen con más detalle en la sección 3.7.2.1.

11

11.1.3.3 *Updated Impediment Log (Registro de impedimento actualizado)*

Descrito en la sección 10.1.3.3.

11.1.3.4 *Dependencias actualizadas*

Descrito en la sección 9.3.3.3.

11.2 Demostración y validación del Sprint

La figura 11-5 muestra todas las entradas, las herramientas y las salidas para el proceso *Demostración y validación del Sprint*.

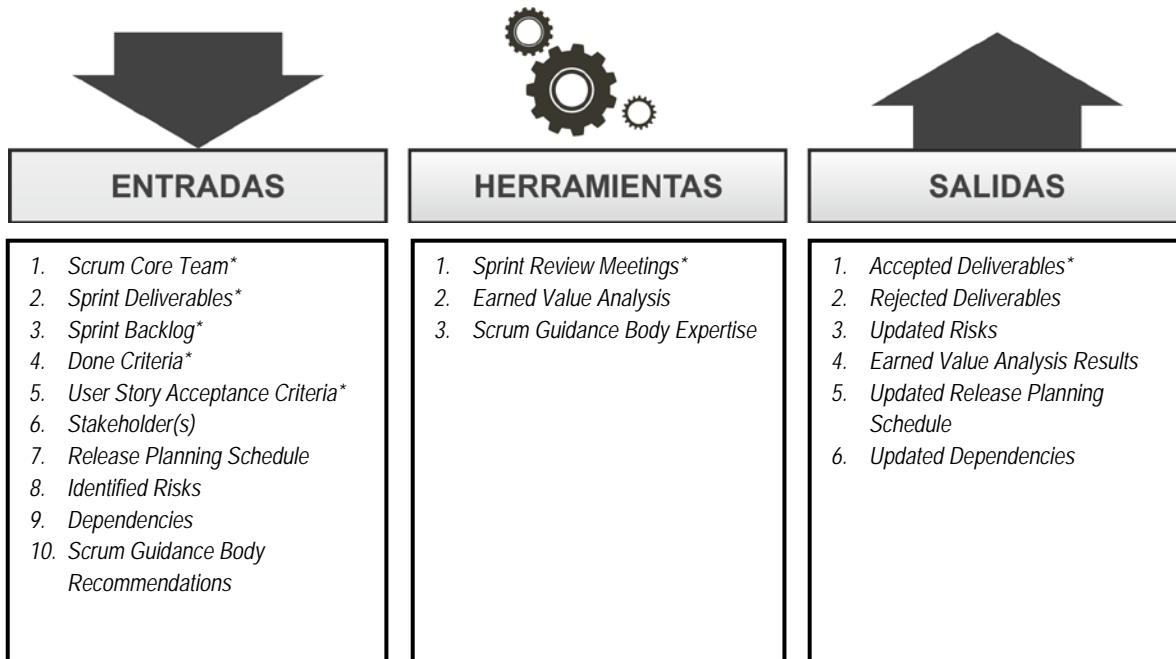


Figura 11-5: *Demostración y validación del Sprint*—Entradas, Herramientas y Salidas

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

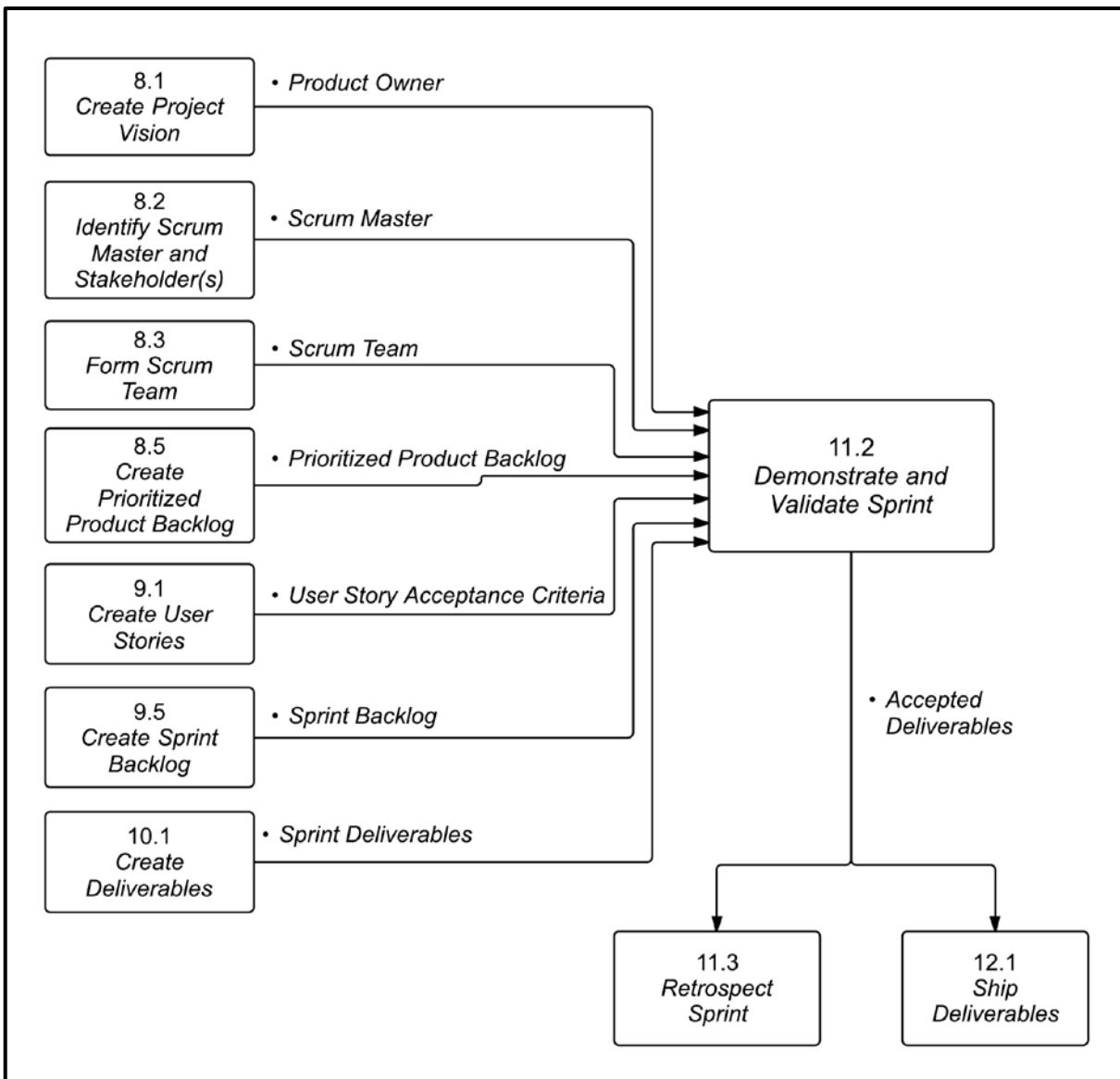


Figura 11-6: Demostración y validación del Sprint—Diagrama de Flujo de Datos

11.2.1 Entradas

11.2.1.1 *Scrum Core Team**

Descrito en la sección 8.4.1.1.

11.2.1.2 *Sprint Deliverables**

Descrito en la sección 10.1.3.1

11.2.1.3 *Sprint Backlog**

Descrito en la sección 9.5.3.1.

11.2.1.4 *Done Criteria**

Descrito en la sección 8.5.3.2.

11.2.1.5 *User Stories Acceptance Criteria**

Descrito en la sección 9.4.1.3.

11.2.1.6 *Stakeholder(s)*

Descrito en la sección 8.2.3.2.

11.2.1.7 *Release Planning Schedule*

Descrito en la sección 8.6.3.1.

11.2.1.8 *Risks identificados*

Descrito en la sección 8.4.3.4.

11.2.1.9 Dependencias

Descrito en la sección 9.3.3.3

11.2.1.10 Cuerpo de asesoramiento de Scrum Recommendations

Descrito en la sección 8.1.1.12

En el proceso de *Demostración y validación del Sprint*, *Cuerpo de asesoramiento de Scrum Recommendations* puede incluir las mejores prácticas sobre cómo llevar a cabo un *Sprint Review Meetings* y de cómo evaluar los resultados de *Earned Value Analysis*. Además, puede haber una orientación acerca de cómo compartir experiencias con otras personas en el *Scrum Core Team* y con otros *Equipos Scrum* en el proyecto.

11.2.2 Herramientas

11.2.2.1 Sprint Review Meeting*

Los miembros del *Scrum Core Team* y el/los *Stakeholder(s)* correspondiente(s) participan en *Sprint Review Meetings* para aceptar los entregables en acuerdo con los *User Story Acceptance Criteria*, y donde se rechazan las entregas inaceptables. Estas reuniones son convocadas al final de cada Sprint. El *Equipo Scrum* demuestra los logros del *Sprint*, incluyendo las nuevas funcionalidades o productos creados. Esto proporciona una oportunidad para que el *Propietario del producto* y el/los *Stakeholder(s)* inspeccionen lo que se ha completado hasta el momento, y para determinar si algún cambio se debe hacer en el proyecto o procesos en *Sprints* posteriores.

11.2.2.2 Earned Value Analysis

Descrito en la sección 4.6.1

11.2.2.3 Cuerpo de asesoramiento de Scrum Expertise

Descrito en la sección 8.4.2.7.

En el proceso *Demostración y validación del Sprint*, *Cuerpo de asesoramiento de Scrum Expertise* podría relacionarse con las mejores prácticas documentadas acerca de cómo llevar a cabo *Sprint Review*

Meetings. También puede haber algunos expertos que podrían ayudar a proporcionar orientación sobre cómo facilitar mejor un *Sprint Review Meeting*.

11.2.3 Salidas

11.2.3.1 Accepted Deliverables*

Los entregables (*deliverables*) que cumplen con los *User Story Acceptance Criteria* son aceptados por el *Propietario del producto*. El objetivo de un *Sprint* es crear entregables potencialmente listos para ser entregados, o presentar incrementos de productos que cumplan con los *Acceptance Criteria* definidos por el *customer* y el *Propietario del producto*. Estos son considerados *Accepted Deliverables* que puedan entregarse al *customer* si así lo desean. Una lista de los *Acceptance Criteria* se mantiene y se actualiza después de cada *Sprint Review Meeting*. Si una entrega no cumple con los *Acceptance Criteria* definidos, no se considera aceptado y por lo general se lleva adelante en un *Sprint* posterior para rectificar cualquier *issue*. Eso no es ideal ya que el objetivo de todos los *Sprint* es cumplir con los criterios de aceptación.

11.2.3.2 Rejected Deliverables

Si los entregables no cumplen con los *Acceptance Criteria*, tales entregables son rechazados. *User Stories* asociados con tales *Rejected Deliverables* se añaden al *Prioritized Product Backlog*, para que dichas entregas se pueden considerar como parte de un *Sprint* posterior.

11.2.3.3 Updated Risks

Descrito en la sección 8.4.3.4.

11.2.3.4 Earned Value Analysis Results

Descrito en la sección 4.6.1.

11.2.3.5 Updated Release Planning Schedule

Descrito en la sección 10.3.3.2.

11.2.3.6 Dependencias actualizadas

Descrito en la sección 9.3.3.3.

11.3 Retrospectiva de Sprint

La figura 11-7 muestra todas las entradas, las herramientas y las salidas para el proceso *Retrospectiva de Sprint*.

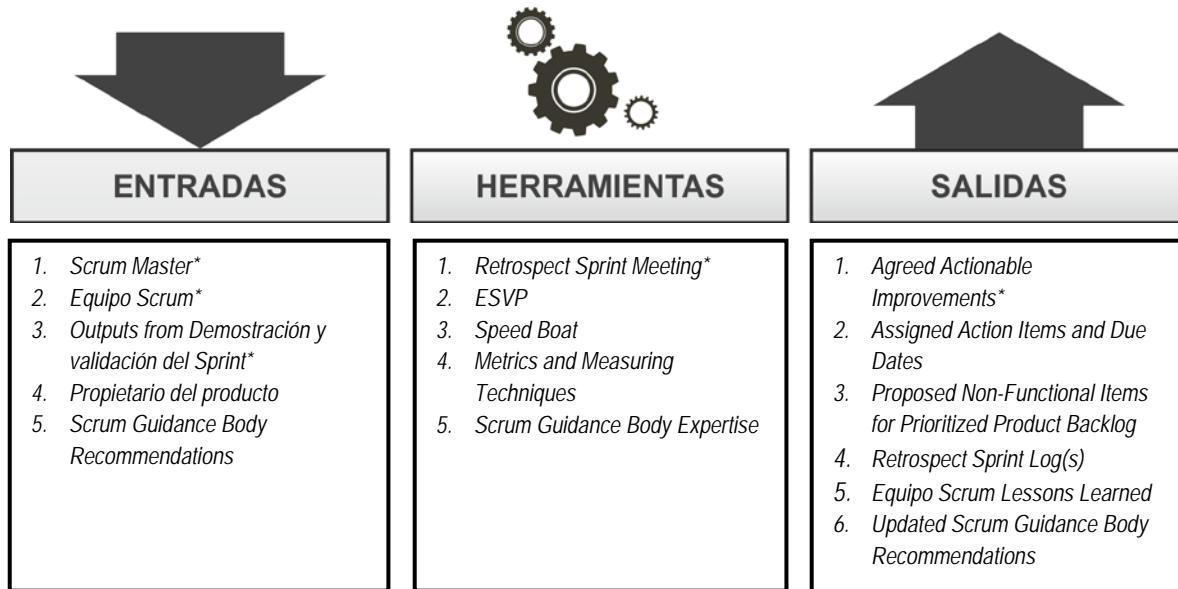


Figura 11-7: *Retrospectiva de Sprint*—Entradas, Herramientas y Salidas

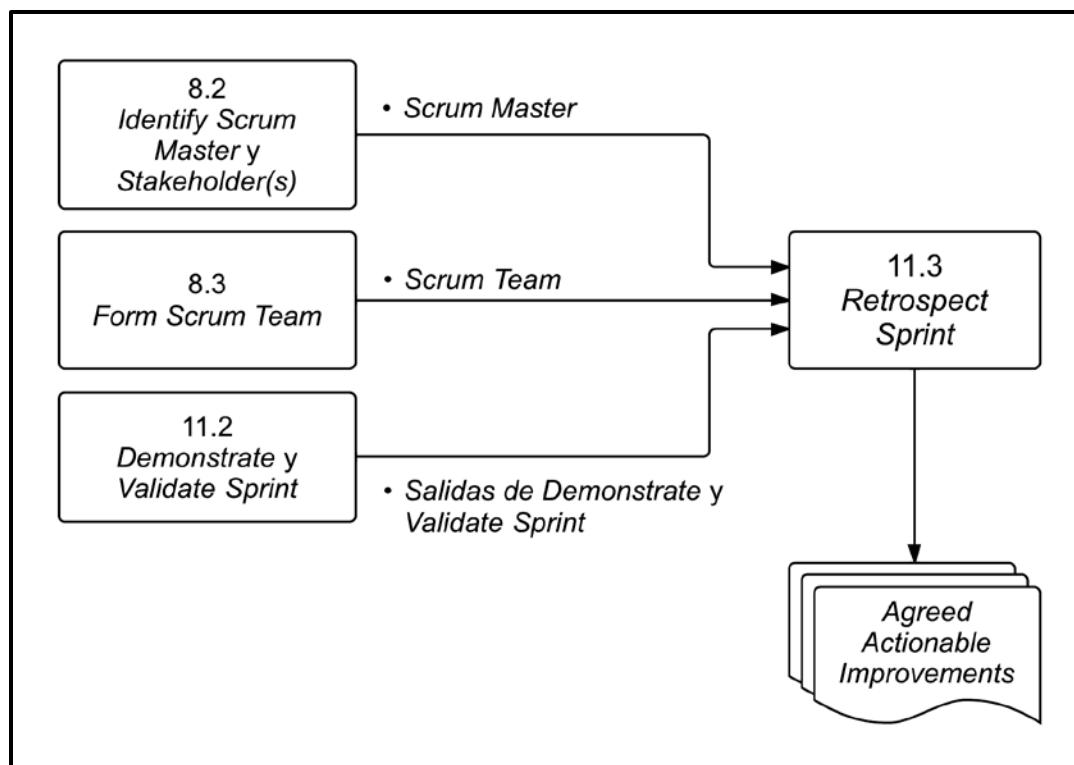


Figura 11-8: *Retrospectiva de Sprint*—Diagrama de Flujo de Datos

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

11.3.1 Entradas

11.3.1.1 *Scrum Master**

Descrito en la sección 8.2.3.1.

11.3.1.2 *Equipo Scrum**

Descrito en la sección 8.3.3.1.

11.3.1.3 Las salidas de *Demostración y validación del Sprint**

Descrito en la sección 11.2.3.

Los resultados/salidas del proceso *Demostración y validación del Sprint* proporcionan información valiosa mientras se realiza el proceso de *Retrospectiva de Sprint*.

11.3.1.4 *Propietario del producto*

Descrito en la sección 8.1.3.1.

11.3.1.5 *Cuerpo de asesoramiento de Scrum Recommendations*

El *Cuerpo de asesoramiento de Scrum* puede proporcionar las directrices para dirigir *Retrospect Sprint Meetings*, incluyendo sugerencias sobre herramientas que se utilizarán y la documentación o resultados que se esperan de las reuniones.

11.3.2 Herramientas

11.3.2.1 *Retrospectiva de Sprint Meeting**

Retrospectiva de Sprint Meeting es un elemento importante del marco de Scrum llamado "inspeccionar-adaptar" y es el paso final en un *Sprint*. Todos los miembros del *Equipo Scrum* asisten a la reunión, que es facilitada o moderada por el *Scrum Master*. Se recomienda, pero no se requiere que el *Propietario del producto* asista. Un miembro del equipo documenta las charlas y los elementos para acciones futuras. Es esencial tener esta reunión en un ambiente abierto y relajado para obtener la participación de todos los

miembros del equipo. Los debates en el *Retrospectiva de Sprint Meetings* abarcan tanto lo que salió mal como lo que salió bien. Los objetivos principales de la reunión son identificar tres elementos específicos:

- 1) Las cosas que el equipo tiene que seguir haciendo: mejores prácticas
- 2) Las cosas que el equipo necesita empezar a hacer: mejoras en el proceso
- 3) Las cosas que el equipo necesita dejar de hacer: problemas de proceso y embotellamiento

Estas áreas se discuten y una lista de *Agreed Actionable Improvements* es creada.

11.3.2.2 Explorer—Shopper—Vacationer—Prisoner (ESVP)

Este es un ejercicio que puede realizarse al inicio del *Retrospectiva de Sprint Meeting* para entender la mentalidad de los participantes y establecer el tono de la reunión. Se les pide a los asistentes que indiquen de forma anónima el término que mejor representa cómo se sienten con respecto a su participación en la reunión.

- Explorer—quiere participar y aprender todo lo que se discutió en la retrospectiva
- Shopper—quiere escuchar todo y elegir lo que desea de la retrospectiva
- Vacationer—quiera relajarse y ser turista en la retrospectiva
- Prisoner—quiere estar en otro lugar y asiste a la retrospectiva ya que se requiere

El *Scrum Master* luego recopila las respuestas, prepara y comparte la información con el grupo.

11.3.2.3 Speed Boat

Speed Boat es una técnica que se puede utilizar para llevar a cabo el *Retrospect Sprint Meeting*. Los miembros del equipo hacen que son tripulantes en un *speed boat*. El barco debe llegar a una isla, lo cual es simbólico de la visión del *project*. Las notas adhesivas son utilizadas para registrar los "motores" y "anclas". Los motores les ayudan a llegar a la isla, mientras que los anclajes les impiden llegar a la isla. Este ejercicio es *Time-boxed* a unos pocos minutos. Una vez que todos los elementos están documentados, la información se analiza y prioriza mediante un proceso de votación. Los motores se reconocen y se planifican las acciones de mitigación de los anclajes, dependiendo de la prioridad.

11.3.2.4 Métricas y técnicas de medición

Varios indicadores pueden ser utilizados para medir y contrastar el desempeño del equipo en el *Sprint* actual a desempeños en *Sprints* anteriores. Algunos ejemplos de estas métricas incluyen:

- *Team velocity*—Número de *Story Points* hecho en un determinado *Sprint*
- *Done success rate*—Porcentaje de los *Story Points* que han sido *Done*, en relación a los que se han llevado a cabo.

- Estimación de eficacia—Número o porcentaje de discrepancia entre el tiempo previsto y el tiempo verdadero que se ha utilizado en las tareas y *User Stories*
- Clasificación de retroalimentación de repaso—La retroalimentación puede ser solicitada por el/los *Stakeholder(s)*, usando calificaciones cuantitativas o cualitativas que proporcionan una medición del rendimiento del equipo.
- Calificación de la moral del equipo—El resultado de autoevaluaciones de la moral del equipo
- Retroalimentación de los compañeros—Retroalimentación de 360 grados se puede utilizar para solicitar información y crítica constructiva sobre el rendimiento del equipo
- Progreso para la liberación o el lanzamiento—El valor del negocio proporcionado en cada versión, así como el valor representado por el progreso actual hacia una liberación. Esto contribuye a la motivación del equipo y el nivel de satisfacción en el trabajo.

11.3.2.5 *Cuerpo de asesoramiento de Scrum Expertise*

También se describe en la sección 8.4.2.7.

En el proceso de *Retrospectiva de Sprint*, *Cuerpo de asesoramiento de Scrum Expertise* podría estar relacionado con las mejores prácticas sobre cómo llevar a cabo *Retrospectiva de Sprint Meetings*. También puede haber algunos expertos que podrían ayudar a proporcionar orientación sobre la forma en la que se utilizan las herramientas del proceso *Retrospectiva de Sprint* para entregar *Agreed Actionable Improvements* para *Sprints* futuros.

11.3.3 Salidas

11.3.3.1 *Agreed Actionable Improvements**

Agreed Actionable Improvements es el primer resultado del proceso *Sprint Retrospect*. Es la lista de elementos configurables que el equipo ha llegado a hacer frente a los problemas y mejorar los procesos con el fin de mejorar su desempeño en el futuro *Sprints*.

11.3.3.2 *Assigned Action Items and Due Dates*

Una vez que *Agreed Actionable Improvements* se han elaborado y refinado, los puntos de acción para aplicar las mejoras pueden ser considerados por el *Equipo Scrum*. Cada elemento de acción tendrá una fecha de vencimiento definida para su conclusión.

11.3.3.3 Elementos no funcionales propuestos para el *Prioritized Product Backlog*

Cuando el *Prioritized Product Backlog* inicial se desarrolla, se basa en *User Stories* y funcionalidades requeridas. A menudo, los requisitos no funcionales pueden no ser totalmente definidos en las primeras etapas del proyecto y pueden surgir durante el *Sprint Review* o *Retrospectiva de Sprint Meetings*. Estos artículos deben ser añadidos al *Prioritized Product Backlog* a medida que se descubren. Algunos ejemplos de los requisitos no funcionales son los tiempos de respuesta, las limitaciones de capacidad y de seguridad relacionados con *issues*.

11.3.3.4 *Retrospectiva de Sprint Log(s)*

El *Retrospectiva de Sprint Log* es un registro de las opiniones, discusiones y artículos recurribles planteados en un *Retrospectiva de Sprint Meeting*. El *Scrum Master* podría facilitar la creación de este registro con entradas del *Scrum Core Team*. La colección de todos los registros retrospectivos *Sprint* se convierte en la agenda del proyecto y de los detalles exitosos del proyecto, al igual que los *issues*, problemas y resoluciones. Los registros son documentos públicos a disposición de cualquier persona en la organización.

11.3.3.5 *Equipo Scrum Lessons Learned*

Se asume que un *Equipo Scrum* auto-organizado y poderoso llegará a aprender de los errores cometidos durante el *Sprint*. Estas lecciones aprendidas ayudan a los equipos a mejorar su desempeño en futuros *Sprints*. Estas lecciones aprendidas también pueden ser documentadas en *Cuerpo de asesoramiento de Scrum Recommendations* para compartir con otros *Equipos Scrum*.

Puede haber varias lecciones positivas aprendidas como parte de un *Sprint*. Estas lecciones positivas aprendidas son una parte clave de la retrospectiva, y deben ser adecuadamente compartidas dentro del equipo y con el *Cuerpo de asesoramiento de Scrum*, ya que los equipos trabajan hacia la auto-mejora continua.

11.3.3.6 *Updated Cuerpo de asesoramiento de Scrum Recommendations*

Como resultado de un *Retrospectiva de Sprint Meeting*, las sugerencias se pueden hacer para revisar o mejorar las *Cuerpo de asesoramiento de Scrum Recommendations*. Si el Conjunto de Orientación (*Guidance Body*) acepta estas sugerencias, éstas se incorporarán como cambios a la documentación *Cuerpo de asesoramiento de Scrum*.

11.4 Fase Diagrama de flujo de Datos

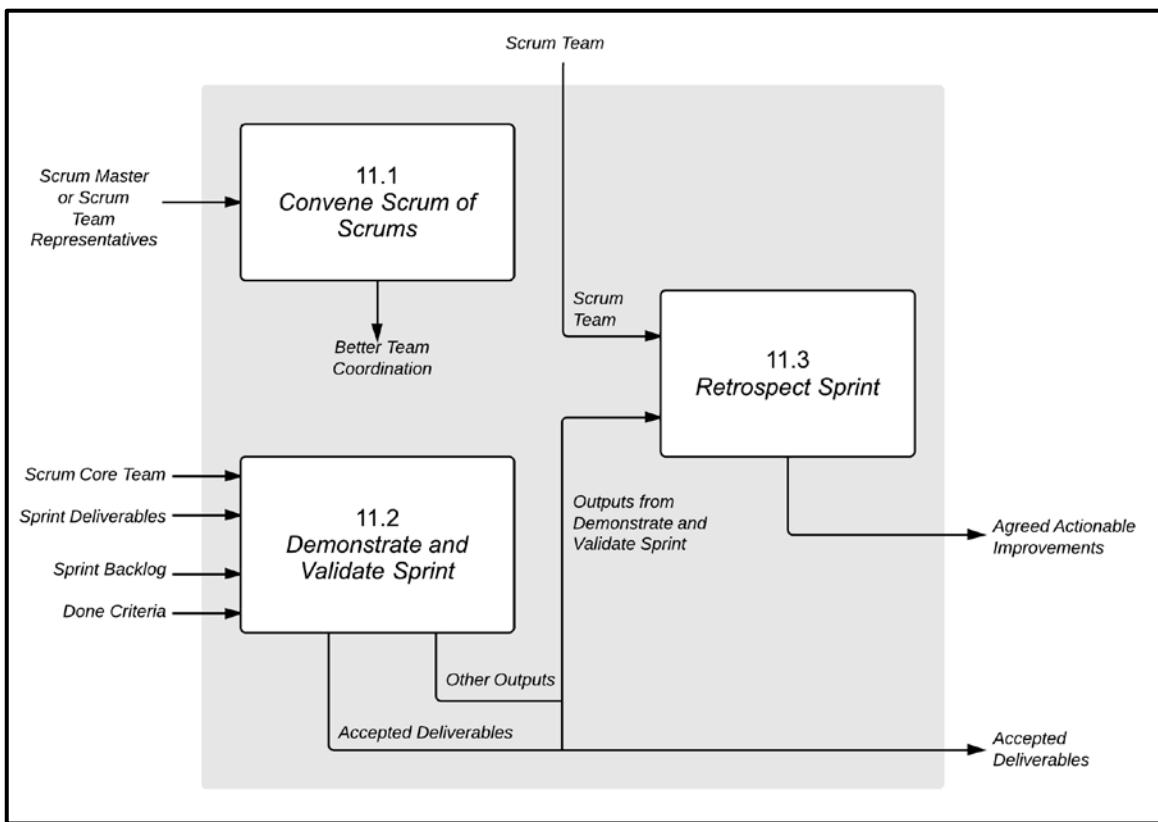


Figura 11-9: Fase Review and Retrospect—Diagrama de flujo de datos

12. LANZAMIENTO

La fase de lanzamiento (*release*) destaca la entrega de los *Accepted Deliverables* al *customer* y la identificación, documentación, e internalización de las lecciones aprendidas durante el proyecto.

Lanzamiento, tal como se define en la *Guía de los Fundamentos de la Scrum del Conocimiento (Guía SBOK™)*, es aplicable a los siguientes:

- *Portfolios, programs y/o projects* de cualquier sector
- *Products*, servicios o cualquier otro resultado que se entregarán a los *socios*
- *Projects* de cualquier tamaño y complejidad

El término "producto" en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación. Scrum se puede aplicar de manera efectiva a cualquier proyecto en cualquier industria-desde pequeños proyectos o equipos con tan sólo seis miembros por equipo, hasta proyectos grandes y complejos que cuentan con cientos de miembros por equipo.

Con el fin de facilitar la mejor aplicación del marco de Scrum, en este capítulo se identifican las entradas, herramientas y salidas de cada proceso, ya sea como algo "obligatorio" o "facultativo". Las entradas, herramientas y salidas indicadas por asteriscos (*) son obligatorias, mientras que los que no tienen asteriscos son opcionales.

Se recomienda que el *Equipo Scrum* y aquellas personas que estén en el inicio de su aprendizaje sobre el marco y los procesos de Scrum se centren principalmente en las aportaciones obligatorias, las herramientas y los productos; mientras que los *Propietario del producto*, *Scrum Masters*, y otros practicantes de Scrum con experiencia se esfuerzen por alcanzar un conocimiento más profundo de la información en todo este capítulo. También es importante darse cuenta de que aunque todos los procesos se definen de forma única en la *Guía SBOK™*, no necesariamente se llevan a cabo de forma secuencial o por separado. A veces, puede ser más apropiado combinar algunos procesos, dependiendo de los requisitos específicos de cada proyecto.

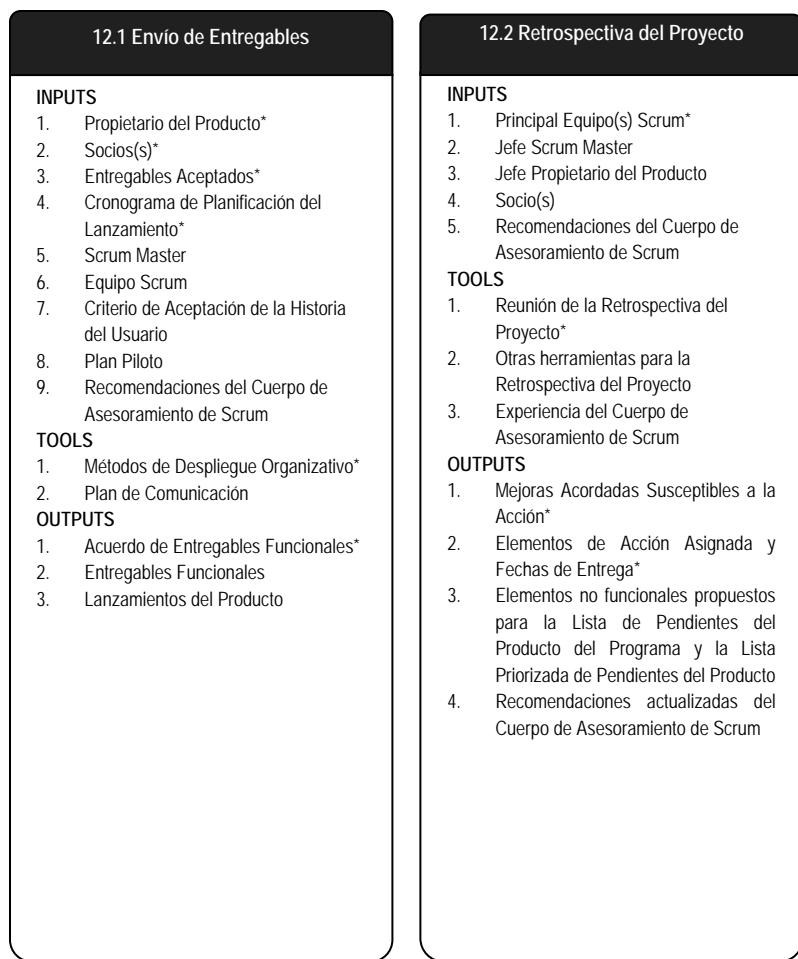
12

Este capítulo está escrito desde la perspectiva de un *Equipo Scrum* que está trabajando en un *Sprint* para producir entregables potencialmente listos para enviar como parte de un *project* más amplio. Sin embargo, la información que se describe es igualmente aplicable a proyectos completos, *programs* y *portfolios*. Información adicional relacionada con el uso de Scrum para *projects*, *programs* y *portfolios* está disponible en los capítulos 2 al 7, que cubren los principios y aspectos de Scrum.

La figura 12-1 proporciona una visión general de los procesos en fase de lanzamiento, que son los siguientes:

12.1 Envío de entregables—En este proceso, los *Accepted Deliverables* se les entregan o trasladan a los *socios* pertinentes. Un *Working Deliverable Agreement* formal documenta la finalización con éxito del *Sprint*.

12.2 Retrospectiva del proyecto—En este proceso, que completa el proyecto, los *socios* de la organización y el *Scrum Core Team* se reúnen para la retrospectiva del *project* e identificar, documentar e internalizar las lecciones aprendidas. A menudo, estas lecciones llevan a la documentación de *Agreed Actionable Improvements*, que se aplicarán en futuros proyectos.

Figura 12-1: Resumen de *Release* (Lanzamiento)

La figura 12-2 muestra las entradas obligatorias, herramientas y salidas para los procesos en fase de lanzamiento.

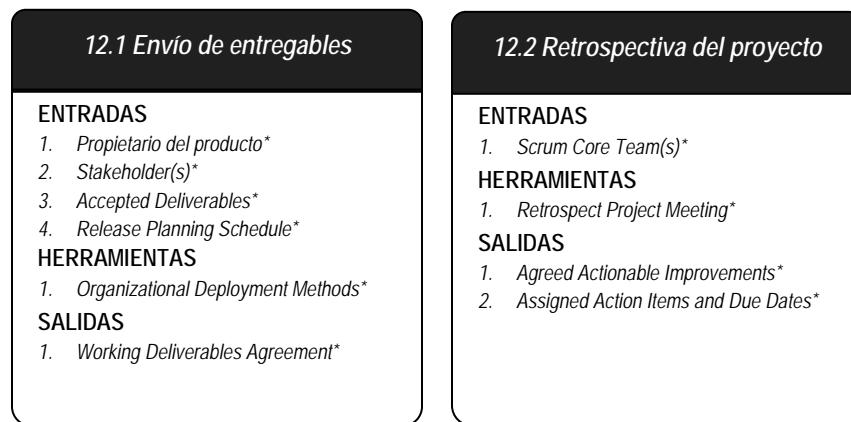


Figura 12-2: Resumen de Release (Esenciales)

Nota: Los asteriscos () denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.*

12.1 Envío de entregables

La figura 12-3 muestra todas las entradas, las herramientas y las salidas para el proceso de *Envío de entregables*.



Figura 12-3: *Envío de entregables*—Entradas, Herramientas y Salidas

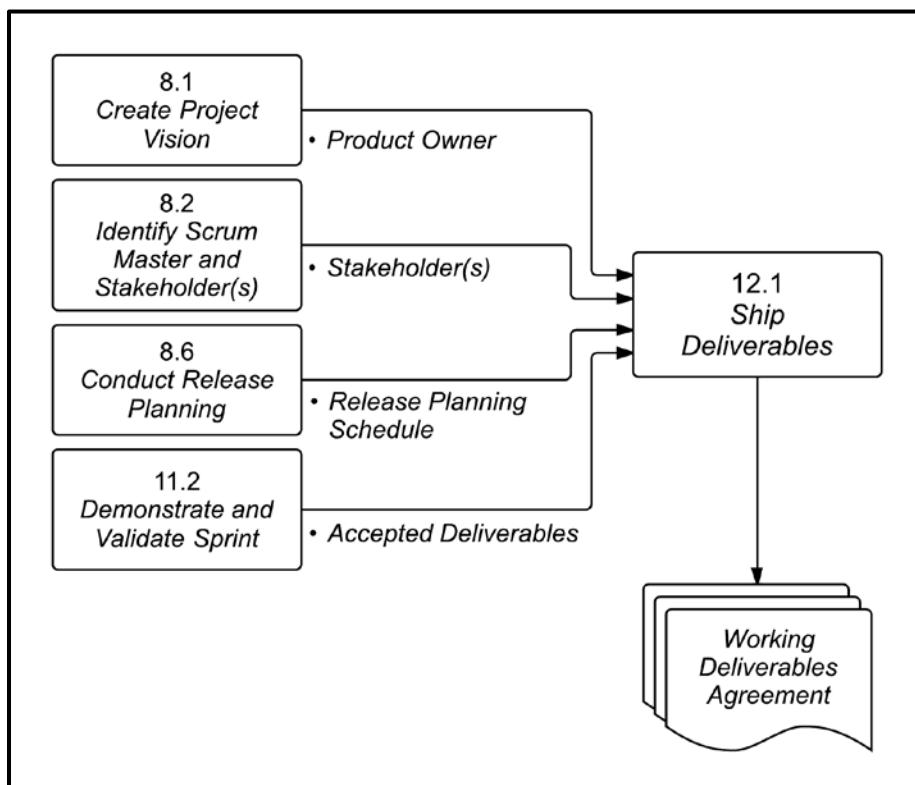


Figura 12-4: *Envío de entregables*—Diagrama de flujo de datos

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

12.1.1 Entradas

12.1.1.1 *Propietario del producto**

Descrito en la sección 8.1.3.1.

12.1.1.2 *Stakeholder(s)**

Descrito en la sección 8.2.3.2.

12.1.1.3 *Accepted Deliverables**

Descrito en la sección 11.2.3.1.

12.1.1.4 *Release Planning Schedule*

Descrito en la sección 8.6.3.1.

12.1.1.5 *Scrum Master*

Descrito en la sección 8.2.3.1.

12.1.1.6 *Equipo Scrum*

Descrito en la sección 8.3.3.1.

12.1.1.7 *User Story Acceptance Criteria*

Descrito en la sección 9.1.3.2.

12.1.1.8 Piloting Plan

Un *Piloting Plan* es una entrada (*input*) opcional que se puede utilizar para trazar una implementación piloto en detalle. El alcance y los objetivos del despliegue, *target deployment user base* (despliegue objetivo la base de usuarios), un cronograma de implementación, planes de transición, preparación de usuario necesaria, los criterios de evaluación para el despliegue, y otros elementos claves relacionados con el despliegue se especifican en el *Piloting Plan* y son compartidos con los socios.

12.1.1.9 Cuerpo de asesoramiento de Scrum Recommendations

Descrito en la sección 8.1.1.12.

En el proceso de *Envío de entregables*, el *Cuerpo de asesoramiento de Scrum* puede proporcionar recomendaciones y directrices para la implementación de productos. Estas son las mejores prácticas que se deben tener en cuenta al implementar un *product* al *customer* con el fin de maximizar el valor entregado.

12.1.2 Herramientas

12.1.2.1 Organizational Deployment Methods*

Los mecanismos de despliegue de cada organización tienden a ser diferentes en función de su industria, los usuarios en mente y posicionamiento. Dependiendo del producto que se entrega, el despliegue puede tener lugar de forma remota o puede implicar el envío físico o transición de un elemento. Debido a que la implementación tiende a implicar un alto nivel de riesgo, las organizaciones suelen tener mecanismos de implementación bien definidos y establecidos, con procesos detallados para garantizar el cumplimiento de todas las normas aplicables y medidas de *quality assurance*. Estos podrían incluir aprobaciones finales de los representantes específicos de gestión, mecanismos de aprobación de usuario, y directrices para la funcionalidad mínima de un comunicado.

12

12.1.2.2 Communication Plan

En muchos proyectos, un *Communication Plan* existe. Este plan especifica los registros que deben ser creados y mantenidos durante todo el *project*. Una variedad de métodos se utilizan para transmitir información importante del *project* a los socios. El *Communication Plan* define estos métodos, así como quién es el responsable de varias actividades de comunicación. Como los Entregables se ponen a prueba, el estado de las actividades de prueba se comunica según el *Communication Plan* según lo determinado por el *Propietario del producto* y patrocinador. Un mecanismo común de comunicación es tener algo visual que represente información importante en un formato fácil de interpretar, publicado en un lugar accesible, y que se mantenga al día con la información más actual.

12.1.3 Salidas

12.1.3.1 *Working Deliverables Agreement**

Las entregas que cumplen con los *Acceptance Criteria* reciben un cierre de negocio formal y la aprobación por parte del *customer* o patrocinador. El alcanzar una aceptación formal del *customer* es fundamental para el reconocimiento de ingresos. La responsabilidad para su obtención será definida por las políticas de la empresa y no es necesariamente la responsabilidad del *Propietario del producto*.

12.1.3.2 *Working Deliverables*

Esta salida es el Entregable final para el que fue aprobado el proyecto. A medida que se crean nuevos incrementos de los productos, son integrados continuamente a los incrementos anteriores, por lo que hay un producto potencialmente entregable disponible en todo momento a lo largo del *project*.

12.1.3.3 Lanzamientos del producto

Los lanzamientos de productos deben incluir lo siguiente:

- *Release Content*—Esto consiste en la información esencial acerca de las prestaciones que pueden ayudar al Equipo de Apoyo al Cliente (*customer*).
- *Release Notes*—*Release Notes* debe incluir criterios externos o del mercado de envío del *product* a entregar.

12.2 Retrospectiva del proyecto

La figura 12-5 muestra todas las entradas, las herramientas y las salidas para el proceso de *Retrospectiva del proyecto*.

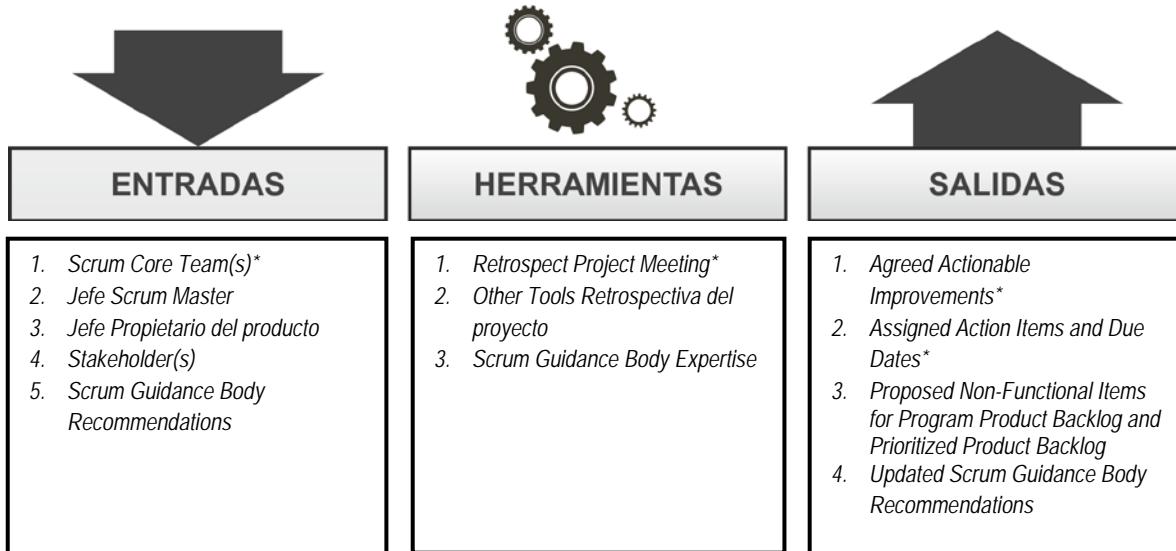


Figura 12-5: *Retrospectiva del proyecto*—Entrada, Herramientas y Salidas

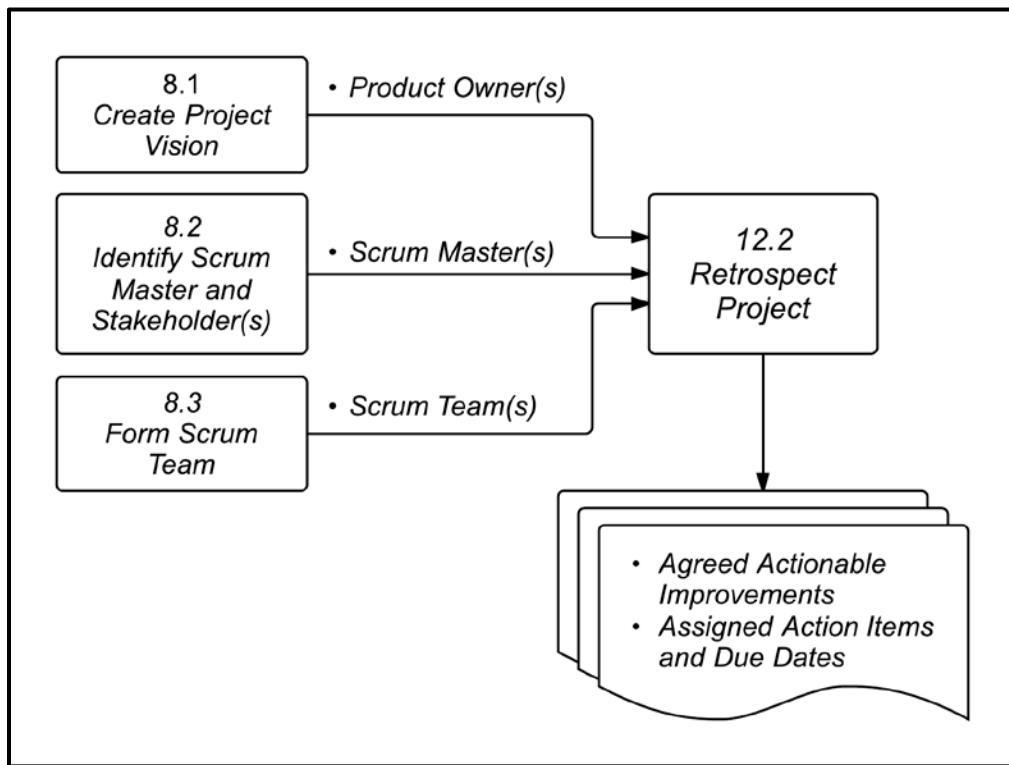


Figura 12-6: *Retrospectiva del proyecto*—Diagrama de flujo de datos

Nota: Los asteriscos (*) denotan una entrada, herramientas o salida “obligatoria” para el proceso correspondiente.

12.2.1 Entradas

12.2.1.1 *Scrum Core Team(s)**

Descrito en la sección 8.4.1.1.

12.2.1.2 *Jefe Scrum Master*

Descrito en la sección 8.2.1.6.

12.2.1.3 *Jefe Propietario del producto*

Descrito en la sección 8.1.1.5.

12.2.1.4 *Stakeholder(s)*

Descrito en la sección 8.2.3.2.

12.2.1.5 *Cuerpo de asesoramiento de Scrum Recommendations*

Descrito en la sección 8.1.1.12.

En el proceso de *Retrospectiva del proyecto*, las recomendaciones del *Cuerpo de asesoramiento de Scrum* pueden incluir un repositorio de plantillas internas que apoyan los proyectos futuros, al igual que orientación para la realización del *Retrospectiva del proyecto Meeting*. La orientación proporcionada puede relacionarse con los procedimientos administrativos, auditorías, evaluaciones y los criterios de transición proyecto. A menudo, también incluyen cómo la organización va a mantener la base de conocimiento de lecciones aprendidas y de información sobre todos los proyectos.

12.2.2 Herramientas

12.2.2.1 *Retrospectiva del proyecto Meeting**

Retrospectiva del proyecto Meeting es una reunión para determinar las formas en que *Colaboración* y la eficacia entre el equipo puede mejorar en futuros proyectos. También se discuten aspectos positivos, negativos y posibles *opportunities* de mejora. Esta reunión no es *Time-boxed* y se puede realizar en persona o en un formato virtual. Entre los asistentes figuran el Equipo del Proyecto, Jefe *Scrum Master*, Jefe *Propietario del producto*, y *Stakeholder(s)*. Durante la reunión, las lecciones aprendidas se documentan y los participantes buscan *opportunities* para mejorar los procesos y atender las ineficiencias.

12.2.2.2 Otras herramientas para *Retrospectiva del proyecto*

Algunas de las herramientas utilizadas en el proceso de *Sprint Retrospect* también se pueden utilizar en este proceso. A continuación se ofrecen ejemplos:

- Ejercicios de *Explorer—Shopper—Vacationer—Prisoner (ESVP)*
- *Speed Boat*
- Métricas y técnicas de medición

12.2.2.3 *Cuerpo de asesoramiento de Scrum Expertise*

12

Discutido en la sección 8.4.2.7.

En el proceso de *Retrospectiva del proyecto*, la responsabilidad principal del *Cuerpo de asesoramiento de Scrum* es asegurar que las lecciones aprendidas en cada proyecto no se hayan perdido y que estén integradas en la organización.

Además, un conjunto de orientación puede aportar su experiencia en diversos ámbitos, entre ellos la Calidad, Recursos Humanos y Scrum, que pueden ser útiles en el proceso de *Retrospectiva del proyecto*. Además, pueden ofrecerse sugerencias en el *Cuerpo de asesoramiento de Scrum Recommendations* sobre cómo debe llevarse a cabo el *Retrospectiva del proyecto Meeting*.

12.2.3 Salidas

12.2.3.1 *Agreed Actionable Improvements**

Descrito en la sección 11.3.3.1.

12.2.3.2 *Assigned Action Items and Due Dates**

Descrito en la sección 11.3.3.2.

12.2.3.3 Elementos no funcionales propuestos en el *Program Product Backlog* y *Prioritized Product Backlog*

Cuando el *Program Product Backlog* o el *Prioritized Product Backlog* iniciales son desarrollados, se basan en *User Stories* y funcionalidades requeridas. A menudo, los requisitos no funcionales pueden no ser totalmente definidos en las primeras etapas del proyecto y pueden surgir durante el *Sprint Review*, *Retrospectiva de Sprint* o *Retrospectiva del proyecto Meetings*. Estos artículos deben ser añadidos al *Program Product Backlog* (para el programa) y el *Prioritized Product Backlog* (para el proyecto) a medida que se descubren. Algunos ejemplos de los requisitos no funcionales son los tiempos de respuesta, las limitaciones de capacidad e *issues* relacionados con la de seguridad.

12.2.3.4 *Updated Cuerpo de asesoramiento de Scrum Recommendations*

Descritas en las secciones 8.1.1.12 and 11.3.3.5

12.3 Fase Diagrama de flujo de datos

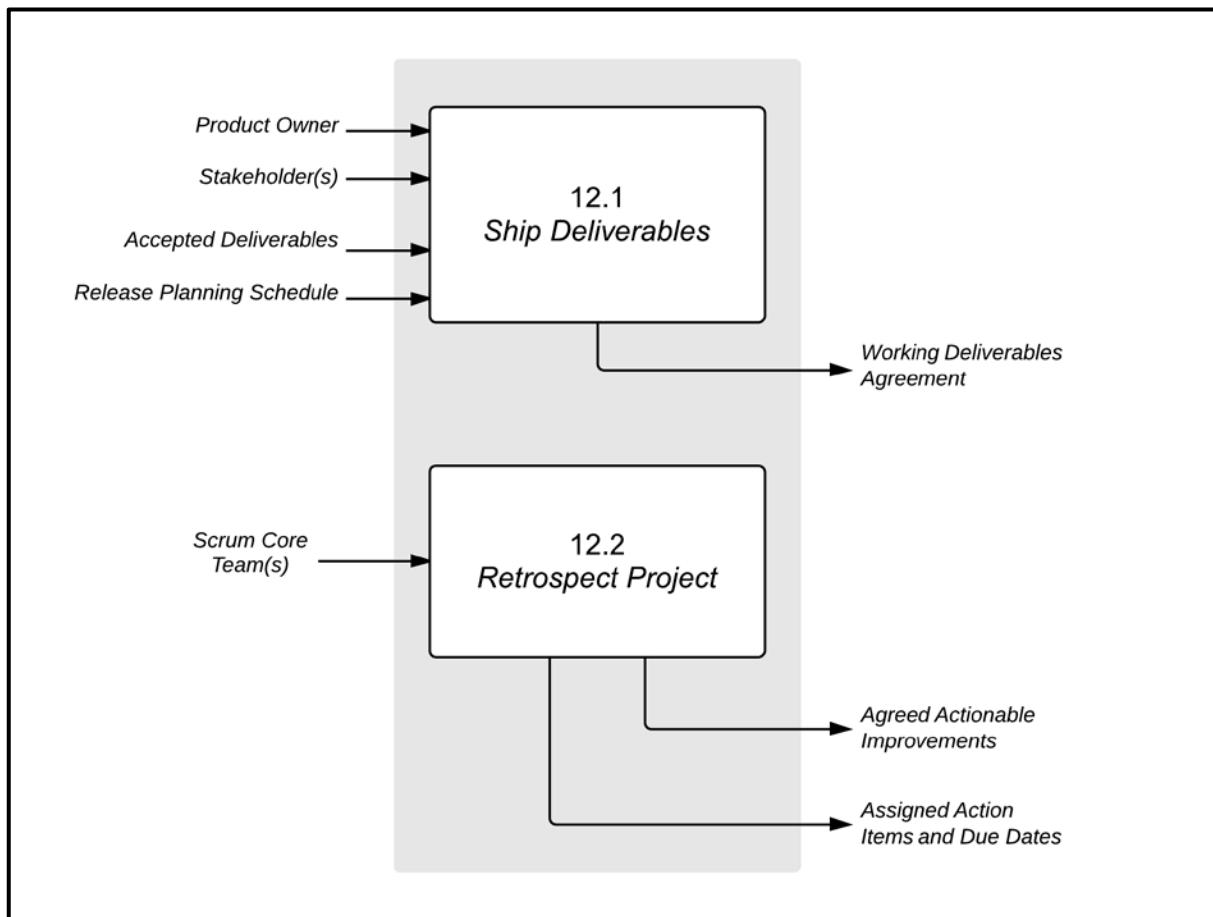


Figura 12-7: *Release Phase* (Fase de lanzamiento)—Diagrama de Flujo de Datos

APÉNDICE A. RESUMEN DE AGILE

A.1 Introducción

Este apéndice tiene la intención de familiarizar a los lectores con el concepto de desarrollo Agile y las diversas metodologías ágiles.

En las siguientes secciones se incluyen:

A.2 Resumen—En esta sección se analiza la definición y los factores que explican el gran interés por Agile.

A.3 Manifiesto Ágil—En esta sección se presenta *el Manifiesto Ágil*, sus principios, y *la Declaración de Interdependencia* para proporcionar el contexto histórico de Agile.

A.4 Agile-Métodos —En esta sección se ofrece una breve descripción de las metodologías ágiles específicas tales como:

- *Lean Kanban*
- *Extreme Programming*
- *Crystal Methods*
- *Dynamic Systems Development Methods*
- *Feature Driven Development*
- *Test Driven Development*
- *Adaptive Software Development*
- *Agile Unified Process*
- *Domain Driven Development*

A.2 Generalidades

El término *agile* (*ágil*) generalmente se refiere a ser capaz de moverse o responder rápidamente y fácilmente. En cualquier tipo de disciplina de gestión, ser ágil es una cualidad, por lo tanto esto debe ser una meta que se debe tratar de alcanzar. La gestión de *projects* Agile especialmente, implica la adaptabilidad durante la creación de un producto, servicio, o cualquier otro resultado.

Es importante entender que a pesar de que el desarrollo de los métodos ágiles es altamente adaptable, de todos modos es necesario tener en cuenta la estabilidad en sus procesos de adaptación.

A.2.1 El gran interés por Agile

Los rápidos cambios en la tecnología, las demandas y expectativas del mercado han creado grandes desafíos en relación a los *products* y servicios en desarrollo que usan los modelos tradicionales de gestión de proyectos. Esto abrió el camino para la conceptualización e implementación de métodos y valores ágiles en muchas organizaciones. Los modelos de desarrollo Agile atienden las deficiencias asociadas con los modelos tradicionales de gestión de *projects* para satisfacer las crecientes demandas ambientales y expectativas que las organizaciones encaran. Dado a que los modelos tradicionales de gestión de *projects* en general hacen hincapié en una amplia planificación por adelantada y que se ajustan al plan una vez que se establece, tales modelos no tuvieron éxito al encarar la realidad de los frecuentes cambios ambientales.

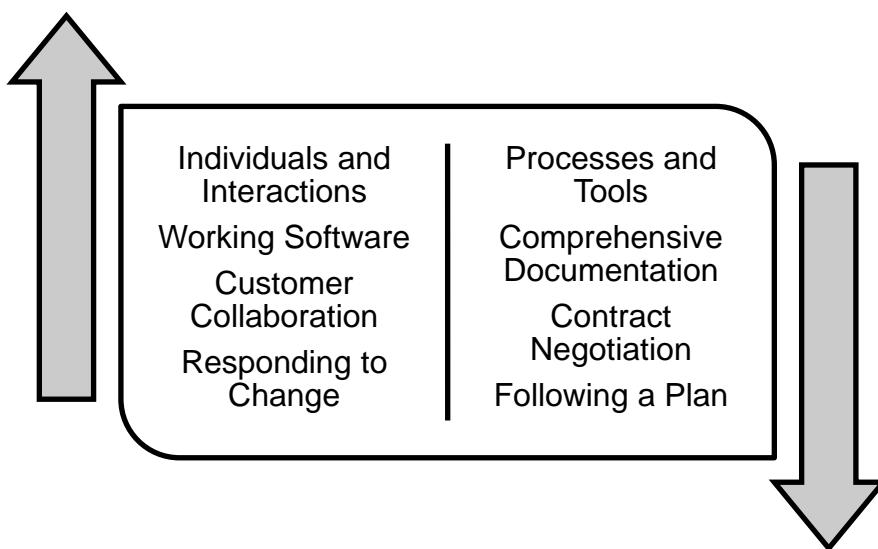
Agile se basa en la planificación de adaptación y en el desarrollo y la entrega iterativa. Se centra principalmente en que las personas hagan el trabajo con eficacia. Aunque las metodologías adaptivas e incrementales han existido desde la década de 1950, sólo las metodologías que se ajustan a *El Manifiesto Ágil* son generalmente consideradas verdaderamente como "Ágil".

A.3 *The Agile Manifesto*

En febrero del 2001, un grupo de 17 gurús de la informática, desarrolladores de software y administradores se reunieron para discutir los métodos de desarrollo de software de peso ligero. Formaron *Agile Alliance* y las deliberaciones de esas reuniones más tarde dieron lugar al *Manifesto for Agile Software Development*. El manifiesto fue escrito por Fowler y Highsmith (2001) y luego fue firmado por todos los participantes para establecer los lineamientos básicos para cualquier metodología Agile.

El propósito de *The Agile Manifesto* fue distribuido de la siguiente manera:

*Estamos descubriendo mejores formas de desarrollar software haciéndolo y ayudando a que otros lo hagan.
A través de este trabajo hemos llegado a valorar:*



Es decir, mientras que hay valor en los elementos de la derecha, valoramos más los elementos a la izquierda.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

El permiso para copiar fue proporcionado por los autores mencionados mediante aviso en <http://agilemanifesto.org/>.

Las cuatro compensaciones de *The Agile Manifesto* se elaboran de la siguiente manera:

1. Individuos e interacciones sobre procesos y herramientas

Aunque los procesos y las herramientas ayudan a completar con éxito un proyecto, son las personas que se dedican, participan, determinar qué procesos y herramientas se han de utilizar e implementan un proyecto. Los principales en cualquier proyecto son, por lo tanto, los individuos, por lo que el énfasis debe estar en ellos y sus interacciones, en lugar de poner énfasis en procesos e herramientas complicados.

2. Software de buen rendimiento sobre la documentación detallada

Aunque la documentación es necesaria y útil para cualquier *project*, muchos equipos se centran en la recopilación y el registro de las descripciones cualitativas y cuantitativas de los entregables, cuando el valor real que se le entrega al *customer* es en forma de un software de buen rendimiento. Por lo tanto, el enfoque Agile se encuentra en la entrega de un software de buen funcionamiento en incrementos a lo largo del ciclo de vida del producto en lugar de la documentación detallada.

3. Client Colaboración sobre la negociación del contrato

Tradicionalmente, los *customers* han sido vistos como participantes exteriores que están envueltos principalmente al inicio y al final del ciclo de vida del *product* y cuyas relaciones se basaban en contratos y el cumplimiento de éstos. Agile cree en un enfoque de valor compartido en el que los *customers* son vistos como colaboradores. El equipo de desarrollo y el *customer* trabajan juntos para evolucionar y desarrollar el *product*.

4. Responder al cambio en vez de seguir un plan

En el mercado actual en el que los requisitos del *customer*, las tecnologías disponibles, y los patrones de negocio están cambiando constantemente, es esencial abordar el desarrollo de *products* de una manera adaptativa que permita la incorporación de cambios y los ciclos de vida de desarrollo de *products* de forma rápida, en lugar de enfatizar el concepto de seguir planes que fueron creados quizás con datos obsoletos.

A.3.1 Principios de *Agile Manifesto*

Los 12 principios del *Agile Manifesto* (Manifiesto Ágil) por Fowler y Highsmith (2001) son los siguientes:

1. Nuestra máxima prioridad es satisfacer al *customer* a través de la entrega temprana y continua de un software de gran utilidad.
2. Darle la bienvenida a requisitos cambiantes incluso tarde en el desarrollo. Los procesos ágiles aprovechan el cambio y lo transforman en una ventaja competitiva para el *customer*.
3. Entregar software de buen funcionamiento con frecuencia, a partir de un par de semanas a un par de meses, con una preferencia por el tiempo más corto.
4. La gente de negocios y los desarrolladores deben trabajar juntos todos los días durante todo el *project*.
5. Construir proyectos alrededor de individuos motivados, darles el entorno y el apoyo que necesitan y confiar en ellos para hacer el trabajo.
6. El método más eficiente y eficaz de comunicación con y dentro de un equipo de desarrollo es cara a cara.
7. Un software funcional es la medida de progreso principal.
8. Los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, desarrolladores y usuarios deben ser capaces de mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y el buen diseño mejora la agilidad.
10. Simplicidad— el arte de maximizar la cantidad de trabajo no realizado—es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
12. En intervalos regulares, el equipo reflexiona sobre cómo ser más eficaz, y en base a eso ajusta su comportamiento.

A.3.2 Declaration of Interdependence

La gestión de proyectos Agile *Declaration of Interdependence* (*Declaración de independencia*) fue escrita a principios del 2005 por un grupo de 15 líderes de projects como un suplemento a *El Manifiesto Ágil*. Enumera seis valores de gestión necesarios para reforzar una mentalidad de desarrollo ágil, particularmente en la gestión de projects complejos e inciertos.

La declaración destaca que los equipos de proyectos, *customers* y otros *stakeholders* son interdependientes y están conectados, algo que deben reconocer para tener éxito. Los valores también son interdependientes.

Nosotros...

aumentamos el Return on Investment, al enfocarnos en el flujo continuo de valor.

ofrecemos resultados fiables mediante la participación de customers en las interacciones frecuentes, donde también son responsables por el trabajo.

asumimos que habrán incertidumbre y las superamos a través de iteraciones, anticipación y adaptación.

damos rienda suelta a la creatividad y la innovación al reconocer que las personas son la fuente máxima de valor y creamos un entorno en el que puedan tener un impacto positivo.

aumentamos el rendimiento a través de la rendición de cuentas por parte del grupo en cuestión de resultados y eficacia del equipo, responsabilidades que todos comparten.

mejoramos la eficacia y la fiabilidad a través de estrategias situacionalmente específicas, procesos y prácticas.

Anderson. D., Augustine, S., Avery, C., Cockburn, A., Cohn, M., et al. 2005

A.4 Métodos Agile

Una serie de metodologías ágiles originó y ganó fuerza en la década de 1990 y principios del 2000. Si bien difieren en una variedad de aspectos, lo que tienen en común se deriva de su adhesión a *The Agile Manifesto*.

Los siguientes métodos ágiles se discuten brevemente a continuación:

1. *Lean Kanban*
2. *Extreme Programming (XP)*
3. *Crystal Methods*
4. *Dynamic Systems Development Methods (DSMD)*
5. *Feature Driven Development (FDD)*
6. *Test Driven Development (TDD)*
7. *Adaptive Software Development (ASD)*
8. *Agile Unified Process (AUP)*
9. *Domain-Driven Design (DDD)*

A.4.1 Lean Kanban

El concepto de Lean optimiza el sistema de una organización para producir resultados valiosos sobre la base de sus recursos, necesidades y alternativas, mientras reduce de las perdidas. Las perdidas, por ejemplo, podrían ser la construcción incorrecta de un producto, el no saber aprender, o las prácticas que impiden el proceso. Debido a que estos factores son de naturaleza dinámica, una organización ágil evalúa la totalidad de su sistema y continuamente hace ajustes de sus procesos. El fundamento de Lean es que la reducción de la longitud de cada ciclo (es decir, una iteración) conduce a un aumento de productividad mediante la reducción de los retrasos, ayuda en la detección de errores en una etapa temprana, y por consecuencia reduce la cantidad total de esfuerzo requerido para terminar una tarea. Los principios de software Lean se han aplicado con éxito en el desarrollo de software.

Kanban significa literalmente un "cartel" o "cartelera" y enfatiza el uso de ayudas visuales para ayudar y realizar un seguimiento de la producción. El concepto fue introducido por Taiichi Ohno, considerado como el padre de los sistemas Toyota Proudction Systems (TPS). El uso de ayudas visuales es eficaz y se ha convertido en una práctica común. Los ejemplos incluyen las tarjetas de tareas, *Scrumboards*, y *Burndown Charts*. Estos métodos recibieron mucha atención debido a su práctica en Toyota, el cual es un líder en gestión de procesos. Lean Kanban integra el uso de los métodos de visualización según lo prescrito por Kanban junto con los principios de Lean creando así un sistema de gestión de proceso evolutivo incremental y visual.

A.4.2 *Extreme Programming*

Extreme Programming (XP), que se originó en Chrysler Corporation, ganó fuerza en la década de 1990. XP hace que sea posible mantener el costo de cambiar el software sin que éste aumente radicalmente con el tiempo. Los atributos claves de XP incluyen el desarrollo gradual, horarios flexibles, pruebas automatizadas de código, la comunicación verbal, el diseño en constante evolución, *Colaboración cercana* y la vinculación de las unidades, de largo como de corto plazo, de todos los involucrados.

XP valora la comunicación, la retroalimentación, la simplicidad y el correr riesgos. Los diferentes roles en el enfoque XP incluyen al *customer*, desarrolladore, rastreador y entrenador. Prescribe varias prácticas de negocios, codificación y desarrollo, así como eventos y artefactos para lograr un desarrollo eficaz y eficiente. XP ha sido adoptado ampliamente debido a sus prácticas de ingeniería bien definidas.

A.4.3 *Crystal Methods*

Las metodologías de desarrollo de software *Crystal* fueron presentadas por Alistair Cockburn a principios de 1990. Los métodos Crystal se centran en las personas, son ligeros y fáciles de adaptar. Porque la gente es lo primordial, los procesos y las herramientas de desarrollo no son fijos sino que se ajustan a las necesidades y características específicas del *project*. El espectro de color se utiliza para decidir sobre la variante de un proyecto. Los factores tales como la comodidad, el dinero discrecional, el dinero esencial, y la vida juegan un papel vital en la determinación del "peso" de la metodología, que se representa en varios colores del espectro. Crystal se divide en *Crystal Clear*, *Crystal Yellow*, *Crystal Orange*, *Crystal Orange Web*, *Crystal Red*, *Crystal Maroon*, *Crystal Diamond* y *Crystal Sapphire*.

Todos los métodos de Crystal tienen cuatro roles – patrocinador, diseñador principal, desarrolladores y usuario experto. Los métodos Crystal recomiendan diversas estrategias y técnicas para lograr agilidad. Un ciclo de proyectos Crystal consta de gráficos, ciclo de entrega y de recapitulación.

A.4.4 *Dynamic Systems Development Methods (DSDM)*

El marco *Dynamic Systems Development Methods (DSDM)* se publicó inicialmente en 1995 y es administrado por el Consorcio DSDM. DSDM establece la calidad y el esfuerzo en términos de costo y el tiempo desde el principio y ajusta los entregables del proyecto para cumplir con los criterios establecidos, dando prioridad a las prestaciones en las siguientes categorías: lo que "deben tener", "deberían tener", "podrían tener", y "no tendrán" (mediante la técnica *MoSCoW Prioritization*). DSDM es un método orientado al sistema con seis distintas fases de pre-proyecto; Viabilidad; Fundamentos; Exploración e Ingeniería; Despliegue y Evaluación de Beneficios.

Una versión posterior de DSDM conocida como DSDM Atern, presentada en el 2007, se centra tanto en la *priorization* de los entregables, como en usuarios consistentes o *customer Colaboración*. La nueva versión está inspirada por un Artic Tern, lo que es un marco de desarrollo de software centrado en el desarrollador para la entrega a tiempo y en presupuesto de las características del *project* relacionadas con el control de calidad y valor para el usuario.

A.4.5 Feature Driven Development (FDD)

Feature Driven Development (FDD) fue presentado por Jeff De Luca en 1997 y opera bajo el principio de la realización de un proyecto donde éste se separa en pequeñas funciones valoradas por el cliente que pueden ser entregadas en menos de dos semanas. FDD tiene dos principios - el desarrollo de software es una actividad humana y el desarrollo de software es una funcionalidad valorada por el cliente.

FDD define seis roles principales - Gerente de Obras, Arquitecto Principal, Gerente de Desarrollo, Programadores Principales, Dueños de clase, y expertos en el dominio con un número de papeles secundarios. El proceso de FDD es iterativo y consiste en el desarrollo de un modelo general, la construcción de una lista de características, la planificación, el diseño y la construcción por característica.

A.4.6 Test Driven Development (TDD)

También conocido como *Test-First Development*, *Test Driven Development* fue presentado por Kent Beck, uno de los creadores de *Extreme Programming (XP)*. *Test Driven Development* es un método de desarrollo de software que consiste en escribir primero un código de prueba automatizado y en el desarrollo de la menor cantidad de códigos necesarios para luego pasar la prueba. El *project* se divide en características pequeñas de valor para el cliente que deben ser desarrolladas en el ciclo de desarrollo más corto posible. Las pruebas se escriben basadas en los requisitos y especificaciones de los clientes. Las pruebas diseñadas en la fase precedente se utilizan para diseñar y escribir el código de producción.

TDD se puede clasificar en dos niveles: *Acceptance TDD (ATDD)* que requiere una prueba de aceptación específica y *Developer TDD (DTDD)* que tiene que ver con escribir sólo una prueba de desarrollador. TDD se ha vuelto popular debido a las numerosas ventajas que ofrece, tales como resultados rápidos y fiables, la retroalimentación constante y la reducción del tiempo de depuración.

A.4.7 Adaptive Software Development (ASD)

Adaptive Software Development (ASD) surgió a partir de la rápida labor de desarrollo de aplicaciones por Jim Highsmith y Sam Bayer. Los aspectos más destacados de los ASD son *adaptation* constantes de los procesos de trabajo, el suministro de soluciones a los problemas que surgen en los grandes *projects*, y el desarrollo incremental iterativo con prototipos continuos.

Al ser un enfoque de desarrollo impulsado por el riesgo y tolerante de los cambios, ASD indica que un plan no puede admitir las incertidumbres y *risks*, ya que eso sería indicativo de un plan deficiente. ASD se basa en las funciones y es impulsado por los objetivos. La primera fase del desarrollo de ASD es Especular (a diferencia de Planificar), seguido por las fases Colaborar y Aprender.

A.4.8 Agile Unified Process (AUP)

Agile Unified Process (AUP) evolucionó del proceso llamdo *Rational Unified Process* de IBM. Desarrollado por Scott Ambler, AUP combina técnicas ágiles de la industria ya probadas como *Test Driven Development (TDD)*, *Agile Modeling*, gestión del cambio ágil y la base de datos *Refactoring* para ofrecer un *product* de trabajo de la mejor calidad.

AUP modela sus procesos y técnicas basado en los valores de Simplicidad, Agilidad, Personalización, *Auto-organización*, Independencia de las herramientas, y se centra en actividades de alto valor. Los principios y valores AUP se ponen en acción en las fases de Inicio, Elaboración, Construcción y Transición.

A.4.9 Domain-Driven Design (DDD)

Domain-Driven Design se trata de un enfoque de desarrollo ágil con la intención de manejar diseños complejos con aplicación vinculada a un modelo en evolución. Fue concebido por Eric Evans en el año 2004 y gira en torno al diseño de un dominio básico. "Dominio" se define como un área de actividad a la que el usuario aplica un programa o funcionalidad. Muchas de estas áreas se procesan por lotes y un modelo es diseñado. El modelo consiste de un sistema de abstracciones que se pueden utilizar para diseñar el proyecto general y resolver los problemas relacionados con los dominios loteados. Los valores centrales de DDD incluyen el diseño orientado al dominio basado en modelos, lenguaje ubicuo y un contexto limitado.

En DDD, un idioma ubicuo es establecido y modelado. Luego sigue el diseño, desarrollo y las pruebas de seguimiento. La refinación y *refactoring* del modelo de dominio se realizan hasta que sea satisfactorio.

APÉNDICE B. AUTORES Y REVISORES DE LA GUÍA SBOK™

Este apéndice lista los nombres de aquellas personas que han contribuido al desarrollo y la producción de la *Guía SBOK™*.

SCRUMstudy™ le agradece a todas estas personas por su apoyo continuo y reconoce sus contribuciones para el desarrollo de la *Guía de SBOK™*.

B.1 Autor Principal

Tridibesh Satpathy

B.2 Los Co-autores y Expertos en la Materia

R-A Alves
Winfried Hackmann
Quincy D. Jordan
Gaynell Malone
J. Drew Nations
Buddy Peacock
Karen Lyncock
Jaime M. Rush
Elizabeth Lynne Warren
Ruth Kim
Mehul Doshi
Gaurav Garg
Ajey Grandhem
Sayan Guha
Vinay Jagannath
Deepak Ramaswamy
Ahmed Touseefullah Siddiqui

B.3 Revisores y Equipo de Edición

Corey T. Bailey
Sohini Banerjee
Vince Belanger
Bobbie Green
Magaline D. Harvey
Ravneet Kaur
Robert Lamb
Mimi LaRaque
Melissa Lauro
Richard Mather
Lachlan McGurk
Madhuresh Kumar Mishra
Neha Mishra
Yogaraj Mudalgi
Jose Nunez
Obi Nwaojigba
Bryan Lee Perez
James Pruitt
Charles J. Quansah
Frank Quinteros
Nadra Rafee
Tommie L. Sherrill
Barbara Siefken
Sandra A. Strech
Frances Mary Jo Tessler
Chrys Thorsen
Mike Tomaszewski
Ron Villmow

REFERENCIAS

Anderson, D., Augustine, S., Avery, C., Cockburn, A., Cohn, M., DeCarlo, D., Fitzgerald, D., Highsmith, J., Jepsen, O., Lindstrom, L., Little, T., McDonald, K., Pixton, P., Smith, P., and Wysocki, R. (2005) "Declaration of Interdependence," accessed Septiembre 2013, <http://www.pmdoi.org/>.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. (2001) "Manifesto for Agile Software Development," obtenido el Septiembre 2013, <http://agilemanifesto.org/>.

Fellers, G. (1994) *Why Things Go Wrong: Deming Philosophy In A Dozen Ten-Minute Sessions*. Gretna, LA: Pelican Publishing.

Greenleaf, R. K. (1977) *Servant Leadership: A Journey into the Nature of Legitimate Power and Greatness*. Mahwah, NJ: Paulist Press.

Kano, N., Seraku, N., Takahashi, F., and Tsuji, S. (1984) "Attractive Quality and Must Be Quality." *Quality*, 14 (2): 39–48.

Leffingwell, D. and Widrig, D. (2003) *Managing Software Requirements: A Use Case Approach*, 2nd ed. Boston: Addison-Wesley.

Maslow, A. H. (1943) "A Theory of Human Motivation." *Psychological Review*, 50 (4): 370–396.

McGregor, D. (1960) *The Human Side of Enterprise*. New York: McGraw-Hill.

Patton, J. (2005) "It's All in How You Slice." *Better Software*, Enero: 16–40.

Spears, L. C. (2010) "Character and Servant Leadership: Ten Characteristics of Effective, Caring Leaders." *The Journal of Virtues & Leadership*, 1 (1): 25–30.

Takeuchi, H. and Nonaka, I. (1986) "The New New Product Development Game." *Harvard Business Review*, January–February: 137–146.

GLOSARIO

100-Point Method

100-Point Method fue desarrollado por Dean Leffingwell y Don Widrig (2003). Se trata de darle al *customer* 100 puntos que puedan usar para votar por las características que consideren más importantes.

Accepted Deliverables

Las entregas que cumplen con los *User Stories Acceptance Criteria* son aceptadas por el *Propietario del producto*. Estas son consideradas *Accepted Deliverables* que se le pueden entregar al *customer* si así lo desea.

Adaptation

Adaptation sucede cuando el Equipo Principal de Scrum (*Scrum Core Team*) y el/los *stakeholder(s)* aprende(n) a través de la transparencia e *inspection* y luego se adaptan a lo aprendido para mejorar el trabajo.

Affinity Estimation

Affinity Estimation es una técnica utilizada para estimar rápidamente un gran número de *User Stories* utilizando categorías. Las categorías pueden ser pequeñas, medianas o grandes, o pueden ser numeradas usando los valores de punto de la historia para indicar el tamaño relativo. Algunos de los beneficios claves de este enfoque es que el proceso es muy transparente, visible para todos, y es fácil de llevar a cabo.

Agreed Actionable Improvements

Agreed Actionable Improvements son los resultados primarios del proceso *Retrospectiva de Sprint*. Es la lista de elementos configurables que el equipo ha logrado hacer frente a los problemas y así mejorar los procesos con el fin de mejorar su desempeño en futuros *Sprints*.

Aprobar, estimar y asignar historias de usuarios

En este proceso el *Propietario del producto* aprueba los *User Stories* para un *Sprint*. Luego, el *Scrum Master* y *Equipo Scrum* estiman el esfuerzo necesario para desarrollar la funcionalidad descrita en cada *User Story*. Por último, el *Equipo Scrum* se compromete a entregar los requisitos explícitos por el *customer* en forma de *Approved, Estimated and Committed User Stories*.

Approved Change Requests

Approved Change Requests son los cambios que han sido aprobados para su inclusión en el *Prioritized Product Backlog*. A veces, *Approved Change Requests* pueden originar de los gerentes del *program o portfolio* y serían entradas que se añadirán a la lista de cambios de *project* aprobados para su ejecución en futuros *Sprints*.

Approved, Estimated, and Committed User Stories

Los *User Stories*, que son un aporte a este proceso, tienen estimaciones de alto nivel de los procesos *Creación de la lista priorizada de pendientes del producto* y *Elaborar historias de usuario*. Estas estimaciones son utilizadas por el *Propietario del producto* para aprobar *User Stories* para el *Sprint*. Una vez aprobados, los *User Stories* se estiman por el equipo utilizando diversas técnicas de evaluación. Despues de dicha evaluación, el equipo se compromete en un subconjunto de *user stories* aprobados y calculados que creen que pueden completar en el próximo *Sprint*. Estos *User Stories* son *Approved, Estimated y Committed User Stories*, que se convertirán en parte del *Sprint Backlog*.

Assertive Leader

Assertive Leaders confrontan los *issues* y demuestran confianza para establecer autoridad con respeto.

Assigned Action Items and Due Dates

Una vez que *Assigned Action Items and Due Dates* se han elaborado y refinado, los puntos de acción para aplicar las mejoras pueden ser considerados por el *Equipo Scrum*. Cada elemento de acción tendrá una fecha de entrega definida para su conclusión.

Autocratic Leader

Autocratic leaders toman decisiones por su cuenta, dándoles poco o nada de tiempo a los miembros del equipo antes de tomar una decisión. Este estilo de liderazgo se debe utilizar solamente en raras ocasiones.

Automated Software Tools

Automated Software Tools son herramientas de software utilizadas para la planificación, la recopilación de información y la distribución.

Better Team Coordination

Facilita la coordinación de trabajo entre varios *Equipos Scrum*. Esto es especialmente importante cuando hay tareas que impliquen dependencias entre equipos. Esto expone rápidamente incompatibilidades y discrepancias entre el trabajo y los resultados de los diferentes equipos. Este foro también les da a los equipos la oportunidad de mostrar sus logros y dar retroalimentación a los otros equipos.

Brainstorming

Son sesiones donde los *stakeholder* y los miembros del Equipo Principal de Scrum (*Scrum Core Team*) comparten abiertamente ideas a través de discusiones y sesiones de intercambio de conocimientos, las cuales normalmente se llevan a cabo por un facilitador.

Justificación del negocio

Justificación del negocio demuestra las razones para emprender un *project*. Responde a la pregunta "¿Por qué es necesario este *project*? *Justificación del negocio* impulsa todas las decisiones relacionadas con un *project*.

Business Needs

Business needs son los resultados de negocio que se espera que el *project* cumpla, tal como se documenta en el *Prioritized Product Backlog*.

Business Requirements

Business Requirements definen lo que debe ser entregado para cumplir con *business needs* y proporcionarles valor a los *socios*. La suma de todos los conocimientos adquiridos a través de diversas herramientas como las entrevistas al usuario o *customer*, los cuestionarios, JAD Sessions, Gap Analysis, y SWOT Analysis entre otros, ayudan a tener una mejor perspectiva sobre los *business requirements* y en la creación del *Prioritized Product Backlog*.

Change Request(s)

Las solicitudes de cambio se presentan por lo general como *Change Requests*. *Change Requests* se consideran que no están aprobados, hasta que estén formalmente aprobados.

Jefe Propietario del producto

En el caso de los grandes proyectos, el *Jefe Propietario del producto* prepara y mantiene el *Prioritized Product Backlog* para el proyecto. Él o ella coordina el trabajo entre los *Propietario del producto* de los *Equipos Scrum*. Los *Propietario del producto*, a su vez, gestionan sus respectivas partes del *Prioritized Product Backlog*.

Jefe Scrum Master

En el caso de los grandes *projectos*, el *Jefe Scrum Master* es responsable de moderar el *Scrum of Scrums* (SoS) Meeting y la eliminación de *impediments* que afectan a varios equipos.

Coaching/Supportive Leader

Coaching y Supportive Leaders dan instrucciones y luego apoyan y supervisan a los miembros del equipo al escuchar, ayudar, alentar, y presentar una perspectiva positiva en tiempos de incertidumbre.

Colaboración

Colaboración en Scrum se refiere al trabajo e interconexión entre el Equipo Principal de Scrum (*Scrum Core Team*) y los *socios* para crear y validar los resultados del *project* y cumplir con los objetivos planteados en el *Project Vision*. *Colaboración* se produce cuando los equipos trabajan en conjunto para aprender de los demás y aprovechar este conocimiento para luego producir algo más grande.

Colaboración Plan

Colaboración es un elemento muy importante en Scrum. *Colaboración Plan* describe cómo aquellos que toman decisiones, los *socios* y miembros del equipo participan y colaboran entre sí.

Colocation

Colocation es cuando los miembros del equipo Scrum están ubicados en el mismo lugar de trabajo para así aprovechar sus ventajas de una mejor coordinación, resolución de problemas, intercambio de conocimientos y aprendizaje.

Communication Plan

Este plan especifica los registros que deben ser creados y mantenidos durante todo el *project*. Una variedad de métodos se utiliza para transmitir información importante del *project* a los *socios*. *Communication Plan* define estos métodos, así como quién es responsable de las diversas actividades de comunicación.

Company Mission

Company Mission ofrece un marco para la formulación de las estrategias de una empresa u organización que orienta la toma de decisiones en general.

Company Vision

El comprender la visión de la empresa ayuda a que el *project* mantenga su enfoque en los objetivos de la organización y en el potencial de la empresa. El *Propietario del producto* puede usar el *Company Vision* para crear el *Reunión de la Visión del Proyecto*.

Llevar a cabo el Standup diario

Llevar a cabo el Standup diario es un proceso en el que una reunión altamente concentrada y *Time-boxed* se lleva a cabo todos los días. Esta reunión se conoce como un *Daily Standup Meeting*, que es un foro para

que los miembros del *Equipo Scrum* se actualicen el uno al otro sobre sus progresos y cualquier *impediments* que puedan enfrentar.

Realizar el plan de lanzamiento

En este proceso, el Equipo Principal de Scrum (*Scrum Core Team*) revisa los *User Stories* de alto nivel en el *Prioritized Product Backlog* para desarrollar un *Release Planning Schedule*, que es esencialmente un programa de implementación por fases que se puede compartir con el/los *Stakeholder(s)*. El *Length of Sprint* también se determina en este proceso.

Conflict Management

Las técnicas de *Conflict Management* son utilizadas por los miembros del equipo para gestionar los conflictos que surgen durante un *project Scrum*. Las fuentes de conflicto incluyen a menudo los horarios, las prioridades, los recursos, la jerarquía de informes, *issues* técnicos, procedimientos, las diferentes personalidades y los costos.

Continuous Improvement

Continuous Improvement es un enfoque de Scrum en el que el equipo aprende de las experiencias y el compromiso de los socios para mantener al día al *Prioritized Product Backlog* con los cambios de los requisitos.

Continuous Value Justification

Continuous Value Justification se refiere a la evaluación con regularidad del valor de negocio para determinar si la justificación o la viabilidad de la ejecución del *project* sigue existiendo.

Convocar Scrum de Scrums

En este proceso el/los *Scrum Master(s)* o los representantes del Equipo Scrum convocan *Scrum of Scrum Meetings* en intervalos predeterminados, o cuando sea necesario, para colaborar y realizar un seguimiento de su respectivos progresos, *impediments*, y dependencias entre equipos.

Core Role(s)

Core Roles son los papeles que obligatoriamente se requieren para producir el *product* del *project*, están comprometidos con el *project*, y en última instancia son los responsables del éxito de cada *Sprint* dentro del *project* y del *project* en su totalidad.

Crear entregables

Crear entregables es el proceso en el que el *Equipo Scrum* trabaja en las tareas del *Sprint Backlog* para crear *Sprint Deliverables*.

Creación de la lista priorizada de pendientes del producto

En este proceso se refinan y elaboran los *Epic(s)* y luego se priorizan para crear un *Prioritized Product Backlog* para el *project*. Los *Done Criteria* también se establecen en este punto.

Crear la visión del proyecto

En este proceso, el *Caso de Negocio del Proyecto* es revisado para crear un *Reunión de la Visión del Proyecto* que servirá de inspiración y proporcionará un enfoque de todo el *project*. El *Propietario del producto* se identifica en este proceso.

Elaboración de la lista de pendientes del Sprint

En este proceso, el Equipo Principal de Scrum lleva a cabo un *Sprint Planning Meetings* donde el grupo crea un *Sprint Backlog* que contiene todas las tareas que deben completarse en el *Sprint*.

Elaboración de tareas

En este proceso, los *Approved, Estimated, and Committed User Stories* se dividen en tareas específicas y se compilan en un *Task List*. A menudo, un *Task Planning Meeting* se convocará para llevarlo a cabo.

Elaborar historias de usuario

En este proceso, los *User Stories* y sus *User Story Acceptance Criteria* son creados. *User Stories* son generalmente escritos por el *Propietario del producto* y están diseñados para asegurar que los requisitos del *customer* estén claramente representados y puedan ser plenamente comprendidos por todos los *socios*.

Cumulative Flow Diagram (CFD)

Cumulative Flow Diagram (CFD) es una herramienta útil para la elaboración de informes y el seguimiento de los resultados del proyecto. Proporciona una representación sencilla y visual del progreso del proyecto en un punto de tiempo determinado. Se utiliza generalmente para proporcionar un estado de mayor nivel de la totalidad del proyecto y no para actualizaciones diarias de *Sprints* individuales.

Customer

El *Customer* es un individuo o la organización que adquiere el *product* del *project*, servicio, o cualquier otro resultado. Para cualquier organización, dependiendo del *project*, no puede haber dos *customers* internos (es decir, dentro de la misma organización) o *customers* externos (es decir, fuera de la organización).

Customer Value-based Prioritization

Customer Value-based Prioritization le da importancia primordial al *customer* y se esfuerza primero por poner en práctica *User Stories* con el valor más alto. Tales *User Stories* con alto valor se identifican y se pasan a la parte superior del *Prioritized Product Backlog*.

Daily Standup Meeting

Daily Standup Meeting es una reunión diaria de corta duración, *Time-boxed* a 15 minutos. Los miembros del equipo se reúnen para informar de sus progresos al contestar las siguientes tres preguntas:

1. ¿Qué terminé ayer?
2. ¿Qué terminaré hoy?
3. ¿Qué *impediments* u obstáculos estoy enfrentando en la actualidad?

Decomposition

Decomposition es una herramienta donde las tareas de alto nivel se dividen en niveles inferiores y detallados. Los *User Stories* se dividen en tareas por los miembros del *Equipo Scrum*. Los *Prioritized Product Backlog User Stories* deben estar suficientemente fragmentados a un nivel que le proporcione el *Equipo Scrum* información adecuada para formular *Crear entregables* de las tareas mencionadas en el *Task List*.

Delegating Leader

Delegating Leaders están involucrados en la mayoría de la toma de decisiones; sin embargo, delegan parte de planificación y de las responsabilidades de toma de decisión a los miembros del equipo, sobre todo si son competentes para manejar tareas. Este estilo de liderazgo es apropiado en situaciones en las que el líder está en sintonía con los detalles de proyectos específicos y cuando el tiempo es limitado.

Demostración y validación del Sprint

En este proceso, el *Equipo Scrum* les demuestra los *Sprint Deliverables* al *Propietario del producto* y a los *socios* relevantes en un *Sprint Review Meeting*.

Dependency Determination

Una vez que el *Equipo Scrum* ha seleccionado *User Stories* para un determinado *Sprint*, deberían entonces considerar las dependencias, incluyendo las relacionadas con la disponibilidad de las personas, así como las dependencias técnicas. El documentar adecuadamente las dependencias ayuda a los *Equipos Scrum* a determinar el orden relativo en el que las tareas deben ejecutarse para crear *Sprint Deliverables*. Las

dependencias también destacan la relación y la interacción entre las tareas tanto en el *Equipo Scrum* que trabaja en un determinado *Sprint* y con las de otros *Equipos Scrum* en el *project*.

Design Patterns

Design Patterns proporcionan una manera formal de registrar una resolución de un problema de diseño en un campo específico de especialización. Estos patrones registran tanto el proceso que se utiliza y la resolución, lo cual luego puede ser reusado para mejorar la toma de decisiones y la productividad.

Desarrollo de épica(s)

En este proceso, el *Reunión de la Visión del Proyecto* sirve como la base para el desarrollo de *Epics*. *User Group Meetings* se pueden llevar a cabo para la formación de *Desarrollo de épica(s)*.

Development in Phases Contract

El contrato permite que los fondos estén disponibles cada mes o cada trimestre después de que un *release* se ha completado con éxito. Se incentiva al *customer* y proveedor y se asegura de que el riesgo monetario del *customer* se limite a ese período de tiempo determinado, ya que los lanzamientos fracasados no son financiados.

Directing Leader

Directing Leaders instruye a los miembros del equipo con respecto a las tareas que se requieren y sobre cuándo y cómo deben llevarse a cabo.

Discretionary Dependencies

Discretionary Dependencies son dependencias que se colocan en el flujo de trabajo por decisión propia. Normalmente, *discretionary dependencies* son determinados por el *Equipo Scrum*, basado en las experiencias o las mejores prácticas en un campo o dominio en particular.

Done Criteria

Done Criteria es un conjunto de reglas que se aplican a todos los *User Stories*. Una definición clara de *Done* es crítica, ya que elimina la ambigüedad de los requisitos y ayuda a que el equipo se adhiera a las normas de calidad obligatorias. Esta clara definición se utiliza para crear los *Done criteria*, que son un resultado del proceso de *Creación de la lista priorizada de pendientes del producto*. Un *User Story* se considera hecho cuando es aprobado por el *Propietario del producto* quien lo juzga basado en el *Done Criteria* y los *User Story Acceptance Criteria*.

Earned Value Analysis

Earned Value Analysis analiza, en un punto dado, el verdadero desempeño del proyecto con respecto al rendimiento previsto. Mide las variaciones actuales de tiempo y costo del rendimiento del *project* y, basado en este rendimiento actual, prevé el costo final.

Effort Estimated Task List

Effort Estimated Task List es una lista de las tareas asociadas con los *User Stories* incluidos en un *Sprint*. El esfuerzo estimado se expresa en términos de *estimation criteria* acordados por el equipo. El *Effort Estimated Task List* es usado por el *Equipo Scrum* durante el *Sprint Planning Meetings* para crear el *Sprint Backlog* y el *Sprint Burndown Chart*.

Control del proceso empírico

El modelo *Control del proceso empírico* ayuda a tomar decisiones basadas en la observación y la experimentación, más que en la planificación inicial detallada. Se basa en las tres ideas principales de *Transparencia, inspection y adaptation*.

Epic(s)

Epic(s) se escribe(n) en las etapas iniciales del *project*, cuando la mayoría de los *User Stories* son las funcionalidades de alto nivel o descripciones de *product* y los requisitos de los *products* están ampliamente definidos. Son *User Stories* grandes sin refinar en el *Prioritized Product Backlog*.

Estimate Range

Las estimaciones para los *projects* deben ser presentadas en rangos. Las cifras exactas pueden dar la impresión de ser muy específicas cuando en realidad no lo pueden ser. De hecho, las estimaciones, por definición, se entiende que no son precisamente exactas. *Estimate ranges* deben basarse en el nivel de confianza que el equipo tiene en cada estimación.

Estimar tareas process

En este proceso, el Equipo Principal de Scrum, en un *Task Estimation Workshop*, estima el esfuerzo necesario para realizar cada tarea en el *Task List*. El resultado de este proceso es un *Effort Estimated Task List*.

Estimation Criteria

El objetivo principal de utilizar *Estimation Criteria* es mantener los tamaños de estimación relativos y minimizar la necesidad de re-estimación. *Estimation Criteria* se pueden expresar de muchas maneras. Dos ejemplos comunes son los *story points* y el tiempo ideal.

Expected Monetary Value

Se trata de una técnica de *risk assessment* en el que el impacto financiero potencial de un riesgo se determina sobre la base de su *Expected Monetary Value (EMV)*. EMV se calcula multiplicando aproximadamente el impacto monetario por la probabilidad del riesgo, según el *customer*.

Explorer—Shopper—Vacationer—Prisoner (ESVP)

Este es un ejercicio que se puede realizar al inicio del *Retrospectiva de Sprint Meeting* para entender la mentalidad de los participantes y establecer el tono de la reunión. Se les pide a los asistentes que indiquen de forma anónima lo que mejor representa su punto de vista en la reunión.

External dependencies

External Dependencies son las dependencias relacionadas con las tareas, actividades o *products* que están fuera del alcance del trabajo a ser ejecutado por el *Equipo Scrum*, pero son necesarias para completar una tarea de *project* o crear un entregable de *project*. *External Dependencies* están por lo general fuera del control del *Equipo Scrum*.

Fist of Five

Fist of Five es un mecanismo simple y rápido que estimula el debate y que ayuda a llegar a un consenso en un grupo. Tras el debate inicial sobre una determinada propuesta o decisión pendiente, a los miembros del *Equipo Scrum* se les pide que voten en una escala de 1 a 5 usando sus dedos.

Focus Group Meetings

Focus groups reúnen personas en una sesión guiada para proporcionar sus opiniones, percepciones o valoraciones de un *product*, servicio o resultado deseado. Los miembros del grupo de enfoque tienen la libertad de hacer preguntas el uno al otro para obtener aclaraciones sobre temas o conceptos específicos. A través de cuestionamiento, la crítica constructiva y la retroalimentación, los grupos de enfoque conducen a un *product* de mejor calidad y con ello contribuyen a la satisfacción de las expectativas de los usuarios.

Formación de un equipo Scrum

Los miembros del *Equipo Scrum* se identifican durante este proceso. Normalmente, el *Propietario del producto* es el responsable principal de la selección de los miembros del equipo, pero él o ella lo hace a menudo en *Colaboración* con el *Scrum Master*.

Forming Stage

Forming Stage es la primera etapa de la formación del equipo, a menudo considerado un escenario divertido porque todo es nuevo y el equipo aún no ha encontrado alguna dificultad con el *project*.

Four Questions per Team

Esto es un conjunto de preguntas formuladas en cada *Scrum of Scrums (SoS) Meeting*. Cada representante del *Equipo Scrum* proporcionará actualizaciones de su equipo que usualmente se proporcionan en forma de respuestas a cuatro preguntas específicas.

1. ¿En qué ha trabajado mi equipo desde la última reunión?
2. ¿Qué va a hacer mi equipo hasta la próxima reunión?
3. ¿Con qué contaban otros equipos que hiciera nuestro equipo que no se ha hecho?
4. ¿Qué planifica hacer nuestro equipo que podría afectar a otros equipos?

Gap Analysis

Gap Analysis es una técnica que se utiliza para comparar el verdadero estado actual, con algún estado deseado y para determinar la forma de acortar la distancia entre ellos.

Mantenimiento de la lista priorizada de pendientes del producto

Mantenimiento de la lista priorizada de pendientes del producto es un proceso en el que el *Prioritized Product Backlog* se actualiza y se mantiene continuamente.

Identificar al Scrum Master y al socio(s) process

En este proceso, el *Scrum Master* y los socios se identifican utilizando criterios de selección específicos.

Impediment

Un *impediment* es cualquier obstáculo o barrera que reduce la productividad del *Equipo Scrum*.

Implement Phase

Implement Phase incluye los procesos relacionados con la ejecución de las tareas y actividades para crear el producto de un proyecto.

Incentive and Penalty Contract

Este contrato se basa en el acuerdo por el cual el proveedor será recompensado con un incentivo financiero, si los *products* del *project* se entregan a tiempo, pero incurrirá sanciones económicas si la entrega está tarde.

Incremental Delivery Contract

Este contrato incluye puntos de inspección en intervalos regulares. Ayuda a que el *customer* o los *socios* tomen decisiones sobre el desarrollo de *products* periódicamente a lo largo del *project* en cada *inspection point*. El *customer* puede aceptar el desarrollo del producto, decidir sobre el desarrollo del *product*, o solicitar modificaciones de los *products*.

Index Cards

Index cards, a menudo descrito como *Story Cards* (*Tarjetas de Historia*), se utilizan para realizar un seguimiento de los *User Stories* en todo el *project*. Esto aumenta la visibilidad y la transparencia y facilita la detección temprana de cualquier problema que pueda surgir.

Initiate phase

Esta fase se compone de los procesos relacionados con la iniciación de un *project*: *Create Project Vision*, *Identificar al Scrum Master y al socio(s)*, *Formación de un equipo Scrum*, *Desarrollo de épica(s)*, *Creación de la lista priorizada de pendientes del producto* y *Realizar el plan de lanzamiento*.

Inspection

Inspection se refiere a la vigilancia necesaria para seguir *Control del proceso empírico*, para asegurar que los *products* entregables del *project* se ajusten a los requisitos.

Internal Dependencies

Internal Dependencies son las dependencias entre las tareas, *products* o actividades que están bajo el control del *Equipo Scrum* y dentro del alcance de trabajo a ser ejecutado por el *Equipo Scrum*.

Internal Rate of Return (IRR)

Internal Rate of Retrun (IRR) es un tipo de descuento de una inversión en la que el valor presente de los flujos de efectivo se hace igual al valor presente de los flujos de salida de efectivo para evaluar la tasa de rentabilidad de un *project*. Al comparar *projects*, uno con un IRR mayor es típicamente mejor.

Issues

Issues son generalmente certezas bien definidas que actualmente se están produciendo en el *project*, así que no hay necesidad de realizar una evaluación de la probabilidad como lo haríamos para un riesgo.

Desarrollo iterativo

Desarrollo iterativo es la entrega gradual de valor al *customer*.

JAD Sessions

La sesión *Joint Application Design (JAD)* es una técnica de recopilación de requisitos. Se trata de un taller facilitado altamente estructurado que acelera el proceso llamado *Crear la visión del proyecto*, ya que le(s) permite al/a los *stakeholder(s)* y a otros que toman decisiones llegar a un consenso sobre el alcance, los objetivos, y otras especificaciones del *project*.

Joint Venture Contract

Este contrato se utiliza generalmente cuando dos o más socios se unen para llevar a cabo el trabajo de un *project*. Las partes involucradas en el *project* recibirán algún *Return on Investment* porque los ingresos o beneficios generados serán compartidos entre las partes.

Kano Analysis

Kano Analysis fue desarrollado por Noriaki Kano (1984) y consiste en clasificar las características o requisitos en cuatro categorías basado en las preferencias del *customer*:

1. *Exciters/Delighters*
2. *Satisfiers*
3. *Dissatisfiers*
4. *Indifferent*

Laissez Faire Leader

Un estilo de liderazgo en el que el equipo se queda en gran parte sin supervisión, y el líder no interfiere con las actividades laborales diarias. Esto a menudo conduce a un estado de anarquía.

Length of Sprint

Basado en las diversas entradas (*inputs*) que incluyen *Business requirements* y el *Release Planning*, el *Propietario del producto* y el *Equipo Scrum* deciden sobre la longitud de los *Sprints* para el proyecto. Una vez determinada, la longitud del *Sprint* suele ser fija por todo el proyecto.

Length of Sprint es la duración de los *Sprints* determinados para un proyecto.

Risks

Risks incluyen eventos inciertos o no planificados que pueden afectar al *project* de manera positiva o negativa.

Mandatory Dependencies

Estas dependencias son inherentes a la naturaleza del trabajo, como una limitación física, o pueden ser debido a las obligaciones contractuales o a requisitos legales.

Market Study

Se refiere a la investigación organizada, la recopilación, el cotejo y análisis de datos relacionados con las preferencias de los *customers* en relación a los *products*. A menudo incluye numerosos datos sobre las tendencias del mercado, la segmentación del mercado y los procesos de comercialización.

Minimum Acceptance Criteria

Minimum Acceptance Criteria son declarados por la unidad de negocio. Luego se convierten en parte de los *Acceptance Criteria* para cualquier *User Story* para esa unidad de negocio. Cualquier funcionalidad definida por la unidad de negocio debe satisfacer estos *Minimum Acceptance Criteria*, si ha de ser aceptada por el respectivo *Propietario del producto*.

Mitigated Risks

Mitigated Risks se refiere a los *risks* que se tratan o mitigan por el *Equipo Scrum* durante el *project*.

Monopoly Money

Monopoly Money es una técnica que consiste en darle al *Customer* "monopoly money" o "dinero falso" igual a la cantidad del *project budget* y pedirle que lo distribuya entre los *User Stories* en consideración. De esta manera, el *customer* prioriza basado en lo que está dispuesto a pagar por cada *User Story*.

MoSCoW Prioritization

MoSCoW Prioritization es un esquema que deriva su nombre de las primeras letras de las frases "Must have", "Should have," "Could have," y "Won't have" ("debe tener", "debería tener", "podría tener", y "no tendrá"). Las etiquetas están en orden de prioridad descendiente. "Must Have" ("Debe tener") se le pone a las características a las cuales sin ellas, el producto no tendrá valor. "Will not have" ("No tendrá") se les otorga a las características que, a pesar de que sería bueno tener, no son necesarias para ser incluido.

Net Present Value (NPV)

Net Present Value (NPV) es un método utilizado para determinar el valor neto actual de un futuro beneficio económico, dada una inflación prevista o tasa de interés.

Non-core role

Non-core roles son aquellos papeles que no son obligatoriamente necesarios para el proyecto Scrum. Pueden incluir miembros del equipo que están interesados en el *project*, pero que no tienen ninguna función oficial en el equipo del *project*. Pueden interactuar con el equipo, pero tal vez no sean responsables del éxito del *project*.

Norming stage

La tercera etapa de la formación del equipo es cuando el equipo comienza a madurar, resolver sus diferencias internas, y encontrar soluciones para trabajar juntos. Se considera un período de ajuste.

Number of Stories

Number of stories se refiere al número de *User Stories* que se entrega como parte de un sólo *Sprint*. Se puede expresar en términos de conteo simple, o conteo ponderado.

Opportunities

Risks que puedan tener un impacto positivo en el *project* se les conoce como *opportunities*.

Opportunity Cost

Opportunity cost se refiere al valor de la segunda opción de negocio o *project* que fue descartada en favor del *project* elegido.

Organizational Deployment Methods

Los mecanismos de despliegue de cada organización tienden a ser diferentes en función de la industria, los usuarios preferidos, y posicionamiento. Dependiendo del *product* que se entrega, el despliegue puede tener lugar de forma remota o puede implicar el envío físico o transición de un elemento.

Organizational Resource Matrix

Organizational Resource Matrix es una representación jerárquica de una combinación de una estructura de organización funcional y una estructura organizativa de *project*. Las organizaciones matriciales reúnen a los miembros de diferentes departamentos funcionales, tales como tecnología, finanzas, marketing, ventas, manufactura, y otros departamentos para cumplir un *project* - y crean así equipos multifuncionales.

Paired Comparison

Paired Comparison es una técnica donde se prepara una lista de todas los *User Stories* en el *Prioritized Product Backlog*. Luego, cada *User Story* se compara con los otros *User Stories* en la lista, uno a la vez. Cada vez que los *User Stories* se comparan, se toma una decisión en cuanto a cuál de los dos es más importante. A través de este proceso, una lista de prioridades de los *User Stories* se puede generar.

Pareto Analysis

Esta técnica de la evaluación del riesgo implica la clasificación de *risks* por magnitud. Ayuda al *Equipo Scrum* a organizar los *risks* en el orden de sus impactos probables sobre el *project*.

PDCA/PDSA cycle

Plan-Do-Check-Act Cycle—también conocido como el Deming o Shewhart Cycle—fue desarrollado por el Dr. W. Edwards Deming, considerado el padre de *Quality Control moderno*, y el Dr. Walter A. Shewhart. Deming luego modificó *Plan-Do-Check-Act* a *Plan-Do-Study-Act (PDSA)* porque sentía que el término "estudio", enfatiza el análisis en, lugar de enfatizar la idea des *Inspection*, como lo implica el término "Check". Tanto Scrum y el Ciclo Deming/Shewhart/PDCA son métodos iterativos que se centran en *continuous improvement*.

Performing stage

La etapa final de la formación de equipo cuando el equipo está más unido y opera a su nivel más alto en términos de rendimiento. Los miembros se han convertido en un equipo eficiente de profesionales que son consistentemente productivos.

Personas

Personas son personajes de ficción muy detallados, son representantes de la mayoría de los usuarios y de otros *socios* quienes pueden no utilizar directamente el producto final. Las *Personas* se crearon para identificar las necesidades de los usuarios.

Piloting Plan

Piloting Plan se puede utilizar para trazar una implementación piloto en detalle. El alcance y los objetivos del despliegue, la base de usuarios seleccionados para la implementación, un cronograma de implementación, los planes de transición, la preparación necesaria del usuario, los criterios de evaluación para el despliegue, y otros elementos claves relacionados con el despliegue se especifican en el *Pilot Plan* y se comparten con los *socios*.

Plan and Estimate phase

Plan and Estimate phase se compone de los procesos relacionados con la planificación y la estimación de las tareas, que incluyen *Elaborar historias de usuario; Aprobar, estimar y asignar historias de usuarios; Elaboración de tareas, Estimar tareas, y Elaboración de la lista de pendientes del Sprint.*

Planning for Value

Planning for Value se refiere a justificar y confirmar el valor del proyecto. La responsabilidad de determinar cómo se crea valor cae en los *socios* (patrocinadores, *customers* y/o los usuarios), mientras que el *Equipo Scrum* se concentra en lo que está por desarrollar.

Planning Poker

Planning Poker también llamado *Estimación Poker*, es una técnica de estimación que equilibra el pensamiento del grupo y el pensamiento individual para estimar los tamaños relativos de los *User Stories* o el esfuerzo necesario para desarrollarlos.

Points for Cost Estimating

La estimación del costo se puede lograr mediante el uso de unidades relativas (por ejemplo, las estimaciones de esfuerzo) en lugar de unidades absolutas (es decir, los costos reales incurridos). Con el fin de estimar el costo de implementar un *User Story*, el *Equipo Scrum* puede utilizar *story points*. Cuando se hace esto, el costo estimado para cada tarea será en forma de *story points*, en lugar de unidades monetarias.

Portfolio

Portfolio es un grupo de programas relacionados, con el objetivo de entregar resultados de negocio como se define en el *Portfolio Vision Statement*. El *Prioritized Portfolio Backlog* incorpora el *Prioritized Program Backlog* para todos los programas en el *portfolio*.

Portfolio Propietario del producto

El *Portfolio Propietario del producto* define los objetivos y las prioridades estratégicas para el *portfolio*.

Portfolio Scrum Master

El *Portfolio Scrum Master* resuelve los problemas, remueve *impediments*, facilita y lleva a cabo reuniones para el *portfolio*.

Prioritization

Prioritization se puede definir como la determinación del orden de las cosas y separar lo que se hará ahora, de lo que se puede hacer más tarde.

Prioritized Product Backlog

Prioritized Product Backlog es un documento de requisitos individuales que define el alcance del proyecto, proporcionando una lista de prioridades de las características del *product* o servicio a ser entregado por el proyecto.

Probability Impact Grid

Una red donde los *risks* se evalúan para establecer la probabilidad de ocurrencia y de impacto potencial en los objetivos del *project*. En general, una calificación numérica se asigna para tanto la probabilidad y el impacto. Los dos valores se multiplican luego para derivar una puntuación de gravedad de riesgo, que puede ser utilizado para priorizar *risks*.

Probability Trees

Eventos potenciales están representados en un diagrama con una rama para cada resultado posible de los acontecimientos. La probabilidad de cada resultado se indica en la rama apropiada, y estos valores se pueden utilizar para calcular el impacto general de la ocurrencia de riesgos en un *proyecto*.

Product

El término "producto" (*product*) en la *Guía SBOK™* puede referirse a un producto, servicio, o cualquier otra prestación que le proporciona valor al *customer*.

Prioritized Product Backlog Review Meeting

Un *Product Backlog Review Meeting* (también referido como *Prioritized Product Backlog Grooming Session*) es una reunión formal durante el proceso *Mantenimiento de la lista priorizada de pendientes del producto*, que ayuda al Equipo Scrum a obtener consenso sobre el *Prioritized Product Backlog*.

Propietario del producto

El *Propietario del producto* es la persona responsable de maximizar el valor del negocio para el *project*. Él o ella es responsable de articular los requisitos de los *customers* y de mantener *Justificación del negocio* para el *project*.

Program

Un programa es un grupo de proyectos relacionados con el objetivo de entregar resultados de negocio definidos en *Program Vision Statement*. El *Prioritized Program Backlog* incorpora el *Prioritized Product Backlogs* para todos los *projects* del *program*.

Program and Portfolio Risks

Risks relacionados con un *portfolio* o un *program* que también afectarán los proyectos que forman parte del respectivo *portfolio* o *program*.

Program Propietario del producto

El *Program Propietario del producto* define los objetivos estratégicos y las prioridades del *program*.

Program Scrum Master

El *Program Scrum Master* resuelve los problemas, remueve *impediments*, facilita, y lleva a cabo reuniones para el *program*.

Project

Un *project* es una empresa de colaboración para crear nuevos productos o servicios, o para obtener resultados definidos en *Reunión de la Visión del Proyecto*. Los *projects* son por lo general afectados por limitaciones de tiempo, costo, alcance, calidad, la gente y la capacidad de la organización.

Project Benefits

Project benefits incluye todas las mejoras cuantificables en un *product*, servicio o resultado que se preste a través de la finalización con éxito de un *project*.

Project Budget

Project budget es un documento financiero que incluye el costo de las personas, materiales y otros gastos relacionados en un *project*. El *project budget* típicamente se firma por el/los patrocinador(es) para asegurar que haya suficientes fondos disponibles.

Project Charter

Un *Project Charter* es una declaración oficial de los objetivos y resultados deseados del *project*. En muchas organizaciones, el *project charter* es el documento que autoriza oficial y formalmente el *project*, otorgandole al equipo por escrito el derecho para comenzar el trabajo de *project*.

Project Costs

Project costs es la inversión y otros costos de desarrollo de un *project*.

Project Reasoning

Project reasoning incluye todos los factores que requiere el *project*, ya sean positivos o negativos, elegidos o no (por ejemplo, la capacidad insuficiente para satisfacer la demanda prevista y existente, la disminución de la satisfacción del *customer*, los bajos beneficios, requisitos legales, etc.)

Project Timescales

Las escalas de tiempo reflejan la longitud o duración de un *project*. Las escalas de tiempo relacionadas con el caso de negocios también incluyen el tiempo durante el cual se realizarán los beneficios del *project*.

Reunión de la Visión del Proyecto

A *Reunión de la Visión del Proyecto* es una reunión con el/los *Program Stakeholder(s)*, *Program Propietario del producto*, *Program Scrum Master*, y *Jefe Propietario del producto*. Ayuda a identificar el contexto empresarial, *business requirements* y las expectativas de los *stakeholders* con el fin de desarrollar un *Reunión de la Visión del Proyecto* eficaz.

Reunión de la Visión del Proyecto

El producto clave del proceso *Create Project Statement* es un *Reunión de la Visión del Proyecto* bien estructurado. Un buen *Project Vision* explica la necesidad de la empresa y lo que el *project* tiene por objeto satisfacer en lugar de cómo se va a satisfacer la necesidad.

Proposed Non-Functional Items for Product Backlog

Los requisitos no funcionales no pueden ser totalmente definidos en las primeras etapas del *project* y pueden surgir durante el *Sprint Review* o *Retrospectiva de Sprint Meetings*. Estos artículos deben ser añadidos al *Prioritized Product Backlog* a medida que se descubren.

Quality

Quality se define como la capacidad del producto terminado o Entregables para cumplir los *Acceptance Criteria* y alcanzar el valor de negocio que espera el *customer*.

Quality Assurance

Quality assurance se refiere a la evaluación de los procesos y normas que rigen *quality management* en un proyecto para asegurarse de que siguen siendo relevantes. Las actividades de *quality assurance* se llevan a cabo como parte del trabajo.

Quality Control

Quality control se refiere a la ejecución de las actividades de calidad previstas por el *Equipo Scrum* en el proceso de creación de Entregables que potencialmente se pueden mandar. También incluye el aprendizaje de cada conjunto de actividades realizado con el fin de lograr un *continuous improvement*.

Quality Management

Quality management en Scrum le permite a los *customers* tomar conciencia de los problemas en el *project* desde el principio y les ayuda a reconocer si un *project* va a funcionar para ellos o no. *Quality management* en Scrum se facilita a través de tres actividades interrelacionadas:

1. Quality planning
2. Quality control
3. Quality assurance

Quality Planning

Quality Planning se refiere a la identificación y definición del *product* que se requiere de un *Sprint* y del proyecto al igual que los *Acceptance Criteria*, cualquier método de desarrollo que se deba seguir y las responsabilidades principales de los miembros del *Equipo Scrum* en lo que respecta a la calidad.

Refactoring

Refactoring es una herramienta específica para los proyectos de software. El objetivo de esta técnica es mejorar la mantenibilidad del código existente y hacerlo más simple, más conciso y más flexible. *Refactoring* significa mejorar el diseño del código presente, sin cambiar el comportamiento del código. Se trata de lo siguiente:

- La eliminación de código repetitivo y redundante
- Separar los métodos y las funciones en rutinas más pequeñas
- Definir las variables con claridad y los nombres de los métodos
- Simplificar el diseño del código
- Hacer que el código sea más fácil de entender y modificar

Rejected Deliverables

Rejected Deliverables son los entregables que no cumplen con los *Acceptance Criteria* definidos. Una lista de los *Rejected Deliverables* se mantiene y se actualiza después de cada *Sprint Review Meeting* con los entregables que no fueron aceptados.

Relative Prioritization Ranking

Relative Prioritization Ranking es una simple enumeración de *User Stories* en el orden de prioridad. Es un método eficaz para determinar los deseados *User Stories* para cada iteración o versión del *product* o servicio.

Relative Sizing/Story Points

Además de ser utilizado para la estimación de costos, *Story Points*, también se pueden utilizar para estimar el tamaño total de un *User Story* o una función. Este procedimiento consiste en atribuir un valor en puntos basado en una evaluación general del tamaño de un *User Story* con la consideración dada al riesgo, la cantidad de esfuerzo que se requiere, y el nivel de complejidad.

Release Content

Esto consiste en la información esencial acerca de los entregables que puede ayudar al Equipo de Apoyo al *Customer*.

Release Notes

Release Notes debe incluir criterios de envío externos o de mercados regulados del *product* a entregar.

Release Planning Schedule

Un *Release Planning Schedule* es uno de los resultados más importantes del proceso de *Realizar el plan de lanzamiento Schedule*. A *Release Planning Schedule*, indica que entregas van a ser despachados a los *customers*, junto con intervalos planificados, y fechas para los lanzamientos. Puede ser que no haya un lanzamiento previsto a finales de cada iteración *Sprint*.

Release Planning Sessions

El objetivo principal de *Release Planning Session* es crear un *Release Plan Schedule* y ayudar al *Equipo Scrum* a tener una visión general de los *releases* y del calendario de entregas para el *product* que están desarrollando, para que puedan alinearse con las expectativas del *Propietario del producto* y del/de los *Stakeholder(s)*.

Release Prioritization Methods

Release Prioritization Methods se utilizan para desarrollar un *Release Plan*. Estos métodos son específicos a la industria y a la organización y generalmente son determinados por la alta dirección de una organización.

Resolved Issues

En *Scrum of Scrum Meetings*, los miembros del *Equipo Scrum* tienen la oportunidad de discutir con transparencia sobre los *issues* que influyen en su *project*. Esta discusión y resolución oportuna de los *issues* en el *Scrum of Scrum Meeting* mejora notablemente la coordinación entre los diferentes *Equipos Scrum* y también reduce la necesidad de rediseñar y volver a trabajar.

Retrospectiva del proyecto

En este proceso, que completa el *project*, los socios de la organización y los miembros del Equipo Principal de Scrum se reúnen para la retrospectiva del *project* e identificar, documentar e internalizar las lecciones aprendidas. A menudo, estas lecciones llevan a la documentación de *Agreed Actionable Improvements*, que se aplicará en futuros proyectos.

Retrospectiva del proyecto Meeting

Retrospectiva del proyecto Meeting es una reunión para determinar las formas en que *Colaboración* por parte del equipo y la eficacia se puede mejorar en futuros proyectos. También se discuten aspectos positivos, negativos y *opportunities* de posible mejora. Esta reunión no es *Time-boxed* y se puede realizar en persona o en un formato virtual.

Retrospectiva de Sprint

En este proceso, el *Scrum Master* y el *Equipo Scrum* se encuentran para discutir las lecciones aprendidas a lo largo del *Sprint*. Las lecciones aprendidas se documentan y se pueden aplicar a los futuros *Sprints*.

Retrospectiva de Sprint Log(s)

El *Retrospectiva de Sprint Log(s)* es un registro de las opiniones, discusiones y artículos recurribles planteados en un *Retrospectiva de Sprint Meeting*. El *Scrum Master* puede facilitar la creación de este registro con entradas (inputs) de los miembros del Equipo Principal Scrum.

Retrospectiva de Sprint Meeting

Retrospectiva de Sprint Meeting es *Time-boxed* por 4 horas para un *Sprint* de un mes y se lleva a cabo como parte del proceso *Sprint Retrospect*. La longitud se puede escalar hacia arriba o hacia abajo con respecto a la longitud del *Sprint*. Durante esta reunión, el *Equipo Scrum* se reúne para revisar y reflexionar

sobre el *Sprint* anterior en términos de los procesos que fueron aplicados, las herramientas empleadas, *Colaboración* y los mecanismos de comunicación y otros aspectos de interés para el *project*.

Return on Investment (ROI)

Return on Investment (ROI), cuando se utiliza para la justificación del *project*, evalúa el ingreso neto que se espera obtener a partir de un proyecto. Se calcula restando los costos o la inversión en un *project* desde su ingreso esperado y dividiendo este (beneficio neto) por los costos esperados con el fin de obtener una tasa de retorno.

Risk

El riesgo se define como un evento incierto o conjunto de eventos que pueden afectar a los objetivos de un proyecto y pueden contribuir a su éxito o fracaso.

Risk Appetite

Risk appetitive se refiere a la cantidad de incertidumbre un *stakeholder* u organización está dispuesta a asumir.

Risk Assessment

Risk Assessment se refiere a la evaluación y la estimación de *risks* identificados.

Risk Attitude

En esencia, el *Risk Attitude* del/de los *Stakeholder(s)* determina cuánto riesgo el/los *stakeholder(s)* considera(n) aceptable. Este es un factor determinante para decidir que medidas seguir para mitigar los posibles *risks* adversos.

Risk Averse

Risk Averse es una de las categorías de *Utility Function*. Se refiere a un *Stakeholder* que no está dispuesto a aceptar un riesgo, no importa cuál es el beneficio esperado u oportunidad.

Risk Breakdown Structure

En esta estructura, los *risks* se agrupan en función de sus categorías o elementos comunes. Por ejemplo, *risks* pueden ser categorizados como financiero, técnico o relacionada con la seguridad.

Risk Burndown Chart

Un gráfico que muestra la gravedad del riesgo del *project* acumulativo a lo largo del tiempo. La probabilidad de los diversos *risks* se representa uno arriba del otro para mostrar riesgo acumulado en el eje. La identificación y evaluación inicial de los *risks* y la creación del *Risk Burndown Chart* se hace al principio del proyecto.

Risk Checklists

Risks Checklists incluye los puntos claves a considerar al identificar "Riesgos", los riesgos más comunes en los proyectos Scrum, o incluso las categorías de *risks* que deben ser abordadas por el equipo.

Risk Communication

Risk Communication implica la comunicación de los resultados de los primeros cuatro pasos de Risk Management a los *socios* apropiado(s) y la determinación de su percepción con respecto a los sucesos inciertos.

Risk Identification

Risk Identification es un paso importante en la gestión del riesgo que implica el uso de diversas técnicas para identificar todos los posibles *risks*.

Risk Meeting

Risks pueden ser priorizados más fácilmente por el *Propietario del producto* al llevar a cabo una reunión del equipo Central de Scrum y, opcionalmente, invitando a los *socios* relevantes de la reunión.

Risk Mitigation

Risk Mitigation es un paso importante en *Risk Management* que implica el desarrollo de una estrategia adecuada para hacer frente a un riesgo.

Risk Neutral

Risks Neutral es una de las categorías de *Utility Function*, que se refiere al *stakeholder* que no es ni risk averse, ni *risk seeking*; cualquier decisión que no se ve afectada por el nivel de incertidumbre de los resultados. Cuando dos posibles escenarios tienen el mismo nivel de beneficio, el *risk neutral stakeholder* no va a estar preocupado si una hipótesis es más arriesgada que la otra.

Risk Prioritization

Risk Prioritization es un paso importante en *Risk Management* que implica priorizar *risks* que se incluirán para la acción específica en el *Prioritized Product Backlog*.

Risk Prompt Lists

Risk Prompt Lists se utilizan para estimular pensamientos con respecto a la fuente de donde los *risks* pueden originar. *Risk Prompt Lists* para diversas industrias y tipos de *projects* está disponible al público.

Risk Seeking

Risk Seeking es una de las categorías de *Utility Function* que se refiere a que un *stakeholder* esté dispuesto a asumir un riesgo, aún si ofrece un aumento marginal de retorno o beneficio para el *project*.

Risk threshold

Risk threshold se refiere al nivel en el que un riesgo es aceptable para la organización de los *socios*. Un riesgo caerá por encima o por debajo del *risk threshold*. Si está por debajo, el *stakeholder* u organización es más probable que acepte el riesgo.

Risk tolerance

Risk tolerance indica el grado, cantidad o volumen de los riesgos que los *socios* resistirán.

Risk-Based Spike

Risk-Based Spike son, básicamente, los experimentos que involucran la investigación o la creación de prototipos para entender mejor los *risks* potenciales. En un punto, se lleva a cabo un intenso ejercicio de dos a tres días (preferentemente al principio de un *project* antes de los procesos de *Desarrollo de épica(s)* o *Creación de la lista priorizada de pendientes del producto*) para ayudar al equipo a determinar las incertidumbres que podría afectar al *project*.

Scope

El *scope* de un *project* es la suma total de todos los incrementos de los *products* y el trabajo necesario para desarrollar el producto final.

Cuerpo de asesoramiento de Scrum

Cuerpo de asesoramiento de Scrum (SGB) es una función opcional. Por lo general, se compone de un grupo de documentos y/o un grupo de expertos que normalmente están envueltos en la definición de objetivos relacionados con la calidad, las regulaciones gubernamentales, la seguridad y otros parámetros claves de la organización.

Cuerpo de asesoramiento de Scrum Expertise

Cuerpo de asesoramiento de Scrum Expertise se refiere a las normas documentadas y reglamentos, directrices de desarrollo, o los estándares y mejores prácticas.

Scrum Master

El *Scrum Master* es uno de los roles de Equipo Principal de Scrum. Él o ella facilita la creación de entregables del *project*, gestiona risks, cambios e *impediments* durante *Llevar a cabo el Standup diario*, *Retrospectiva de Sprint* y otros procesos de Scrum.

Scrum of Scrums Meeting

Scrum of Scrum (SoS) Meeting es un encuentro importante al escalar Scrum para grandes *projects* en el que asisten representantes de todos los equipos. Esta reunión es generalmente facilitada por el *Jefe Scrum Master* y su objetivo es centrarse en las áreas de coordinación e integración entre los diferentes *Equipos Scrum*. Esta reunión se lleva a cabo en intervalos predeterminados o cuando lo requieran los *Equipos Scrum*.

Equipo Scrum

Equipo Scrum es uno de los roles del Equipo Principal de Scrum (*Scrum Core Team*). El Equipo Principal de Scrum trabaja en la creación de los entregables del *project* y contribuye a la realización de valor de negocio para todos los *socios* y el *project*.

Equipo Scrum Lessons Learned

Se espera que el *Equipo Scrum*, que es un equipo auto-organizado, aprenda de los errores cometidos durante el *Sprint* y que estas lecciones aprendidas ayuden a los equipos a mejorar su desempeño en futuros Sprints.

Equipo Scrum Representatives

Es un representante designado por el equipo para que los represente en los *Scrum of Scrums (SoS) Meetings* basado en quien mejor puede cumplir el papel en función de *issues* actuales y circunstancias.

Scrumboard

Scrumboard es una herramienta utilizada por el *Equipo Scrum* para planificar y realizar un seguimiento de los progresos realizados durante cada *Sprint*. El *Scrumboard* contiene cuatro columnas para indicar el progreso de las tareas estimadas para el *Sprint*: una columna se llama *To Do*, para las tareas aún no iniciadas, otra columna llamada *Progress*, para las tareas en progreso que aún no se han completado, una columna llamada *Testing*, para las tareas completadas pero que están en proceso de hacerse las pruebas, y la columna *Done*, para las tareas que se han completado y probado con éxito.

Auto-organización

Scrum cree que los empleados son auto-motivados y desean una mayor responsabilidad. Por lo tanto, los empleados ofrecen mucho más valor cuando se organizan por cuenta propia.

Servant Leader

Los *Servant Leaders* escuchan, sienten empatía y compromiso, crean una visión, comparten el poder y la autoridad con los miembros del equipo. *Servant leaders* ayudan a que se logren buenos resultados, centrándose en las necesidades del equipo. Este estilo es la realización de la función *Scrum Master*.

Envío de entregables

En este proceso, *Accepted Deliverables* se entregan o pasan al/a los *Stakeholder(s)* pertinente(s). Un *Working Deliverables Agreement* formal documenta la finalización con éxito del *Sprint*.

Simple Schemes

Simple Schemes implican etiquetar elementos como prioridad "1", "2", "3" o "Alto", "Medio" y "Bajo" y así sucesivamente. Aunque se trata de un método sencillo y directo, puede llegar a ser problemático porque a menudo hay una tendencia de etiquetar todo como prioridad "1" o "Alto".

Skills Requirement Matrix

Skill Requirement Matrix, también conocido como un marco de competencias, se utiliza para evaluar las carencias de cualificaciones y los requisitos de formación para los miembros del equipo. Una matriz de habilidades asigna las competencias, las capacidades y el nivel de interés de los miembros del equipo en el uso de esas habilidades y capacidades en un proyecto. Usando esta matriz, la organización puede evaluar las carencias de habilidades en los miembros del equipo e identificar a los empleados que van a necesitar más formación en un área o competencia particular.

Speed Boat

Speed Boat es una técnica que se puede utilizar para llevar a cabo el *Retrospectiva de Sprint Meeting*. Los miembros del equipo hacen de tripulación en un *Speed Boat*. El barco debe llegar a una isla, que es un símbolo de la visión del proyecto. Las notas adhesivas son utilizadas por los asistentes para registrar los motores y anclas. Los motores son cosas que ayudan a llegar a la isla, mientras que los anclas son cosas que están obstaculizando que lleguen a la isla. Este ejercicio es time-boxed a unos pocos minutos.

Sponsor

El *sponsor* (patrocinador) es la persona o la organización que provee recursos y apoyo para el proyecto. El patrocinador es también el *stakeholder*, a quien todos le rinden cuentas.

Sprint

Un *Sprint* es una iteración *time-boxed* de una a seis semanas de duración durante el cual el *Equipo Scrum* crea y trabaja en el *Sprint deliverables*.

Sprint Backlog

Sprint Backlog es una lista de las tareas a ser ejecutadas por el *Equipo Scrum* en el próximo *Sprint*.

Sprint Burndown Chart

Sprint Burndown Chart es un gráfico que muestra la cantidad de trabajo que queda en el *Sprint* corriente.

Sprint Deliverables

Sprint Deliverables hacen referencia a incrementos del producto o de los productos que se realizan al final de cada *Sprint*.

Sprint Planning Meeting

Sprint Planning Meeting se lleva a cabo al comienzo de un *Sprint*, como parte del proceso *Elaboración de la lista de pendientes del Sprint* de *Sprint Backlog*. Es *Time-boxed* por ocho horas durante un *Sprint* de un mes y se divide en dos partes – *Objective Definition* y *Task Estimation*.

Sprint Review Meeting

Sprint Review Meeting es *time-boxed* por cuatro horas para un *Sprint* de un mes y se puede escalar de acuerdo a la longitud del *Sprint*. Durante el *Sprint Review Meeting*, el *Equipo Scrum* presenta los entregables del *Sprint* actual al *Propietario del producto*, que puede aceptar o rechazar las prestaciones.

Sprint Tracking Tools

Sprint Tracking Tools se utilizan para realizar el seguimiento del progreso de un *Sprint* y para saber dónde está el *Equipo Scrum* en términos de completar las tareas del *Sprint Backlog*. Una variedad de herramientas se puede utilizar para realizar el seguimiento del trabajo en un *Sprint*, pero uno de los más comunes es un *Scrumboard*, también conocida como tabla de tarea o un gráfico de progreso.

Sprint Velocity

Sprint Velocity es la velocidad en la que el equipo puede completar el trabajo en un *Sprint*. Por lo general se expresa en las mismas unidades que los utilizados para la estimación, normalmente *Story Points* o el tiempo ideal.

Stakeholder(s)

Stakeholder(s) es un término colectivo que incluye *customers*, los usuarios y patrocinadores que frecuentemente interactúan con el *Propietario del producto*, *Scrum Master* y *Equipo Scrum* para proveer entradas (*inputs*) y facilitar la creación de producto del proyecto, servicio, u otros resultados.

Storming stage

Storming Stage es la segunda etapa de la formación del equipo, donde el equipo comienza a tratar de cumplir con el trabajo. Sin embargo, las luchas por el poder pueden ocurrir resultando en caos o confusión entre los miembros del equipo.

Story Mapping

Story Mapping es una técnica para proporcionar un esquema visual del producto y sus componentes claves. *Story Mapping*, formulada por primera vez por Jeff Patton (2005), es comúnmente utilizado para ilustrar el camino al producto. *Story Maps* representan la secuencia del desarrollo del producto, y trazan que elementos se incluirán en la primera, segunda, tercera, y las versiones posteriores.

Sustainable Pace

Sustainable Pace es el ritmo en el que el equipo puede trabajar cómodamente. Esto se traduce en una mayor satisfacción de los empleados, la estabilidad y aumento de la precisión de la estimación, todo lo cual en última instancia conduce a una mayor satisfacción del *customer*.

SWOT Analysis

SWOT es un enfoque estructurado para la planificación que ayuda a evaluar las fortalezas, debilidades, *opportunities* y amenazas relacionados con un proyecto. Este tipo de análisis ayuda a identificar tanto los factores internos como externos que podrían afectar el proyecto.

Target Customers for Release

No todos los lanzamientos se dirigirán a todas los *socios* o usuarios. Los *stakeholder* pueden optar por limitar ciertos comunicados a un subconjunto de usuarios. El plan de lanzamiento especifica los *target customers* para el lanzamiento.

Task Estimation Workshop

Task Estimation Workshop sirve para estimar el esfuerzo necesario para completar una tarea o conjunto de tareas y estimar el esfuerzo de las personas y otros recursos necesarios para llevar a cabo las tareas dentro de un determinado *Sprint*.

Task List

Esto es una lista completa con todas las tareas a las que el *Equipo Scrum* se ha comprometido al Sprint actual. Contiene descripciones de cada tarea.

Task Planning Meeting

En un *Task Planning Meeting* un *Equipo Scrum* se reúne para planificar el trabajo que se hará en el *Sprint* y el equipo revisa los *User Stories* comprometidos en la parte superior del *Prioritized Product Backlog*. Para ayudar a asegurar que el grupo se queda en el tema, esta reunión debería ser *Time-boxed*, con la longitud estándar limitada a dos horas por semana de duración *Sprint*.

Task-Oriented Leader

Task-Oriented Leaders hacen cumplir la realización de tareas y el cumplimiento de los plazos.

Team Building Plan

Dado a que un *Equipo Scrum* es multi-funcional, cada miembro debe participar activamente en todos los aspectos del proyecto. El *Scrum Master* debe identificar los *issues* potenciales que podrían surgir con miembros del equipo y tratar de resolverlos con diligencia en el *Team Building Plan* a fin de mantener un equipo eficaz.

Team Calendar

Team Calendar contiene información sobre la disponibilidad de los miembros del equipo, incluyendo la información correspondiente a las vacaciones de los empleados, las licencias, acontecimientos importantes, y los días festivos.

Team Expertise

Team Expertise se refiere a la experiencia de los miembros del *Equipo Scrum* para comprender los *User Stories* y las tareas en el *Sprint Backlog* con el fin de crear los entregables finales. *Team Expertise* se utiliza para evaluar las entradas necesarias para ejecutar la obra prevista del proyecto.

Technical Debt

Technical Debt (también conocido como deuda de diseño o deuda de código) se refiere al trabajo que los equipos dan prioridad más baja, omiten o no se completan a medida que trabajan en la creación de los entregables principales asociados con el producto del proyecto. *Technical Debt* acumula y debe ser pagado en el futuro.

Theory X

Líderes de *Theory X* suponen que los empleados son intrínsecamente motivados y evitarán el trabajo si es posible, lo que justifica un estilo autoritario de gestión.

Theory Y

Líderes de *Theory Y* asumen que los empleados son auto-motivados y tratan de aceptar una mayor responsabilidad. *Theory Y* implica un estilo de gestión más participativo.

Threats

Threats son *risks* que podrían afectar al *project* de una manera negativa.

Three Daily Questions

Tres preguntas diarias utilizadas en *Daily Standup Meetings* que se ven facilitada por el *Scrum Master*, donde cada miembro del *Equipo Scrum* proporciona información en forma de respuestas a estas tres preguntas específicas:

- ¿Qué terminé ayer?
- ¿Qué terminaré hoy?
- ¿Qué *impediments* u obstáculos (si los hay) estoy encarando en este momento?

Tiempo asignado

Tiempo asignado se refiere al ajuste de periodos cortos de tiempo para el trabajo por hacer. Si el trabajo realizado está incompleto al final del *Time-box*, se mueve al *Time-box* posterior. *Time-box* proporciona la estructura necesaria para los *projects Scrum* que tienen un elemento de incertidumbre, que son dinámicos por naturaleza y son propensos a cambios frecuentes.

Transparencia

La transparencia permite que todas las facetas de cualquier proceso de Scrum puedan ser observadas por cualquier persona. Compartir toda la información conduce a un entorno de alta confianza.

Unapproved Change Requests

Solicitud de cambio se presentan por lo general como *Change Requests*. Los *Change Requests* permanecen sin ser aprobados hasta que pueden obtener la aprobación formal.

Updated Program Product Backlog

Un *Program Product Backlog* que se somete a la preparación periódica para incorporar los cambios y las nuevas necesidades.

User

Los usuarios son los individuos o la organización que utiliza directamente el producto del proyecto, servicio u otros resultados. Al igual que *customers*, para cualquier organización, pueden haber tanto usuarios internos como externos. En algunos casos, los *customers* y los usuarios pueden ser los mismos.

User Group Meetings

User Group Meetings implica al/ a los *Stakeholder(s)* correspondiente(s), principalmente los usuarios o *customers* del *product*. Ellos proporcionan al Equipo Principal de Scrum con información de primera mano acerca de las expectativas del usuario. Esto ayuda en la formulación de los *Acceptance Criteria* para el producto y proporciona información valiosa para el desarrollo de *Epics*.

User Stories

User Stories se adhieren a una estructura específica, predefinida y son una manera simplista de la documentación de los requisitos y la funcionalidad del usuario final deseado. Los requisitos expresados en *User Stories* son afirmaciones breves, simples y fáciles de entender lo que resulta en una mejor comunicación entre los *socios* y el equipo.

User Story Acceptance Criteria

Cada *User Story* tiene su *Acceptance Criteria*. *User Stories* son subjetivos, por lo que los *Acceptance Criteria* proporcionan la objetividad requerida para que el *User Story* sea considerado como *Done* o no durante el *Sprint Review*. Esto le proporciona claridad al equipo sobre lo que se espera de un *User Story*.

User Story Workshops

User Story Workshops se llevan a cabo como parte del proceso de *Desarrollo de épica(s)*. El *Scrum Master* facilita estas sesiones. El *Equipo Principal de Scrum* entero participa y, a veces, es deseable incluir otros *stakeholder(s)*.

User Story Writing Expertise

El *Propietario del producto*, basado en su interacción con los *socios*, conocimiento del negocio y la experiencia propia, al igual que las aportaciones del equipo, desarrolla *User Stories* que forman el Prioritized Product Backlog inicial para el proyecto.

Utility Function

Utility Function es un modelo utilizado para medir la preferencia del *stakeholder* por el riesgo o la actitud hacia el riesgo. Esto define el nivel del *stakeholder(s)* para aceptar riesgos.

Value Stream Mapping

Value Stream Mapping utiliza diagramas de flujo para ilustrar el flujo de información necesaria para completar un proceso y puede ser usado para simplificar un proceso, ayudando a determinar los elementos que no aportan valor.

Vendor

Los *vendors* incluyen a individuos u organizaciones externas que ofrecen *products* y servicios que no están dentro de las competencias básicas de la organización del *project*.

Voice of the Customer (VOC)

Voice of the Customer (VOC) se puede denominar como los requisitos explícitos e implícitos del *customer*, que deben ser entendidos antes del diseño de un *product* o servicio. El *Propietario del producto* representa la voz del *customer*.

War Room

War Room es el término comúnmente utilizado para describir la ubicación donde trabajan todos los miembros del *Equipo Scrum*. Normalmente, está diseñado de tal manera que los miembros del equipo pueden moverse libremente, trabajar y comunicarse fácilmente ya que se encuentran en proximidades inmediatas.

Wideband Delphi Technique

Wideband Delphi es una técnica de estimación basada en el grupo para la determinación de la cantidad de trabajo involucrado y el tiempo que tardará en completarse. Los individuos dentro de un equipo proporcionan estimaciones anónimas para cada función y estas estimaciones se trazan en un gráfico. Posteriormente, el equipo analiza los factores que influyeron en sus estimaciones y proceden a una segunda ronda de estimación. Este proceso se repite hasta que las estimaciones de los individuos son cerca una de la otra y un consenso para la estimación final se puede llegar.

Working Deliverables

Esta salida es el entregable listo para enviar (*shippable deliverable*) final para el que fue sancionado el proyecto.

Working Deliverables Agreement

Entregas que cumplen con los *Acceptance Criteria* reciben el cierre de negocio y la aprobación formal por parte del *Customer* o del patrocinador.

INDEX

1

100-point method, 171

A

Accepted Deliverables, 248

Actual Cost, 77

Adaptability, 4

Adaptation, 24

Affinity Estimation, 195

Agile Expert Certified (AEC™), 6

Agreed Actionable Improvements, 252

Applicable Contracts, 161

Appropriation, 29

Aprobar, estimar y asignar historias de usuarios, 17, 182

 inputs, 193

 outputs, 196

 tools, 193

Approved Change Requests, 99, 160

Approved Changes, 166

Approved, Estimated, and Committed User Stories, 196

Articulación, 29

Assertive, 61

Assigned Action Items and Due Dates, 252

Autocratic, 61

Awareness, 29

B

Back-up Persons, 157

Better Team Coordination, 243

Brainstorming, 121

Budget at Completion, 77

Business case, 70

Justificación del negocio, 13

Business Requirements, 169

Business value, 85

Business value delivered, 208

C

Change, 14

Change approval process, 99

Jefe Propietario del producto, 45, 140

Jefe Scrum Master, 47, 147

Coaching, 61

Colaboración, 10, 21

Colaboración Plan, 157

Collective ownership, 4

Colocated Teams, 31

Communication Plan, 261

Communication Techniques, 231

Company Vision, 141

Llevar a cabo el Standup diario, 18, 212

 inputs, 223

 outputs, 225

 tools, 224

Realizar el plan de lanzamiento, 17, 134

 inputs, 176

 outputs, 178

 tools, 177

Confirm benefits realization, 71, 80

Conflict Management, 59

Continuous delivery of value, 4

Continuous feedback, 4

Continuous improvement, 4

Continuous integration, 106

Continuous value justification, 71, 76

Convocar Scrum de Scrums, 18, 236

 inputs, 240

 outputs, 243

 tools, 241

Core roles, 11

Cost Performance Index, 77

Cost Variance, 77

Crear entregables, 18, 212

 inputs, 217

 outputs, 220

 tools, 219

Creación de la lista priorizada de pendientes del producto, 17, 134

 inputs, 169

 outputs, 172

 tools, 170

Crear la visión del proyecto, 17, 134, 137, 145, 152, 158,

 167, 174, 185, 192, 197, 201, 205, 215, 222, 227,

 239, 244, 249, 259, 263

- inputs, 139
 outputs, 143
 tools, 142
Elaboración de la lista de pendientes del Sprint, 18, 182
 inputs, 207
 outputs, 209
 tools, 208
Elaboración de tareas, 18, 182
 inputs, 198
 outputs, 200
 tools, 198
Elaborar historias de usuario, 17, 182
 inputs, 187
 outputs, 190
 tools, 188
Customer, 42
Customer centric, 4
Customer Value-based Prioritization, 74, 105
- D**
- Daily Standup Meeting**, 34, 224
Data Flow Diagram
 Implement phase, 233
 Initiate phase, 180
 Plan and Estimate phase, 210
 Release phase, 267
 Review and Retrospect phase, 254
Decomposition, 199
Delegating, 61
Demostración y validación del Sprint, 18, 236
 inputs, 246
 outputs, 248
 tools, 247
Demonstrations, 80
Dependencies, 173, 200
Dependency Determination, 199
Design Patterns, 220
Desarrollo de épica(s), 17, 134
 inputs, 160
 outputs, 165
 tools, 163
Development in Phases Contract, 162
Directing, 61
Discretionary dependencies, 199
Dissatisfiers, 75
Distributed Teams, 31
Done, 89
Done Criteria, 173
- Done success rate, 251
- E**
- Earned Value*, 77
Effective Deliverables, 4
Efficient development process, 4
Effort Estimated Task List, 204
Control del proceso empírico, 10, 21, 22
Epic(s), 165
Estimate at Completion, 77
Estimate Range, 196
Estimar tareas, 18, 182
 inputs, 202
 outputs, 204
 tools, 203
Estimate to Complete, 77
Estimates, 173
Estimation Criteria, 203
Estimation effectiveness, 252
Expected Monetary Value (EMV), 124
Expert Advice from HR, 150, 155
Expert Scrum Master (ESM™), 6
Explorer – Shopper – Vacationer – Prisoner (ESVP), 251
External dependencies, 200
- F**
- Faster problem resolution**, 4
Fist of Five, 194
Focus Group Meetings, 163, 189
Formación de un equipo Scrum, 17, 134
 inputs, 154
 outputs, 156
 tools, 155
Forming, 58
Four Questions per Team, 241
- G**
- Gap Analysis*, 143
Mantenimiento de la lista priorizada de pendientes del producto, 18, 212, 227
 inputs, 229
 outputs, 232
 tools, 231

H

High trust environment, 4
High velocity, 5
Holiday Calendar, 177

I

Identified Propietario del producto, 143
Identified Risks, 166
Identified Scrum Master, 151
Identified Equipo Scrum, 156
Identified Stakeholder(s), 151
Identificar al Scrum Master y al socio(s), 17, 134, 145
 inputs, 147
 outputs, 151
 tools, 149
Impediment Log, 218
Implement, 18
Incentive and Penalty Contract, 162
Incremental Delivery Contract, 161
Index Cards, 199
Indifferent, 75
Initiate, 17, 133
Innovative environment, 5
Inspection, 24
Integrating Change, 106
Internal dependencies, 200
Desarrollo iterativo, 10, 22, 36, 101

J

JAD Sessions, 142
Joint Venture Contract, 162

L

Laissez Faire, 61
Laws and Regulations, 161
Leadership Styles, 61
Length of Sprint, 108, 179
Lose/Lose, 60
Lose/Win, 60

M

Mandatory dependencies, 199
Market Study, 141
Meeting Agenda, 240

Meeting Room, 242
Metrics and Measuring Techniques, 251
Minimum Acceptance Criteria, 88
Mitigated Risks, 221
Monopoly money, 74
MoSCoW prioritization, 170
Motivated Equipo Scrum, 226
Motivation, 4

N

Non-core roles, 12
Norming, 58
Number of stories, 208

O

Organización, 11
Organizational Deployment Methods, 261
Organizational Resource Matrix, 148
Other Estimation Techniques, 195

P

Paired comparison, 170
Pareto analysis, 123
Peer feedback, 252
People Availability and Commitment, 148
People Costs, 155
People Requirements, 148
Percent Complete, 77
Performing, 58
Personas, 165
Personnel Selection, 49
Piloting Plan, 261
Plan and Estimate, 17
Planned Value, 77
Planning for value, 74
Planning Poker, 193
Points for Cost Estimation, 194
Portfolio, 50, 53
Portfolio Propietario del producto, 53
Portfolio Scrum Master, 53
Portfolios
 Change, 112
 Risk, 129
Previous Project Information, 162
Previous Sprint Velocity, 207
Previous Work Day Experience, 223

Prioritized Product Backlog, 86, 172
 Prioritized Product Backlog Review Meetings, 231
Probability Impact grid, 123
Probability trees, 122
 Product Releases, 262
Program, 50, 53
Program and Portfolio Risks, 161
Program Product Backlog, 140
Program Propietario del producto, 53, 139
Program Scrum Master, 53, 139
Program Stakeholder(s), 140
Programs
 Change, 112
 Risk, 129
 Progress to release/launch, 252
Project, 50
Project Budget, 144
 Caso de Negocio del Proyecto, 139
Project Charter, 144
 Project costs, 70
Project timescales, 70
 Reunión de la Visión del Proyecto, 142
 Reunión de la Visión del Proyecto, 144
Proof of Concept, 141

Q

Quality, 14
Quality planning, 90
 Questionnaires, 164

R

Refactoring, 219
Refined Prioritized Product Backlog, 179
Rejected Deliverables, 229, 248
Relative prioritization ranking, 76
Relative Sizing / Story Points, 195
 Release, 19
 Release Planning Schedule, 178
 Release Planning Sessions, 177
 Release Prioritization Methods, 178
Resolved Issues, 243
 Resource Costs, 150, 156
 Resource Requirements, 155
Retrospectiva del proyecto, 19, 256
 inputs, 264
 outputs, 266
 tools, 265

Retrospectiva del proyecto Meeting, 265
Retrospectiva de Sprint, 19, 236
 inputs, 250
 outputs, 252
 tools, 250
Retrospectiva de Sprint Log(s), 253
Retrospectiva de Sprint Meeting, 34, 250
Review and Retrospect, 18
 Review feedback ratings, 252
 Risk, 15
 Risk appetite, 119
Risk assessment, 121
 Risk averse, 119
Risk Breakdown Structure (RBS), 121
Risk checklists, 120
Risk communication, 126
Risk identification, 120
 techniques, 120
 Risk management
 procedure, 120
Risk meeting, 122
 Risk mitigation, 126
 Risk neutral, 119
Risk prioritization, 124
 Risk seeking, 120
 Risk threshold, 119
 Risk tolerance, 119

S

Satisfiers, 75
 Scalability of Scrum, 5
Schedule Performance Index, 77
Schedule Variance, 77
Scrum Aspects, 11
Scrum Core Team, 103, 160
Scrum Developer Certified (SDC™), 6
Cuerpo de asesoramiento de Scrum (SGB), 12
Cuerpo de asesoramiento de Scrum Expertise, 165, 172, 189, 196, 220, 242, 247, 252, 265
 Cuerpo de asesoramiento de Scrum Recommendations, 142, 162, 170, 177, 188, 193, 203, 218, 230, 247, 250, 261, 264
 Scrum Master, 45, 240
Scrum Master Certified (SMC™), 6
 Scrum of Scrums Meeting, 241
Scrum Processes, 16
Scrum Propietario del producto Certified (SPOC™), 6
Equipo Scrum, 47

Equipo Scrum Lessons Learned, 253
Equipo Scrum Representatives, 240
Equipo Scrum Selection, 155
Scrumboard, 217
Selection Criteria, 149
Auto-organización, 10, 21
Senior Management, 104
Servant Leadership, 61
Envío de entregables, 19, 256, 259
 inputs, 260
 outputs, 262
 tools, 261
Skills Requirement Matrix, 149
Software, 219
Speed Boat, 251
Sponsor, 43
Sprint, 33
Sprint Backlog, 209
Sprint Burndown Chart, 209
Sprint Deliverables, 220
Sprint Planning Meeting, 34, 208
Sprint Review Meeting, 34, 247
Sprint Tracking Metrics, 208
Sprint Tracking Tools, 208
Stability, 100
Stakeholder Involvement, 55
Stakeholder(s), 12, 102
Storming, 58
Story Mapping, 76
Supportive, 61
Sustainable pace, 4, 92
SWOT Analysis, 143

T

Target Customers for Release, 179
Task Estimation Meetings, 203
Task List, 200
Task Planning Meetings, 198
Team Building Plan, 157
Team Calendar, 207
Team Expertise, 219
Team morale ratings, 252
Team velocity, 251
Theory X, 64
Theory Y, 64
Three Daily Questions, 224
Tiempo asignado, 10, 22
Traditional Project Management, 20

Training and Training Costs, 150, 156
Transparencia, 4, 22
Trial Project, 141
Tuckman's Model of Group Dynamics, 58

U

Unapproved Change Requests, 99, 160
Uncertainty, 173
Updated Approved, Estimated, and Committed User Stories, 200
Updated or Refined Personas, 191
Updated Prioritized Product Backlog, 191, 232
Updated Program Product Backlog, 230
Updated Release Planning Schedule, 232
Updated Cuerpo de asesoramiento de Scrum Recommendations, 253
Updated Scrumboard, 220
Updated Task List, 204
User Group Meetings, 163
User or Customer Interviews, 164
User Stories, 190, 193
User Story Acceptance Criteria, 190, 202
User Story Estimation Methods, 172, 189
User Story Prioritization Methods, 170
User Story Workshops, 163
User Story Writing Expertise, 188
Users, 43

V

Value, 172
Value Stream Mapping, 74
Value-based Prioritization, 10, 22, 31
Value-driven Delivery, 65, 66
Variance at Completion, 77
Velocity, 208
Vendors, 43
Video Conferencing, 225, 242
Voice of the Customer, 45

W

War Room, 225
Wideband Delphi, 195
Win/Lose, 60
Win/Win, 59
Working Deliverables, 262
Working Deliverables Agreement, 262

La guía esencial para entregar proyectos exitosos utilizando Scrum

La Guía *SBOK™* fue desarrollada como un medio para crear una guía necesaria para organizaciones y profesionales que desean implementar Scrum, así como para aquellos que ya lo están haciendo pero desean aun mejorar sus procesos existentes. Se basa en la experiencia extraída de miles de proyectos a través de una variedad de industrias y organizaciones. Las contribuciones de muchos expertos de Scrum y de profesionales de proyecto han sido consideradas en su desarrollo.

El enfoque de Scrum en la entrega del valor ayuda a los equipos Scrum a entregar resultados tan temprano en el proyecto como sea posible, mejorando la rentabilidad de la inversión (Return on Investment) para las empresas que utilizan Scrum como su marco de entrega de proyecto preferido. Por otra parte, el gestionar cambios es fácil mediante el uso de ciclos de desarrollo de producto cortos e iterativos y de interacción frecuente entre los clientes y el equipo Scrum.

La Guía *SBOK™* puede ser utilizada como una guía de referencia y conocimiento tanto por los desarrolladores de productos y servicios con conocimiento de Scrum, así como por individuos sin experiencia previa o conocimiento de Scrum u otra metodología de entrega de proyectos. El primer capítulo describe el propósito y el marco de la Guía *SBOK™* y proporciona una introducción a los conceptos claves de Scrum y un resumen de los principios, aspectos y procesos del mismo. El capítulo 2 expande en los seis principios de Scrum que constituyen la base en la que se basa el marco de Scrum. Los capítulos 3 al 7 elaboran en los cinco aspectos de Scrum que deben ser abordados a lo largo de cualquier proyecto, organización, justificación del negocio, calidad, cambio y riesgo. Los capítulos 8 al 12 cubren los 19 procesos de Scrum para llevar a cabo un proyecto Scrum. Estos procesos son parte de las 5 fases de Scrum de Iniciar; Planear y Estimar; Implementar, Revisión y Retrospectiva; y la Lanzamiento. Se describen los detalles sobre las entradas y salidas asociadas de cada proceso, así como las diversas herramientas que pueden utilizarse en cada una.

Aunque la Guía *SBOK™* es un libro de referencia muy completo de Scrum, su contenido se organiza como una referencia fácil y lectura agradable, cualquiera que sea el conocimiento previo del lector del Scrum.

