

Requisitos

Dos tipos

- Funcionales
 - qué es lo que el producto o sistema debe ser capaz de hacer (features)
- No Funcionales
 - restricciones adicionales mas orientadas al cómo que al qué
 - performance, usabilidad, mantenibilidad, plataforma

Documento de Requisitos

1. Introduction

- 1.1 Purpose
- 1.2 Document conventions
- 1.3 Intended audience
- 1.4 Additional information
- 1.5 Contact information/SRS team members
- 1.6 References

2. Overall Description

- 2.1 Product perspective
- 2.2 Product functions
- 2.3 User classes and characteristics
- 2.4 Operating environment
- 2.5 User environment
- 2.6 Design/implementation constraints
- 2.7 Assumptions and dependencies

3. External Interface Requirements

- 3.1 User interfaces
- 3.2 Hardware interfaces
- 3.3 Software interfaces
- 3.4 Communication protocols and interfaces

4. System Features

- 4.1 System feature A
 - 4.1.1 Description and priority
 - 4.1.2 Action/result
 - 4.1.3 Functional requirements
- 4.2 System feature B

5. Other Nonfunctional Requirements

- 5.1 Performance requirements
- 5.2 Safety requirements
- 5.3 Security requirements
- 5.4 Software quality attributes
- 5.5 Project documentation
- 5.6 User documentation

6. Other Requirements

- Appendix A: Terminology/Glossary/Definitions list
- Appendix B: To be determined

El problema de las especificaciones

Customer: Hello, I'd like to order a cake.

Employee: Sure, what would you like written on it?

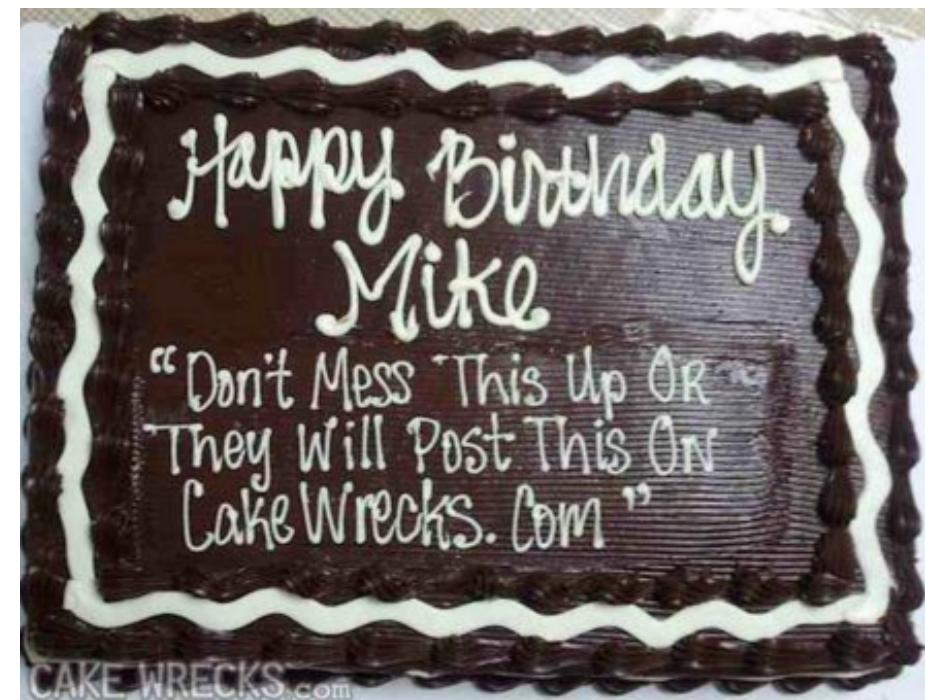
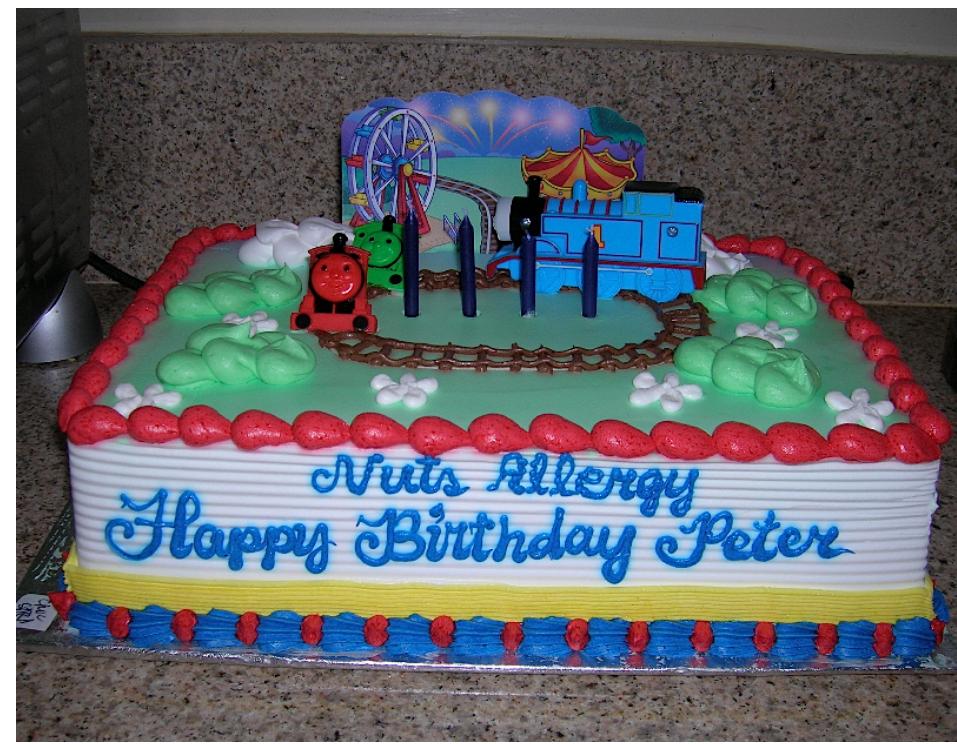
Customer: Could you write "So long, Alicia" in purple?

Employee: Sure.

Customer: And put stars around it?

Employee: No problem. I've written this up, and will hand it to my cake decorator right away. We'll have it for you in the morning.





Levantamiento de Requisitos Funcionales

- Idea - para cada tipo de usuario capturar la forma en que se espera que utilice el producto o sistema
- Diseño centrado en el usuario
- Hay dos metodologías ampliamente usadas
 - Casos de Uso (Use Cases)
 - Relatos de Usuario (User Stories)



2-10

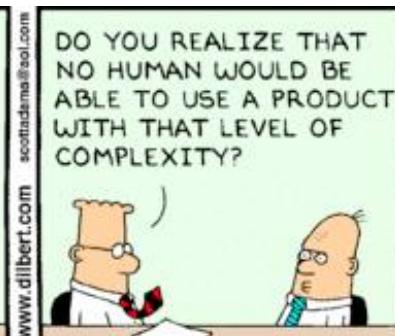
© 1996 Farcus Cartoonist, by Universal Press Syndicate

WAISGLASS/COULTHART



74777-3301 © compuserve.com

**"We've located the problem ...
it's a defective user."**



Relatos de Usuario

- documentos compartidos NO IMPLICA entendimiento compartido
- el objetivo de los relatos de usuario es el entendimiento compartido
- usar todos los medios para recordar la conversación

Un relato de usuario es como una foto de vacaciones



la foto tiene asociado un montón de detalles adicionales invisibles para quien no participó

Lecciones Importantes

- los relatos de usuario NO SON un nuevo formato de requerimientos escritos sino un mecanismo para adquirir un entendimiento compartido
- los relatos representan discusiones sobre solución de problemas de nuestros usuarios y clientes y que llevarán a acuerdos sobre qué construir
- el objetivo NO ES construir más software más rápido sino maximizar el outcome y el impacto

User Stories (Relatos de Usuario)

- Muy usada en procesos ágiles (Scrum)
- Idea es usar una “conversación” para capturar los requerimientos de usuario
- Puede ser interpretado como una promesa de conversación con el usuario
- Inicialmente se usaron tarjetas de 3x5 pulgadas (muy populares en USA)

Fundamentos tras la idea

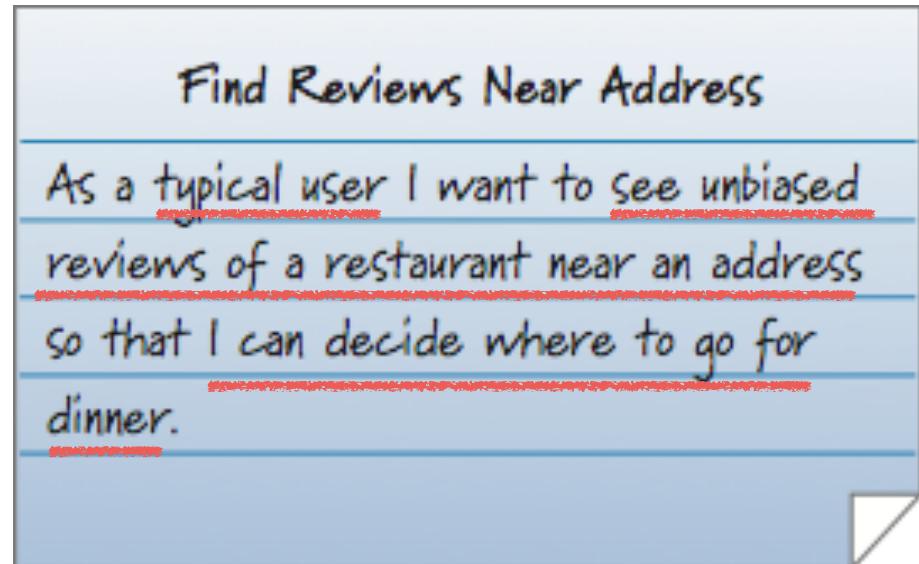
- requerimientos a través de colaboración
- requerimientos escritos y pasados en un documento tienen todas las decisiones tomadas
- con tiempos de entrega cortos es imposible especificar cada detalle de antemano
- responsabilidad de escribir correctamente los requerimientos se distribuye

US como Promesa de Conversación

- Las tarjetas son solo un punto de partida en el levantamiento de lo que se requiere
- User Story constituye una promesa de llevar a cabo una conversación (pueden ser varias) con los interesados
- resultados de la conversación podrían producir algunos documentos adicionales (detalles de la interfaz)

Card, Conversation, Confirmation

- user role - categoría de usuario
- goal - meta de usuario, lo que quiere conseguir



Confirmación

- En el reverso de la tarjeta se ponen las condiciones de satisfacción (SC)
- SC pueden ser expresados como un test de aceptación

Upload File

As a wiki user I want to upload a file to the wiki so that I can share it with my colleagues.

Conditions of Satisfaction

Verify with .txt and .doc files
Verify with jpg, gif, and png files
Verify with .mp4 files <= 1 GB
Verify no DRM-restricted files

As a customer, I want to sign-in using my Facebook account, so that I can easily access the service.

Daddy can you read me the story about princesses instead?



Backlog: Example corporate website

CSR2

Theme: CSR

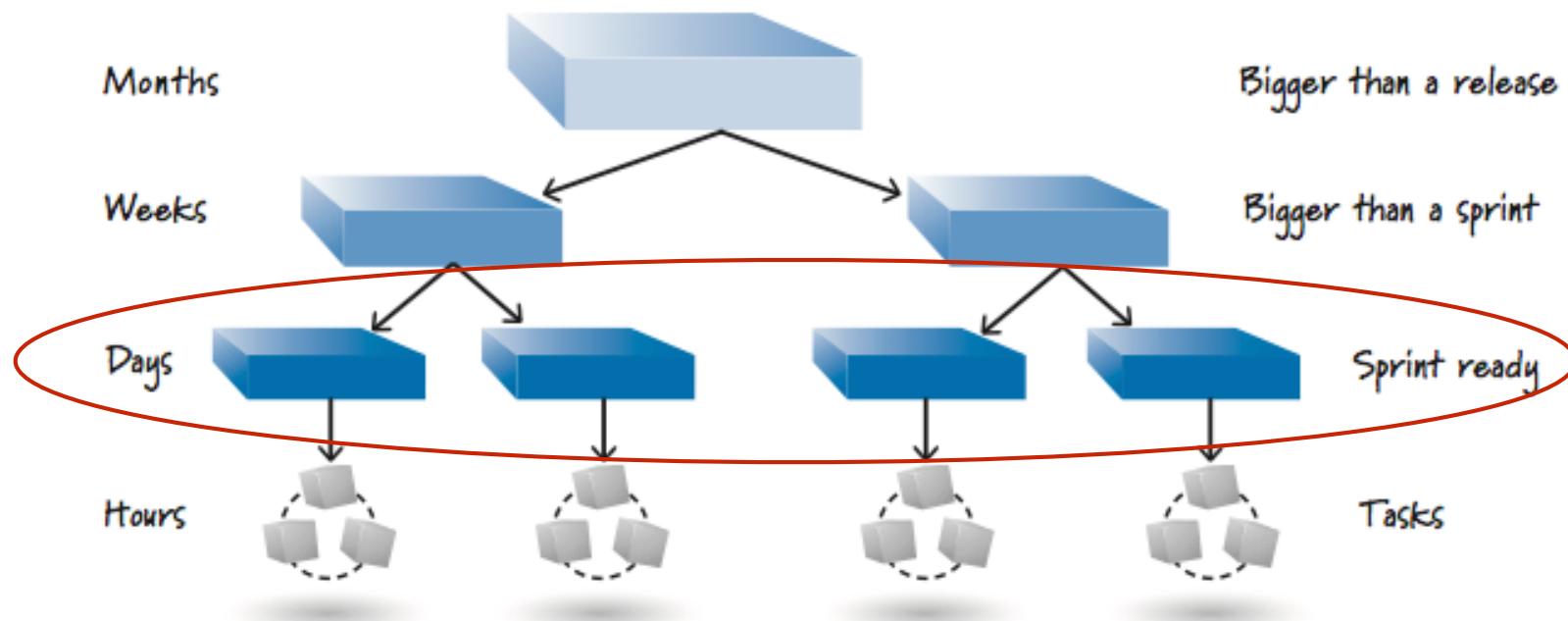
**As a front-end developer
I want to die
So I can end the pain**

Accesibility which

CSR2

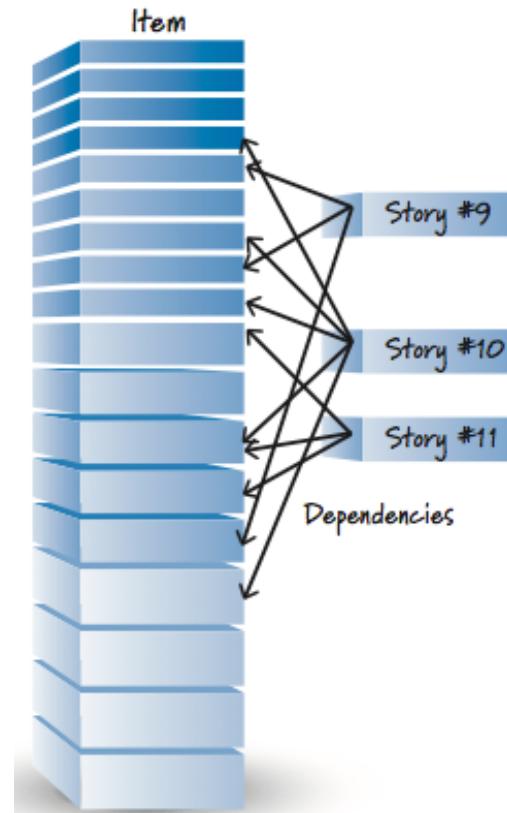
Nivel de Detalle

- pueden usarse varios niveles de abstracción
- nos interesan relatos a nivel de sprint



Ojalá sean independientes

- Se hace muy difícil priorizar si hay muchas dependencias entre las tareas
- tratar de escribir las tareas de modo que se minimicen las dependencias



Deben tener un valor para el cliente

- si hay una US que no es de valor para nadie no se debe hacer
- cuidado con cosas que son de valor para desarrolladores (valor técnico)
 - migrar código a última versión de Rails
- en esos casos es posible a veces reformularlo de forma que tenga un claro valor para el cliente
 - migrar código para mitigar riesgo

Estimable

- Si el equipo no es capaz de estimar el esfuerzo para desarrollar un US puede ser que
 - US sea muy grande
 - US sea ambiguo

¿Como escribir las historias ?

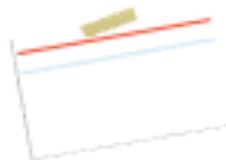
- incorporar a los usuarios al equipo en esta etapa
- workshop - brainstorming con participación de usuarios (al menos product owner)
- puede ser de unas horas hasta unos pocos días
- la idea es generar un set inicial y no un set completo
- comenzar identificando roles de las historias para poder escribir “As a <user role>, I want to ...”
- top-down o bottom-up

Algunos vicios

- tratar de evitar el genérico "as a user" (ser más específico)
- US es muy pequeño
 - resultados deben guardarse como archivo XML
 - resultados deben guardarse como HTML
- US con interdependencias (difícil planeación)
- Goldplating (haciendo más de lo necesario)
 - además de tabs se pueden arrastrar
- Demasiado detalle (por eso se usan tarjetas pequeñas)
- Detalles de interfaz de usuario demasiado pronto
- traspasar requisitos formales a relatos de usuario

Relatos vs Documentos de Requisitos

User stories



Specifications & requirement docs



- Lean, accurate, just-in-time
- Encourage face-to-face communication
- Simplified planning
- Cheap, fast, easy to create
- Never out-of-date
- Based on latest learnings
- Enable real-time feedback
- Avoid false sense of accuracy
- Allow for team-based collaboration and innovation

- Heavy, inaccurate, out-of-date
- Encourage guesswork (false assumptions)
- Complex planning
- Expensive, slow, hard to create
- Always out-of-date
- Based on little or no learning
- Disable real-time feedback
- Promote false sense of accuracy
- Discourage open collaboration and innovation

El propósito no es el software !!

- el software es el *output* del proceso
- lo que nos interesa realmente es el *outcome*
- el outcome es el resultado para los usuarios
 - podrán lograr sus metas con mayor facilidad ?
 - hemos mejorado en algo sus vidas ?
 - hay un impacto ? en qué usuarios ?
- un buen relato no es solo acerca de qué sino de quien y por qué

Maximizar Outcome

- Siempre habrá más que construir que el tiempo y los recursos disponibles
- Es un error intentar maximizar el output (más features, producto más completo)
- Debemos apuntar a maximizar el outcome y el impacto ojalá minimizando el output
- Implica escuchar cuidadosamente a la gente cuyo problema tratamos de resolver

Es posible dejar a la gente correcta muy contenta
incluso construyendo una pequeña
fracción de lo que estaba inicialmente planeado