

# Introduction aux Méthodes « AGILE » CE+T



Bruno Sбилle – 21/06/2013

# Principes de ce Workshop

- **Horaires**
- **Questions (et le parking)**
- **Vocabulaire Français vs English**
- **Après la journée**
- **Téléphones ? Éteints, « silence », vibreurs ?**
- **Votre Participation**



# Scrum, Agile & Lean in Belgium

Lessons Learned using Lean and Agile methods

Bruno Sbille: Coach Agile & Formateur  
[bruno.sbille@gmail.com](mailto:bruno.sbille@gmail.com)  
+32 491 05 05 59  
@BrunoSbille  
Blog Agile: <http://brunosbille.com>



[Blog](#) [Coaching & Training](#) [What is Scrum ?](#) [Contact Bruno](#)



## Agile Games Night II - Brussels - 6th of June



Let's play Agile Games

As you may know, every month, the Agile

17 9  
[Tweet](#) [Like](#)

Belgium Community organizes [Drinkups](#), See [here](#) for more details. During the drinkups of February, several participants told us they would like to have an evening with an agenda, preferably on the topic of Agile Games.

In March we decided to relaunch the

## Choose your Language

English

Français

## Agile2013



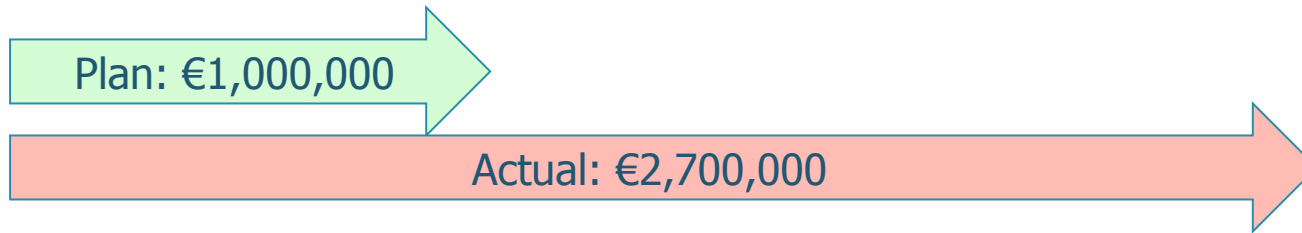
# **ORIGINE DES MÉTHODES AGILE**



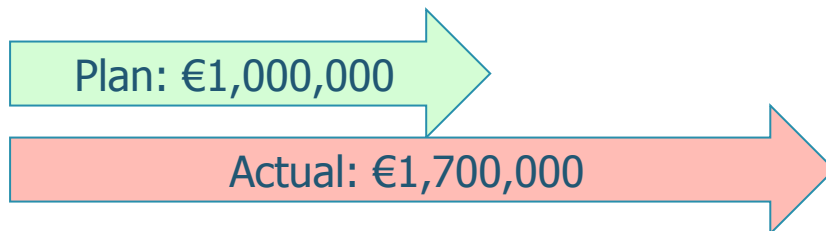
# La plupart des projets IT échouent

Le Standish Group a étudié plus de 40.000 projets en 10 ans.

Taux de réussite des projets IT en 1994: **15%**  
Dépassement moyen (coût et temps) : **≈170%**



Taux de réussite des projets IT en 2004: **34%**  
Dépassement moyen (coût et temps) : **≈70%**

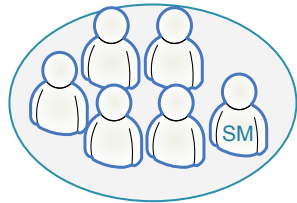


**Sources:**

<http://www.softwaremag.com/L.cfm?Doc=newsletter/2004-01-15/Standish>  
<http://www.infoq.com/articles/Interview-Johnson-Standish-CHAOS>

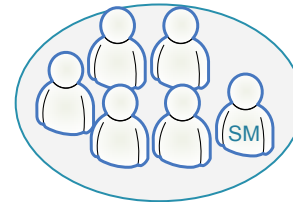
# Comment les estimations sont affectées par la longueur des spécifications

Spec



117 hrs

Spec identiques – plus de pages



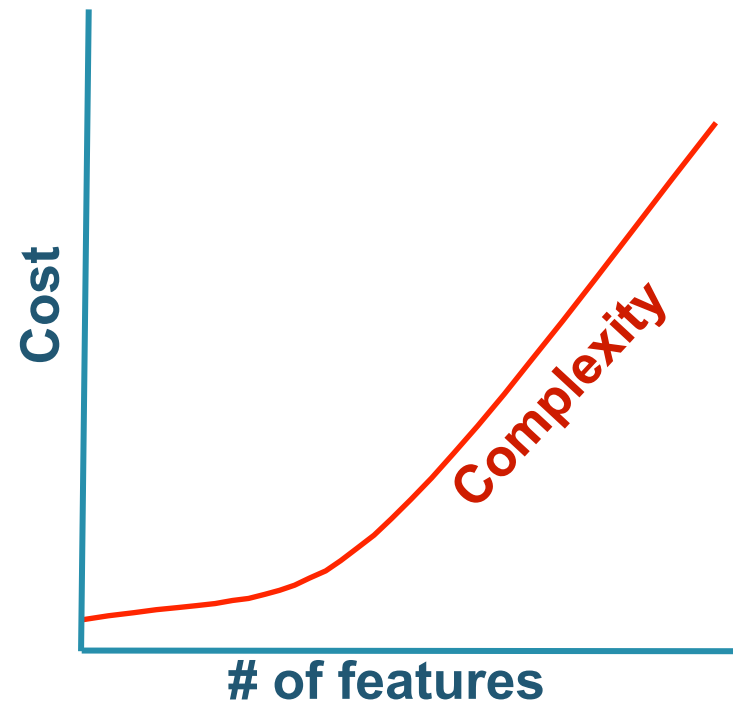
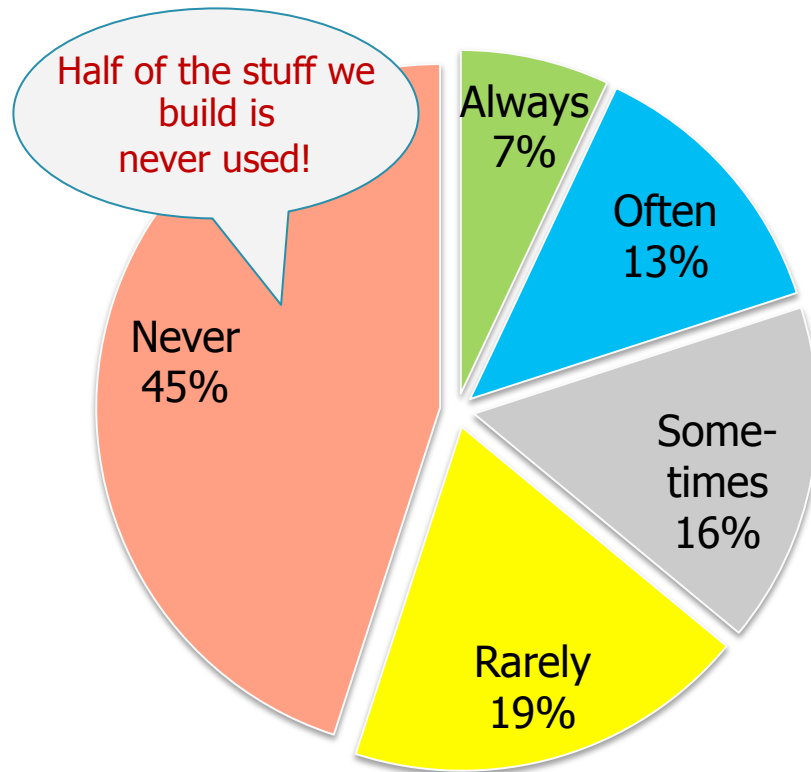
173 hrs

Source: How to avoid impact from irrelevant and misleading info on your cost estimates, Simula research labs estimation seminar, Oslo, Norway, 2006

2007-09-28

# We tend to build the wrong thing

Features and functions used in a typical system



Sources:  
Standish group study reported at XP2002 by Jim Johnson, Chairman

This graph courtesy of Mary Poppendieck



# Le top 5 des raisons du succès

1. Impliquer l'utilisateur
2. Support du Senior Management
3. Objectifs Business Clairs
4. Optimiser le Scope du projet
5. Processus Agile

**Sources:**

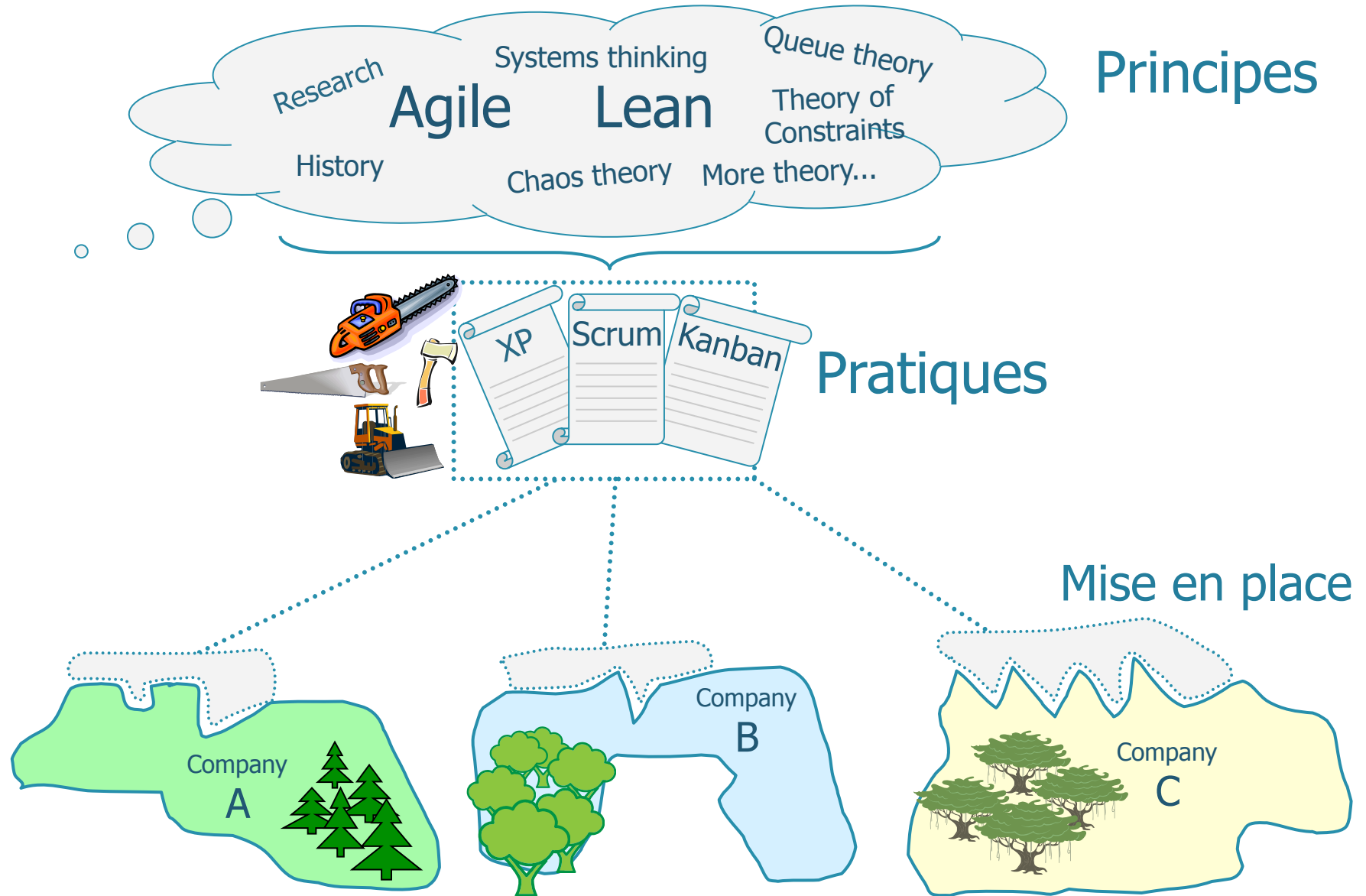
<http://www.softwaremag.com/L.cfm?Doc=newsletter/2004-01-15/Standish>

<http://www.infoq.com/articles/Interview-Johnson-Standish-CHAOS>

"My Life is Failure", Jim Johnson's book

# AGILE

# Dessine moi Agile...





# Manifeste Agile

[www.agilemanifesto.org](http://www.agilemanifesto.org)

Nous découvrons comment mieux développer des logiciels par la pratique et en aidant les autres à le faire.  
Ces expériences nous ont amenés à valoriser:

**Les individus et leurs interactions** plus que **les processus et les outils**

**Des logiciels opérationnels** plus **qu'une documentation exhaustive**

**La collaboration avec les clients** plus que **la négociation contractuelle**

**L'adaptation au changement** plus que **le suivi d'un plan**

Nous reconnaissons la valeur des seconds éléments,  
mais privilégions les premiers.

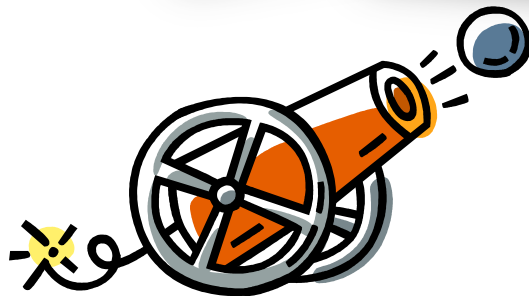
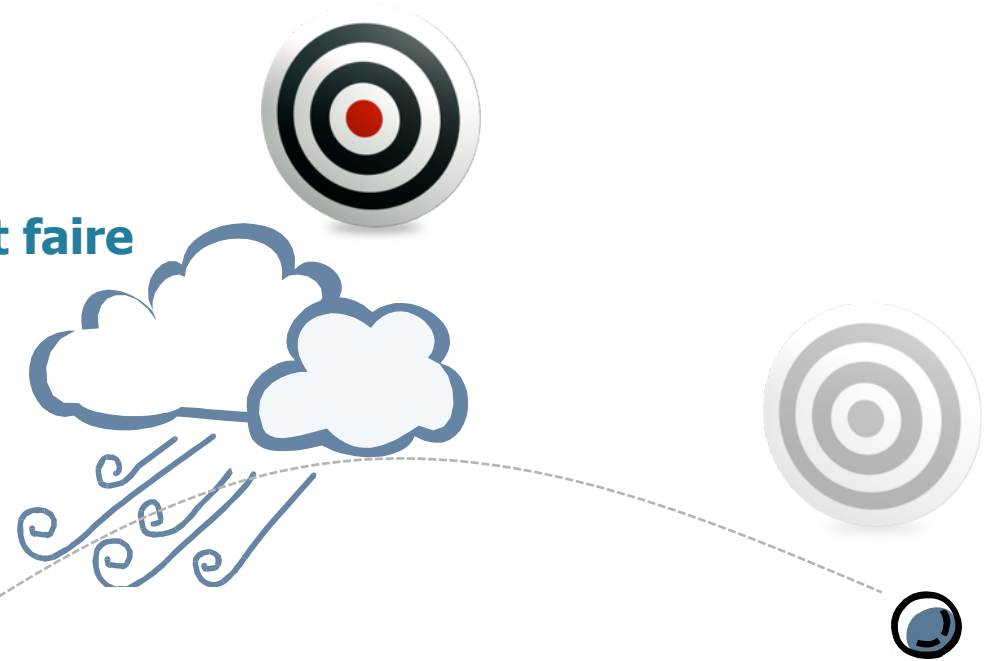
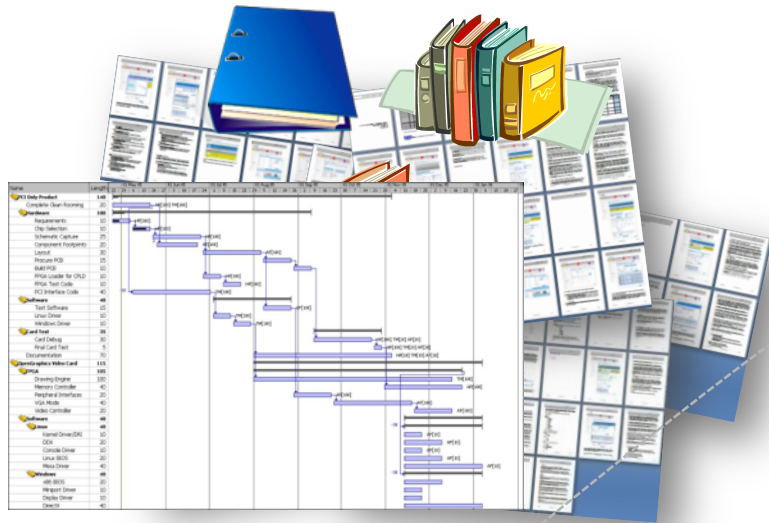
# Principes sous-jacents au Manifeste Agile

- Notre plus haute priorité est de **satisfaire le client** en livrant rapidement et régulièrement des fonctionnalités à grande valeur ajoutée.
  - **Accueillez positivement les changements de besoins**, même tard dans le projet. Les processus Agiles exploitent le changement pour donner un avantage compétitif au client.
  - **Livrez fréquemment un logiciel opérationnel** avec des cycles de quelques semaines à quelques mois et une préférence pour les plus courts.
  - **Les utilisateurs ou leurs représentants et les développeurs doivent travailler ensemble** quotidiennement tout au long du projet.
  - Réalisez les projets avec des **personnes motivées**. Fournissez-leur l'environnement et le soutien dont ils ont besoin et faites-leur **confiance** pour atteindre les objectifs fixés.
  - La méthode la plus simple et la plus efficace pour transmettre de l'information à l'équipe de développement et à l'intérieur de celle-ci est le **dialogue en face à face**.
- Un **logiciel opérationnel** est la principale mesure d'avancement.
  - Les processus Agiles encouragent un **rythme de développement soutenable**. Ensemble, les commanditaires, les développeurs et les utilisateurs devraient être capables de maintenir indéfiniment un rythme constant.
  - Une attention continue à **l'excellence technique** et à une **bonne conception** renforce l'Agilité.
  - La **simplicité** – c'est-à-dire l'art de minimiser la quantité de travail inutile – est essentielle.
  - Les meilleures architectures, spécifications et conceptions émergent d'équipes **auto-organisées**.
  - À intervalles réguliers, l'équipe **réfléchit aux moyens de devenir plus efficace**, puis règle et modifie son comportement en conséquence.

# Les projets IT traditionnels, c'est comme un boulet de canon

Hypothèses (in)conscientes:

- Le Client sait ce qu'il veut
- Les développeurs savent comment faire
- Il n'y aura aucun changement



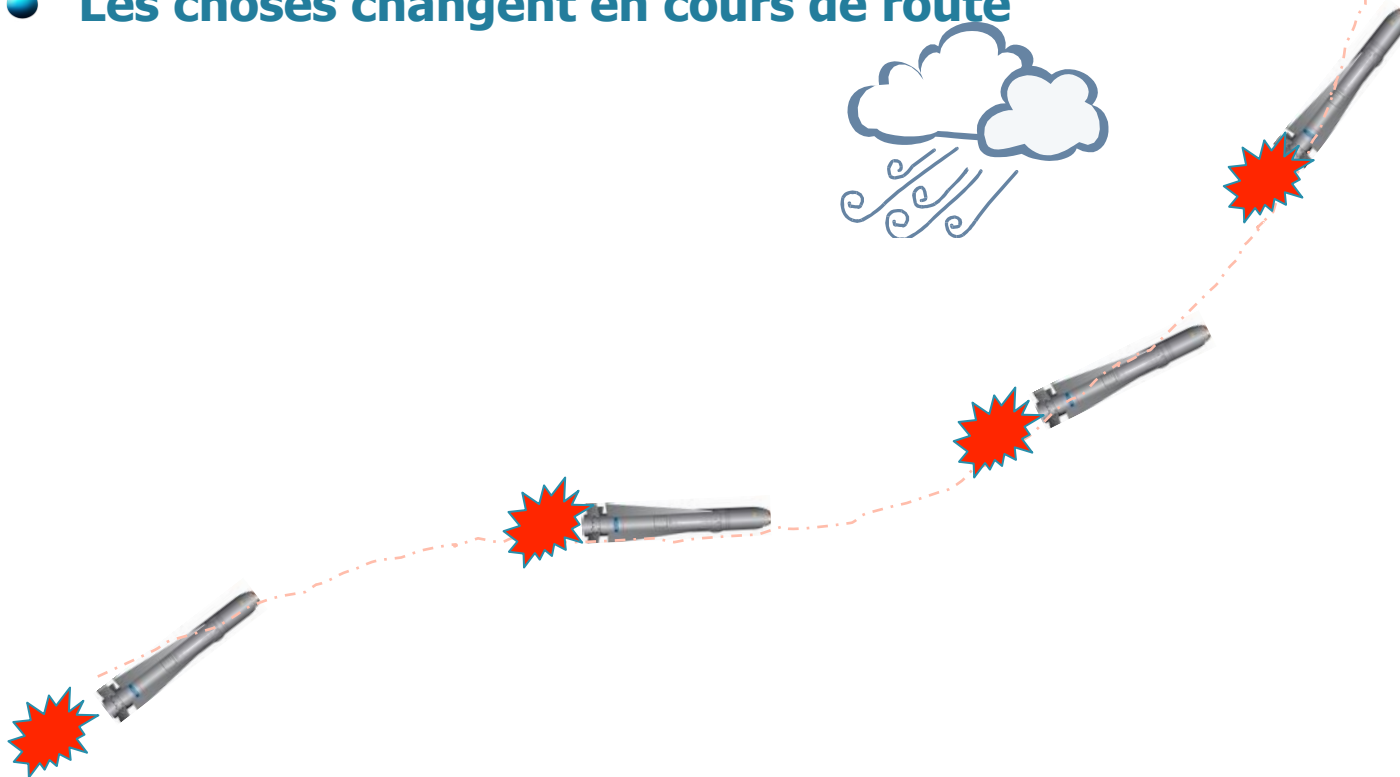




# Les projets Agile = Missile à tête chercheuse

## Hypothèses conscientes:

- Le client découvre ce qu'il veut au fur et à mesure
- Les développeurs découvrent comment le construire
- Les choses changent en cours de route



# Gestion du temps

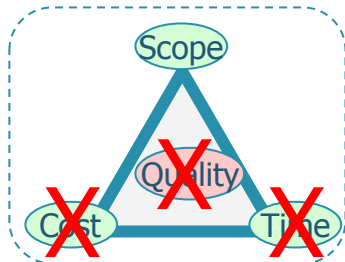
## Plan

(voué à l'échec, mais on ne le sait pas encore)



## Scénario traditionnel

"Nous allons délivrer ABCD en 4 semaines"



Oups, on est en retard...

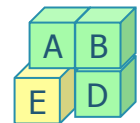
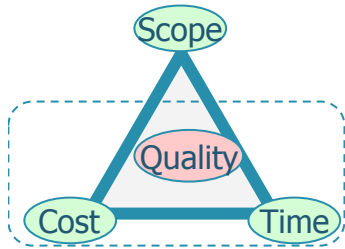


## Scénario Agile

"On délivre à chaque fois quelque chose, et ce à chaque sprint (4 semaines)"

"On pense que l'on peut terminer ABCD en 1 sprint, mais nous ne sommes pas sûr"

"On délivre toujours les fonctionnalités les plus importantes en premier"



Oups, on a seulement fini AB, notre vélocité est plus basse que l'on pensait. Que doit-on faire maintenant ?

# SCRUM

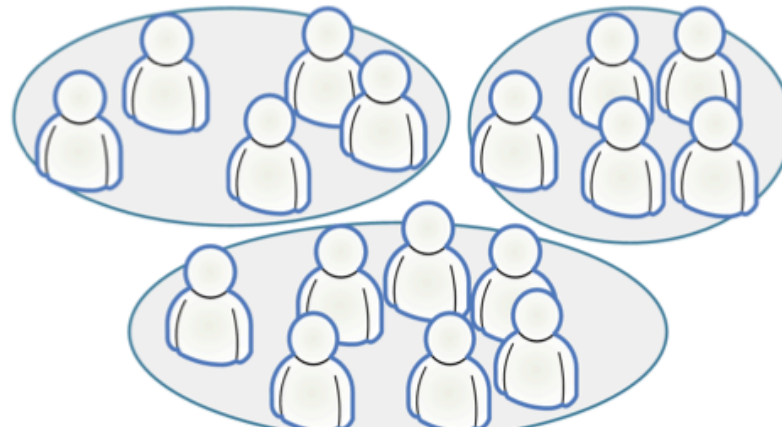


A photograph of two men in white lab coats in a laboratory. The man on the right is holding a small glass vial to his nose, smelling its contents. The man on the left is looking at the vial. In the foreground, there are two Erlenmeyer flasks containing a dark brown liquid with a white head of foam, sitting on a white scale. To the right, a larger flask contains a clear, pale yellow liquid. The background shows laboratory equipment and a tiled ceiling.

# Expérimentez et Adaptez-vous

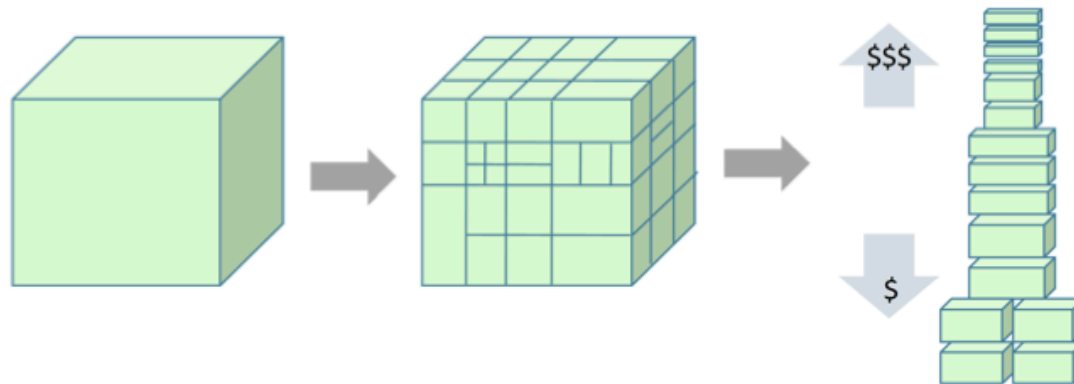
# SCRUM ? (1/4)

- **Divisez votre organisation** en petites équipes multidisciplinaires et auto-organisées.



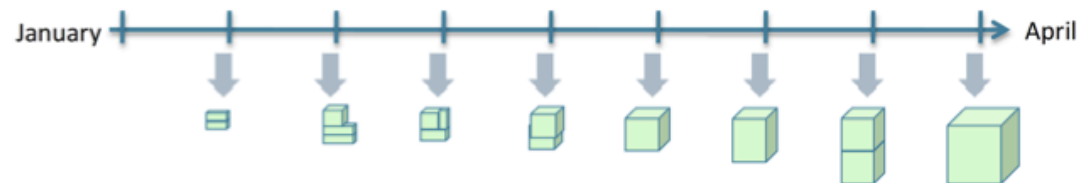
# SCRUM ? (2/4)

- **Divisez votre travail** en une liste de petits livrables concrets. Triez cette liste par priorité et estimez la taille relative de chaque élément.



# SCRUM ? (3/4)

- **Divisez le temps** en petites itérations de durée fixe (appelées des sprints et durant habituellement de 1 à 4 semaines) et faites une démonstration à l'issue de chaque sprint avec un produit potentiellement livrable.





# SCRUM ? (4/4)

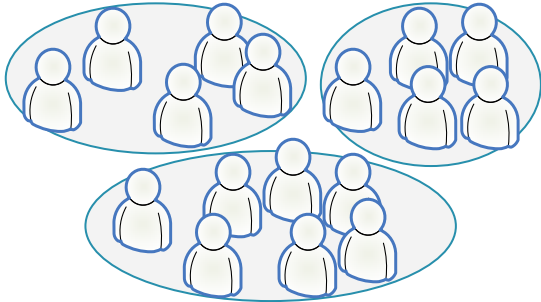
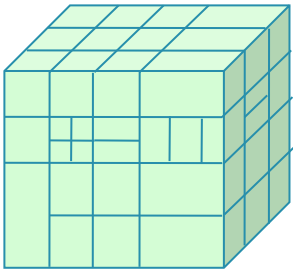
- **Optimisez le planning de la version** et mettez à jour les priorités en collaboration avec le client, sur la base de ce que vous avez appris après chaque sprint.
- **Optimisez le processus** en organisant une rétrospective après chaque sprint.

Ainsi, au lieu d'avoir un **grand groupe** passant **beaucoup de temps** sur la construction d'une **grande chose**, nous avons une **petite équipe** passant **un peu de temps** à construire une **petite chose...** mais **intégrant régulièrement** pour voir l'ensemble.

# Divisez votre organisation en équipes

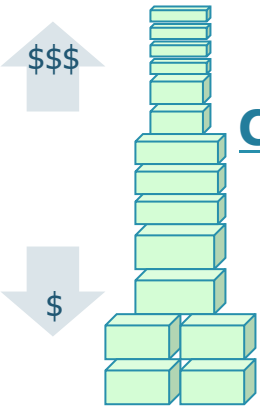
# Scrum en bref

## Divisez votre produit

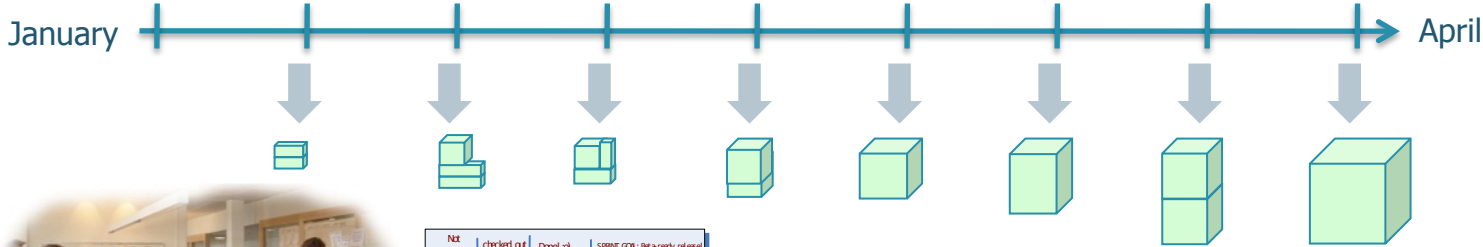
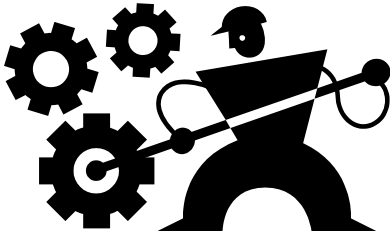


~~Un grand groupe~~ passant ~~beaucoup de temps~~ à construire un ~~gros truc~~  
Une petite équipe passant un peu de temps à construire une petite chose  
... Mais en intégrant rapidement pour voir l'ensemble

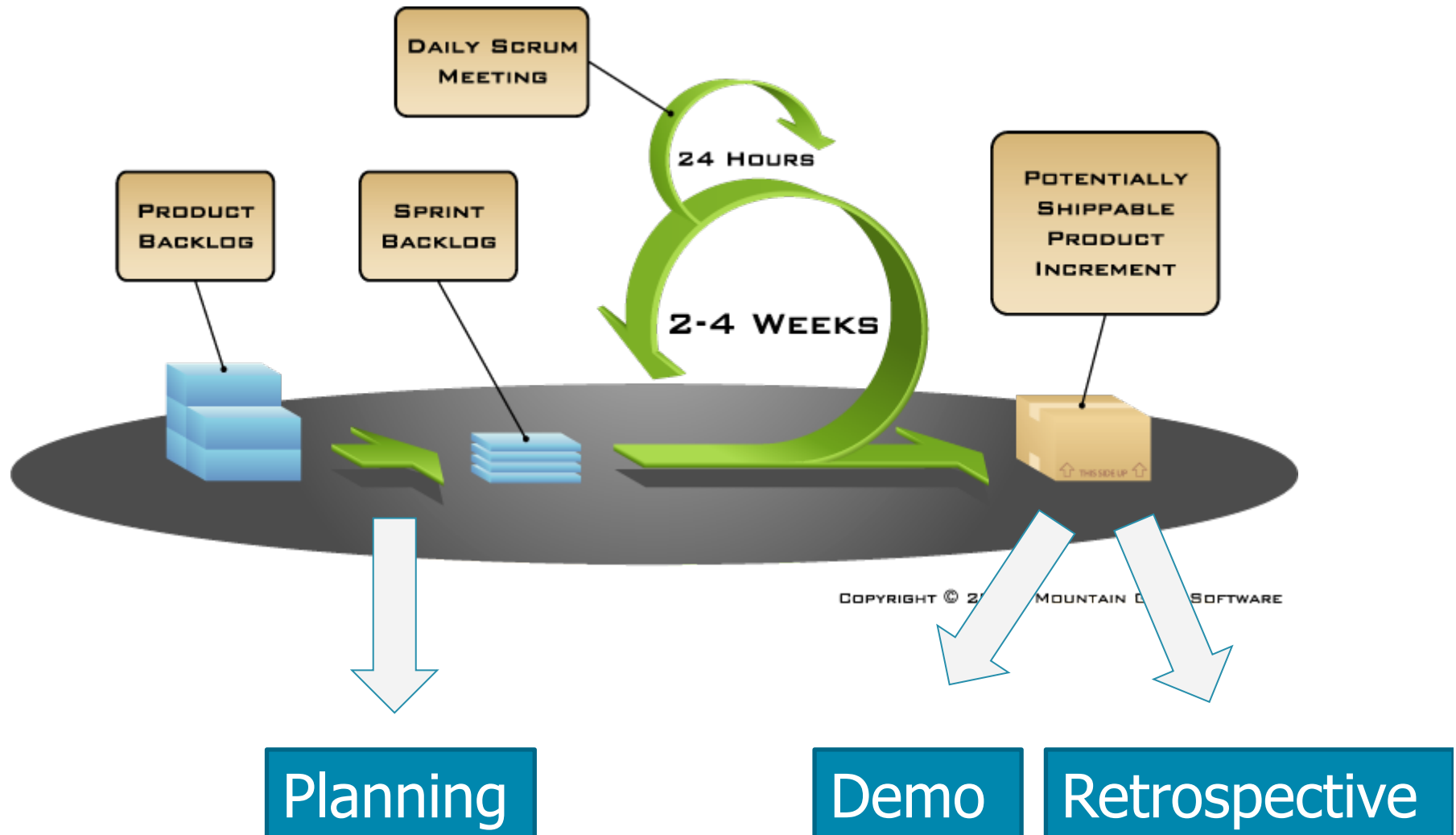
## Optimisez la valeur business



## Court cycles de temps

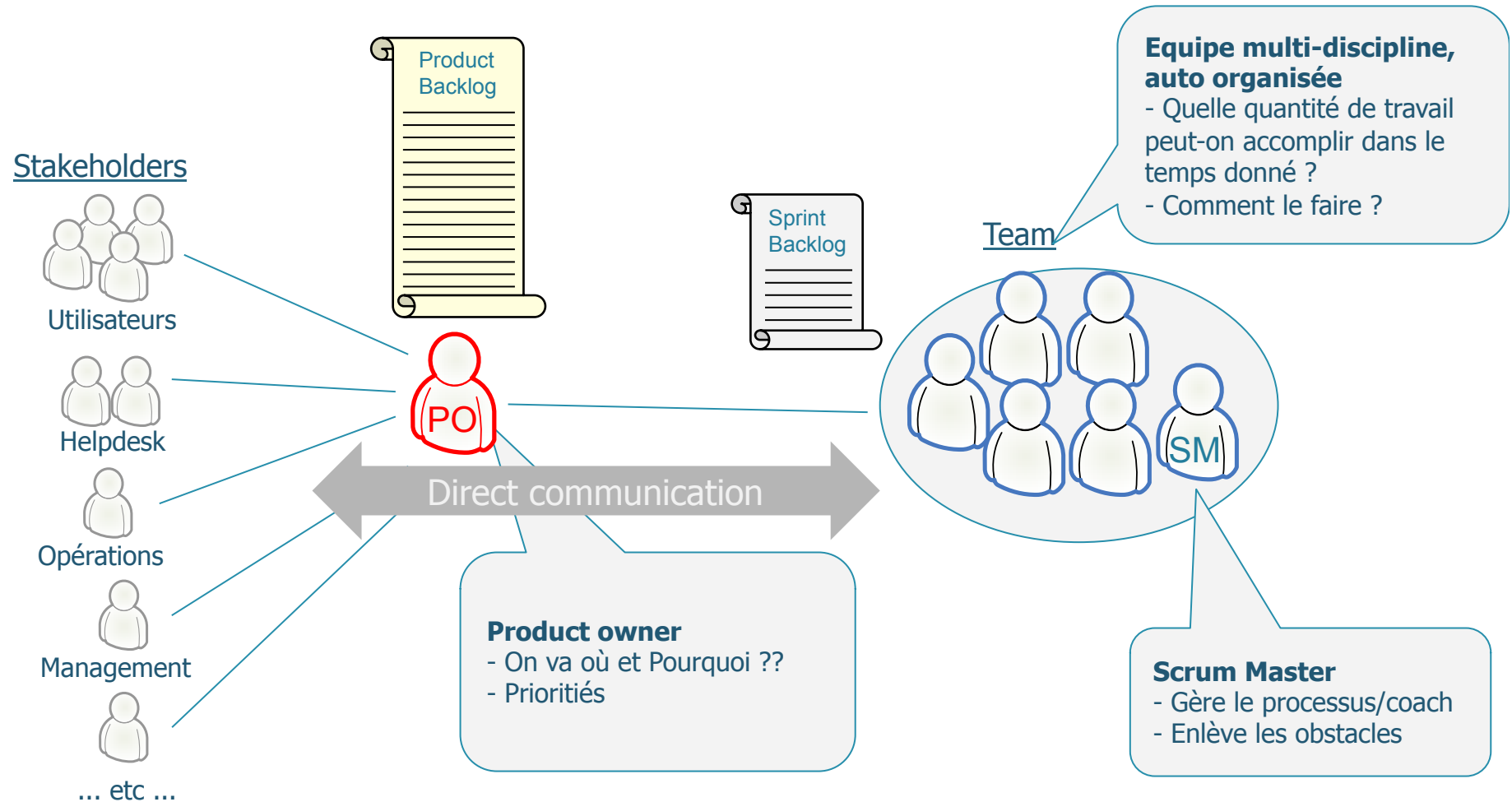


# Scrum en Bref: les différentes réunions et les différents documents





# Scrum aperçu des différents rôles



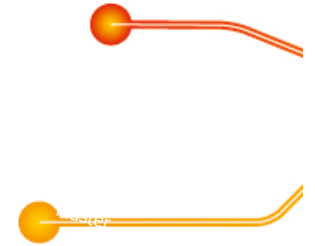
# Le Team member

- C'est lui qui contribue à la réalisation du projet
- Orienté multi-casquettes
- Polyvalent
- Autonome
- Auto-organisés

# The Team Member

## Roles

- **Cross-functional:**  
Programmers, testers, user experience designers, etc.
- **Members should be full-time**  
May be exceptions (e.g., database administrator)
- **Teams are self-organizing**
- **Ideally, no titles but rarely a possibility**
- **Membership should change only between sprints**
- **Scrum Master & Product Owner are part of the team**



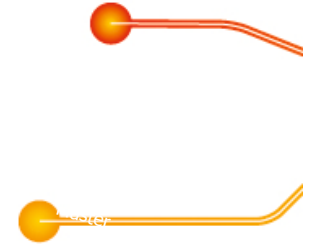
# Le Scrum Master

- est responsable pour l'application de la méthode
- son job: permettre à l'équipe d'avoir ce qu'il faut pour travailler
- accompagne l'équipe, la «coache»
- Protège l'équipe des interférences
- Aide à la résolution des problèmes
- Connaît très bien Scrum

# The Scrum Master 1

## Role

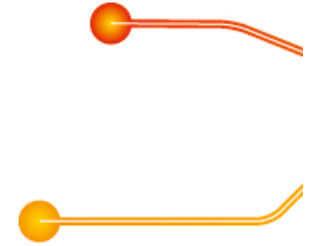
- Represents management to the project
- Responsible for enacting Scrum values and practices
- Removes impediments
- Ensure that the team is fully functional and productive
- Enable close cooperation across all roles and functions



# The Scrum Master 2

## Attributes

- **Responsible**
- **Humble**
- **Collaborative**
- **Committed**
- **Influential**
- **Aware of Business & IT side**





# Le Product Owner

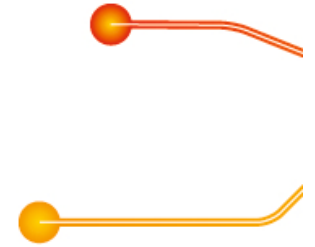
- Donne la direction
- Définit les priorités
- Valide au fur et à mesure
- Fait partie intégrante de l'équipe
- Peut aider à la résolution des problèmes



# The Product Owner 1/2

## Role

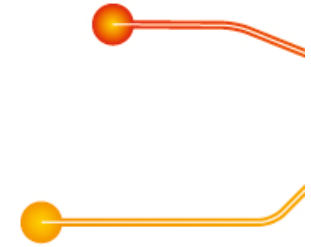
- **Define the features of the product**
- **Decide on release date and content**
- **Be responsible for the profitability of the product (ROI)**
- **Prioritize features according to market value**
- **Adjust features and priority every iteration, as needed**
- **Accept or reject work results**



# The Product Owner 2/2

## Attributes

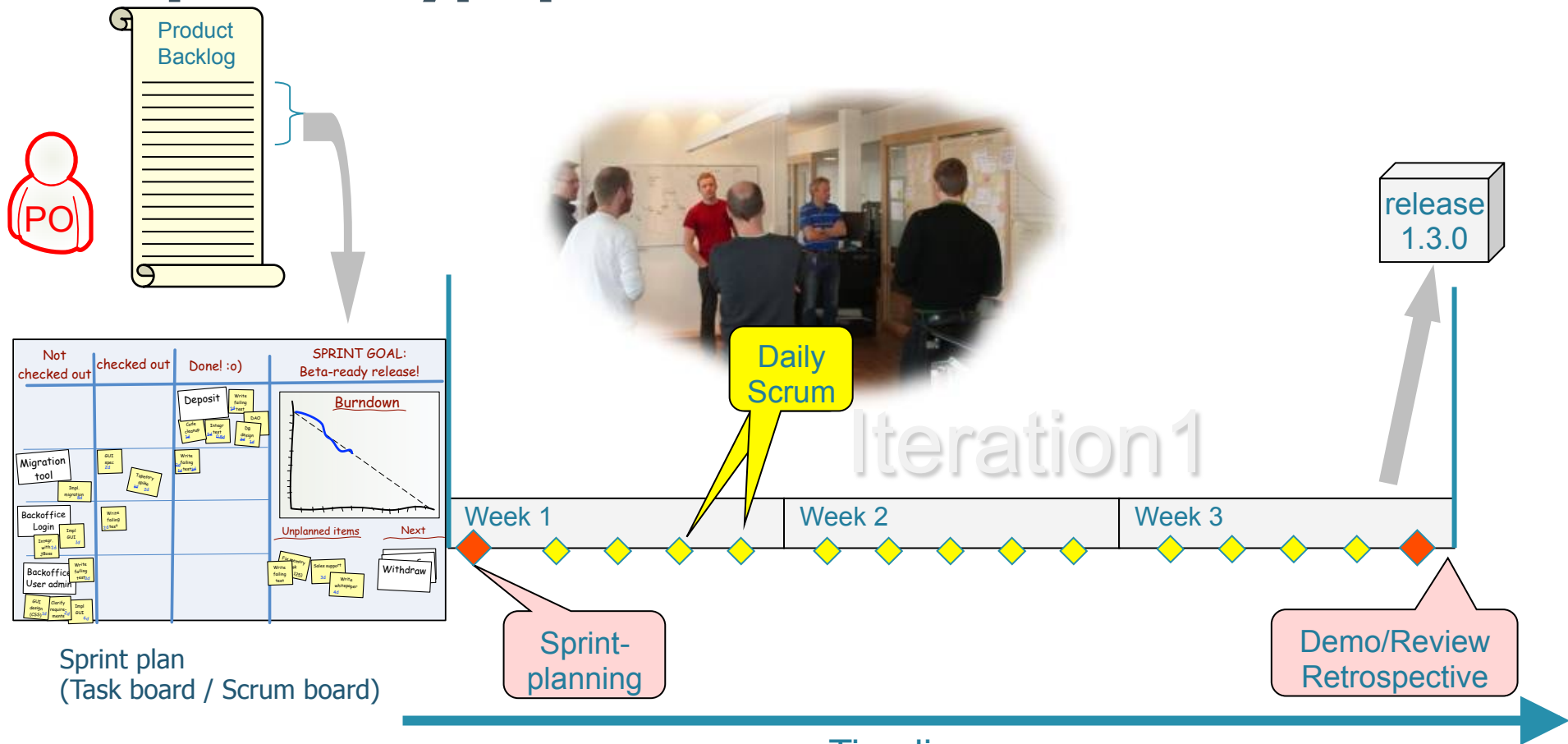
- **Available**
- **He/she understands the Business (Product Manager, Marketing, Business Analyst)**
- **Good Communicator**
- **Decisive**
- **Empowered (has mandate)**



# Les stakeholders

- les clients, le manager, toute personne impliquée mais non engagée
- peuvent être présent à la démo
- peuvent être présent de manière silencieuse à la daily meeting
- souhaitent avoir une vue sur le projet sans s'investir profondément dedans

# Un sprint "Typique"



Not checked out	checked out	Done! :o)	SPRINT GOAL: Beta-ready release!
		Deposit	<p>Unplanned items</p> <p>Next</p>
Migration tool			
Backoffice Login			
Backoffice User admin			Withdraw

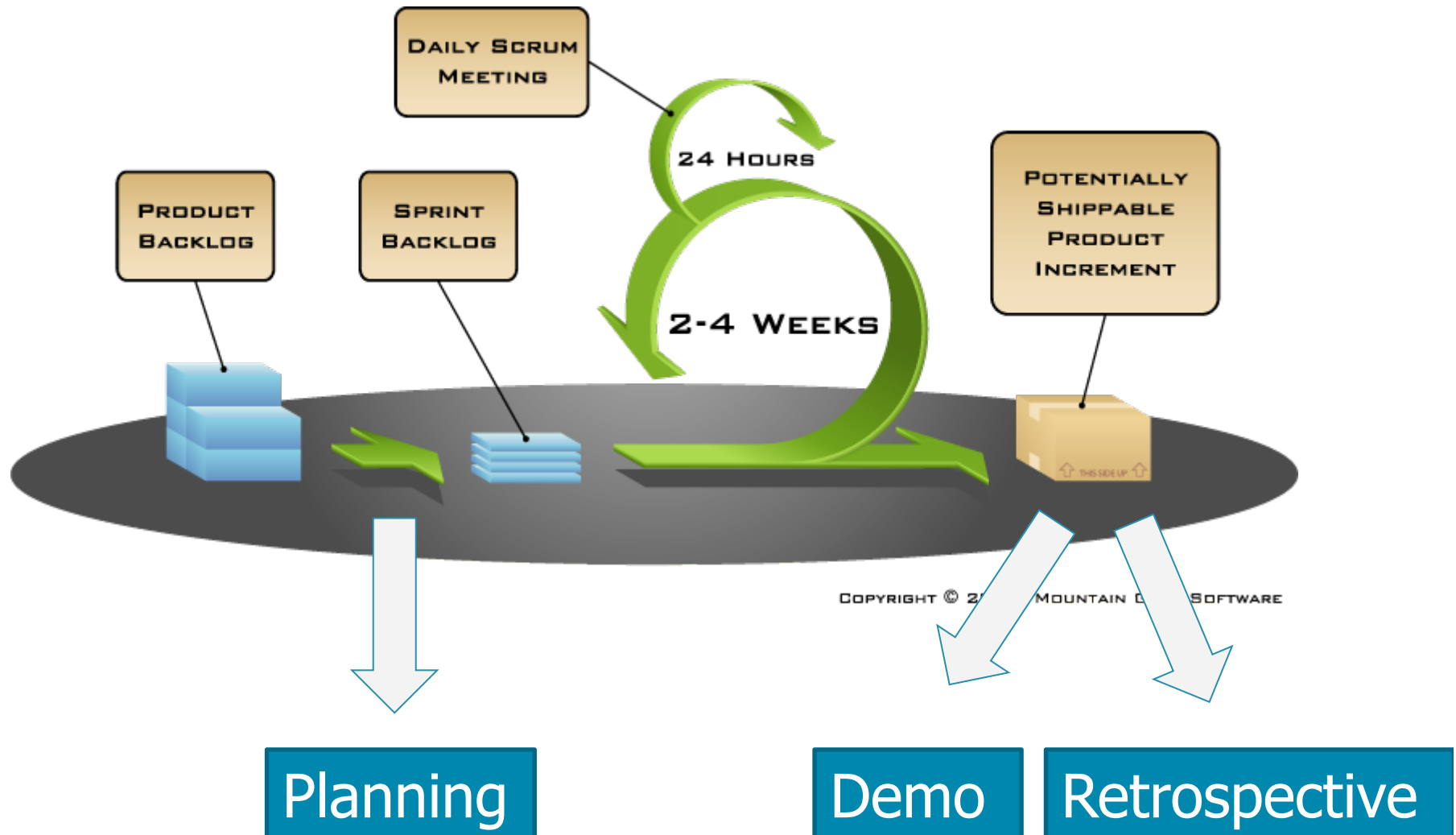
Sprint plan  
(Task board / Scrum board)



**Zoom sur les produits et résultats**

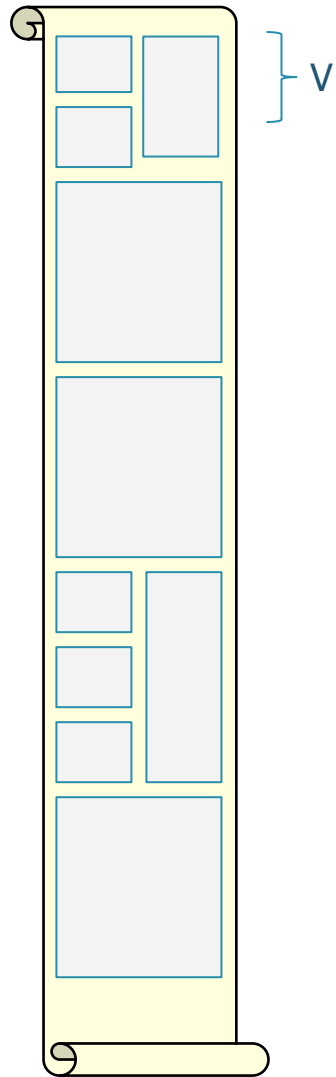
**SCRUM**

# Scrum en Bref: les différentes réunions et les différents documents

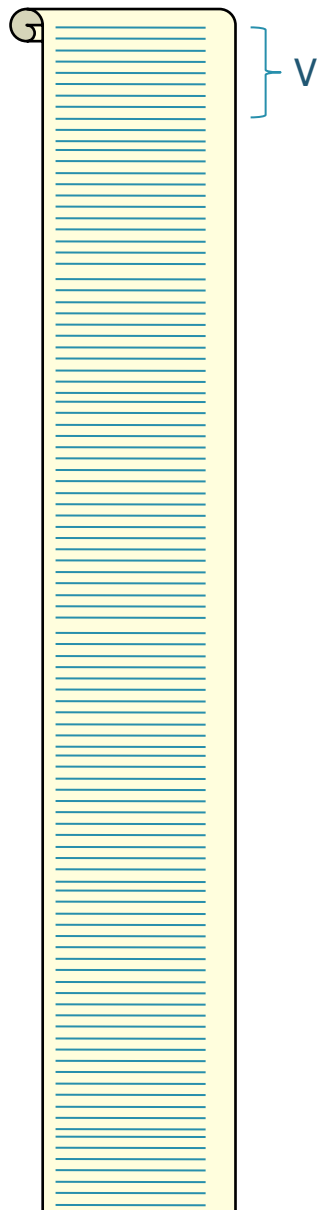


# Une juste balance pour product backlog

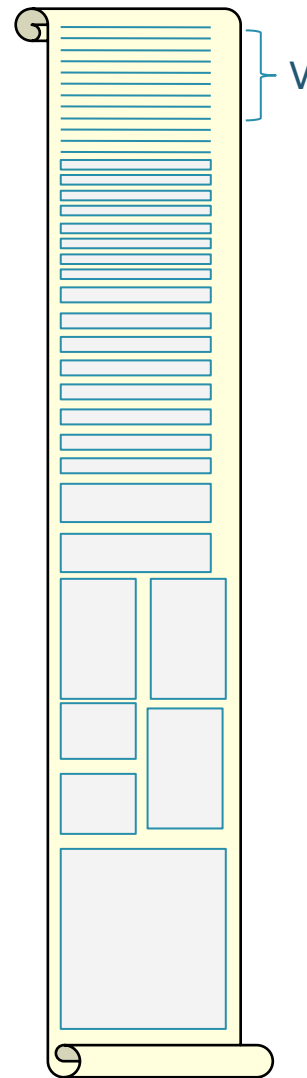
Too big items?



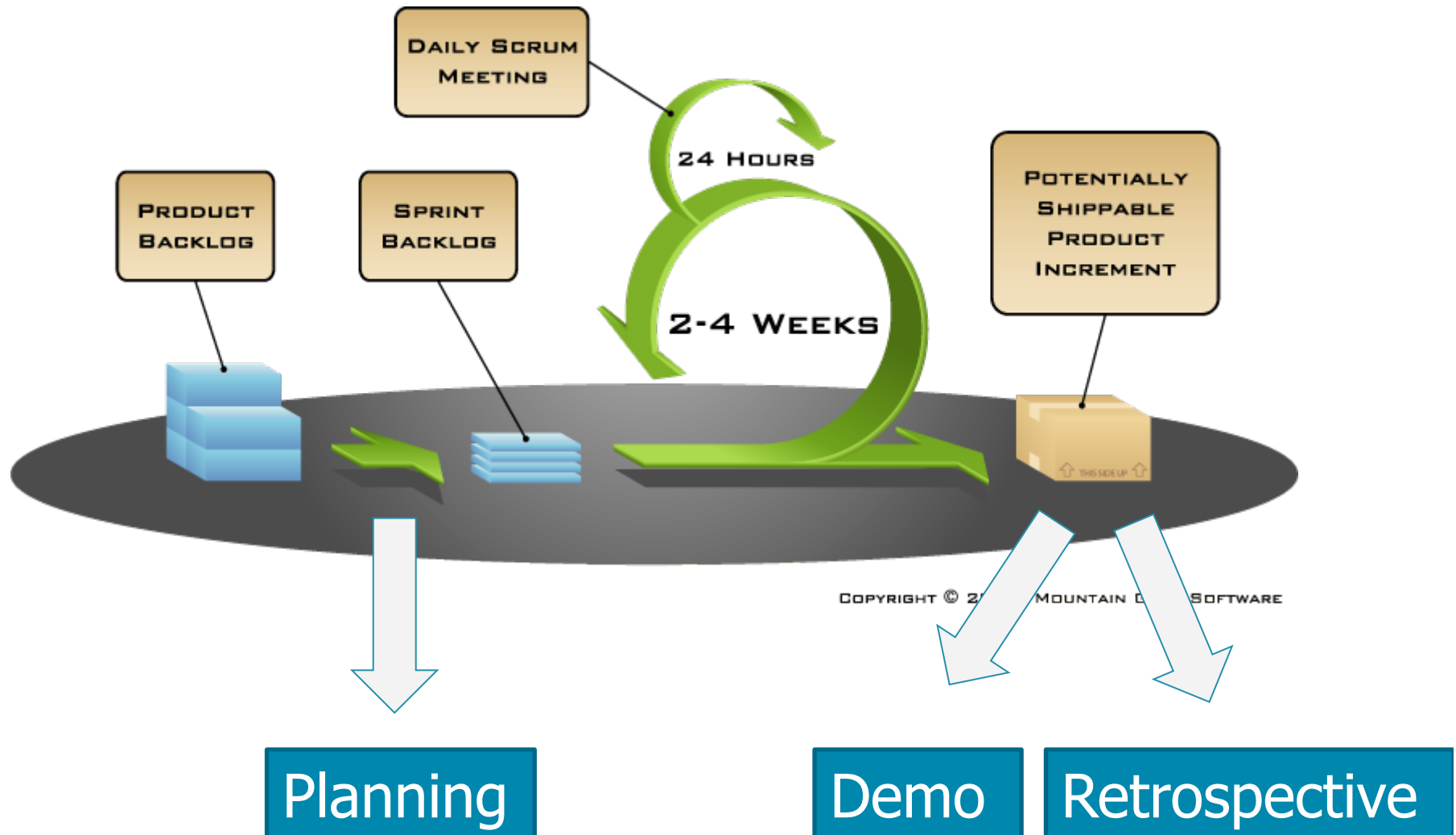
Too many items?



Balanced



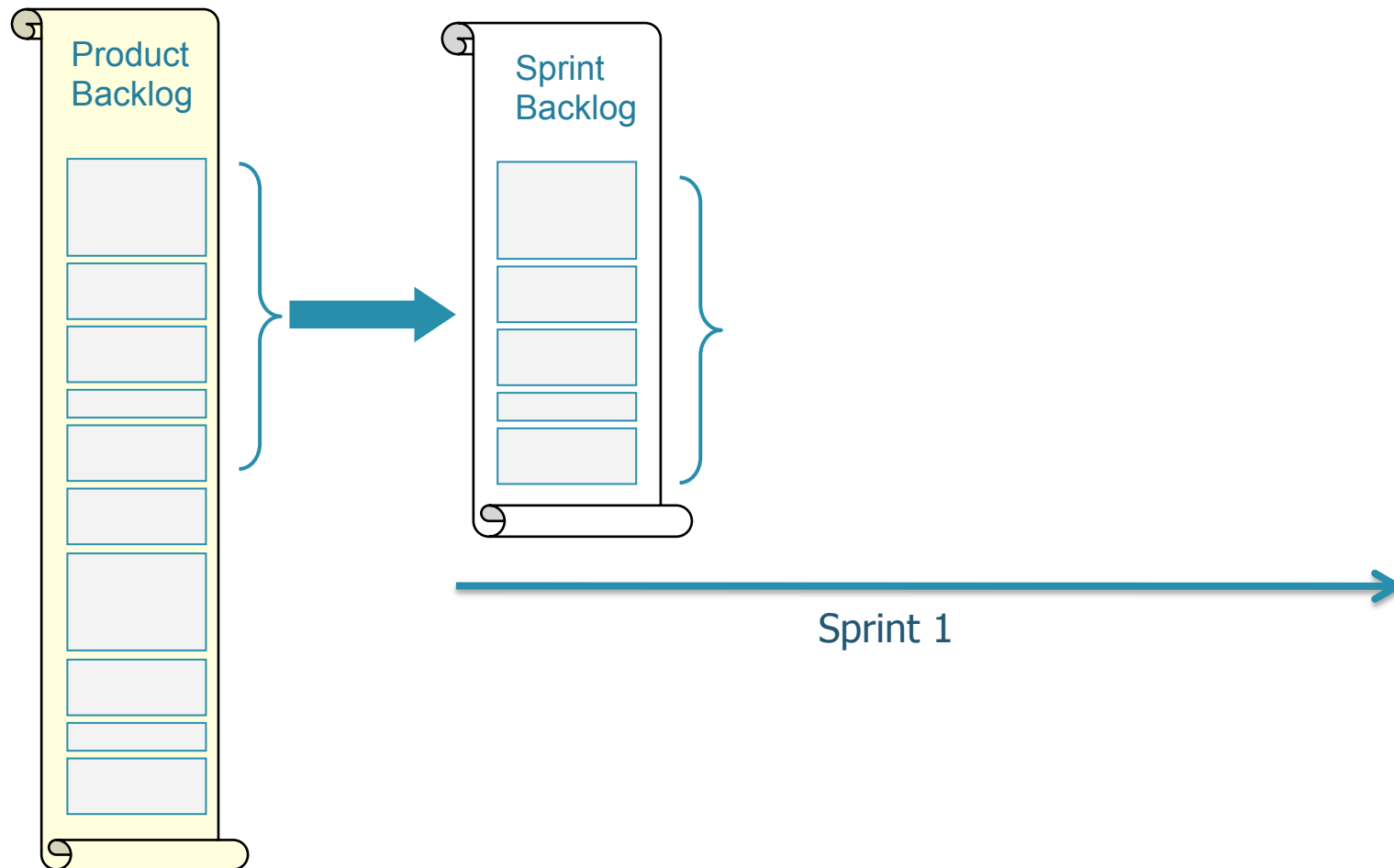
# Scrum en Bref: les différentes réunions et les différents documents





# Le Product Backlog et le Sprint Backlog

Début du sprint 1

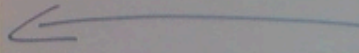


# Sprint Planning

LE MOINS

IMPORTANT

(A) PRIORITÉS



SPRINT

CONFIGURATION DE  
L'OUTIL  
(CONFIG, ADMIN...)

CHOIX  
OUTIL DE  
BUILD

PRENDRE LE FB  
CHEZ DANNY  
PATRICK.

INSTALLATION

OUTIL DE

TESTS D'ACCEPTANCE

CHOIX

OUTIL DE

TESTS D'ACCEPTANCE

INSTALLATION DE  
LA PLATEFORME

ACCORD SUR

LA NOMENCLATURE

+ Valeurs XP  
+ principes SAUJ

PRISE EN MAIN  
DE L'OUTIL.

(A) COMPREHENSION DE  
L'OUTIL: VAC COMMUN  
+ Bibliographie  
PRINCIPES

PASSER D'UN ENV.

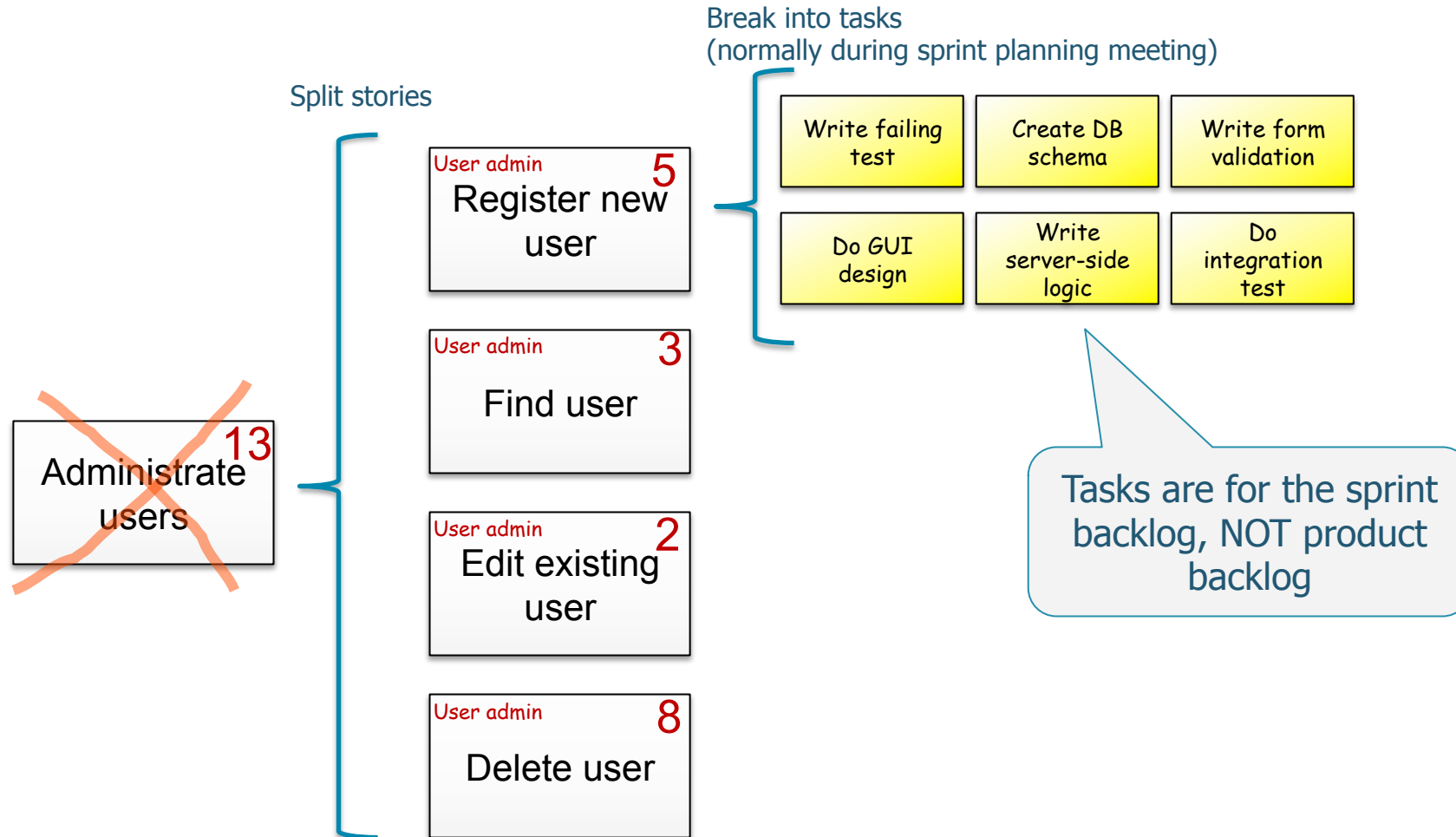
A L'AUTRE D'UNE MANIERE

ADHOC RAPIDE ET  
FIDÈLE

MIN  
AFIN DE ... TPS

INSTALLATION DE  
L'OUTIL DE  
BUILD

# Splitting stories and breaking out tasks



# Le Sprint

- Pas de changement de Scope pendant le sprint
- entre 1 et 4 semaines
- Daily Scrum
- Pas de changement de Scope pendant le sprint
- Liberté d'organisation



# Le Daily Scrum



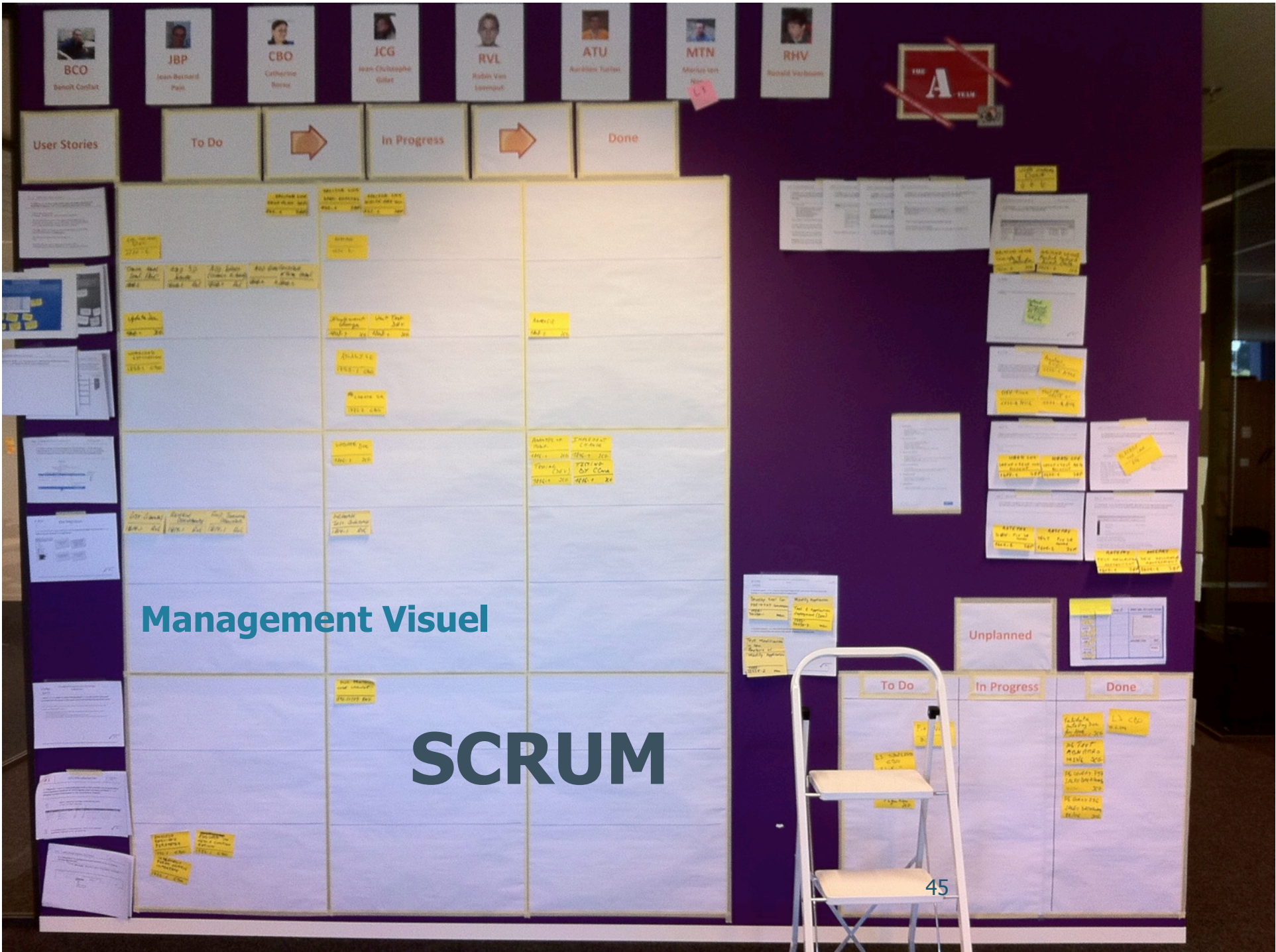
SCRUM



# La Daily Meeting

- Debout
- min 15 minutes
- (I) Ce que j'ai fait depuis la dernière meeting
- (II) Ce que je vais faire
- (III) Quelles difficultés je rencontre





Management Visuel

# SCRUM

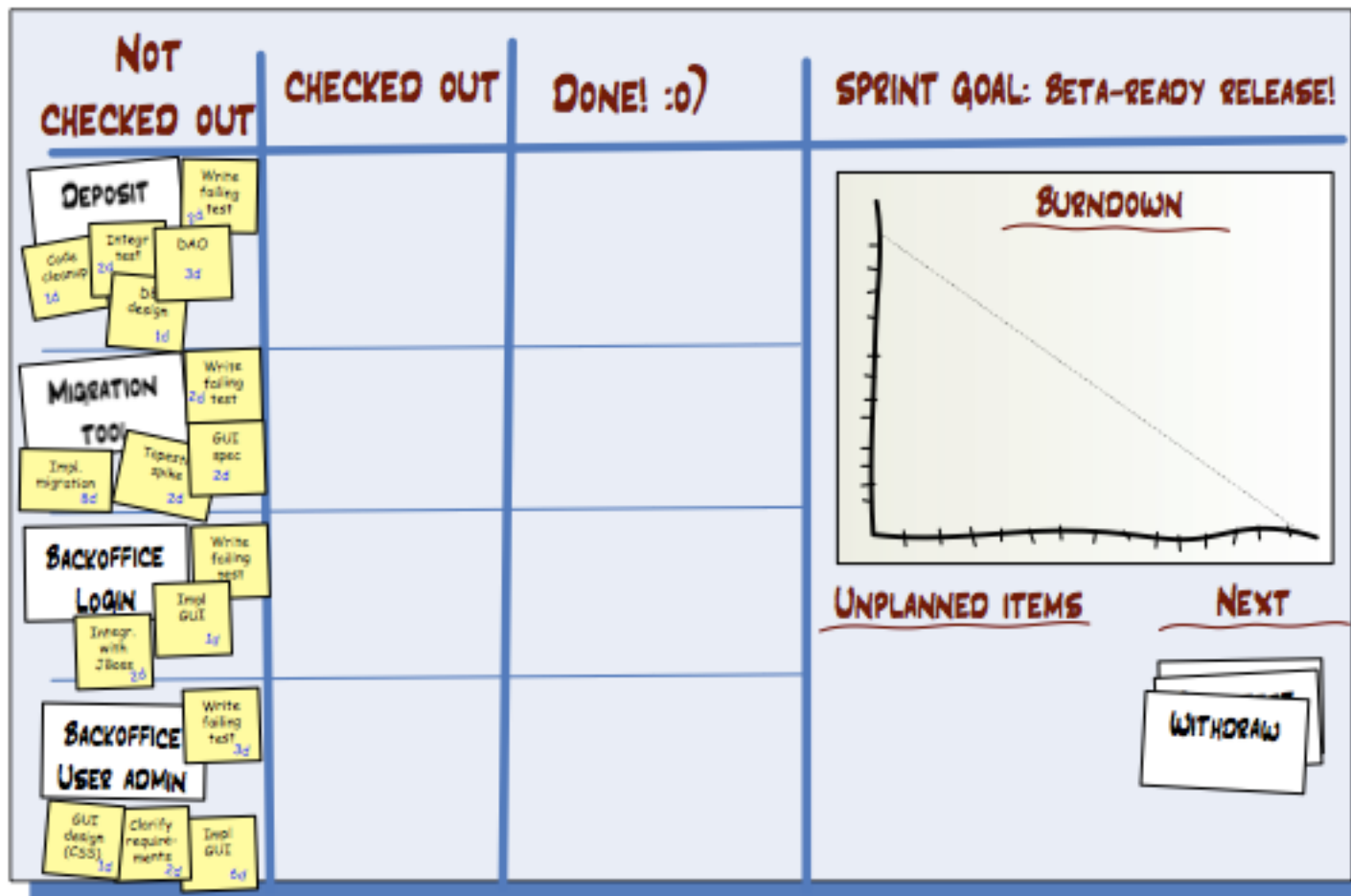


# Scrum Board



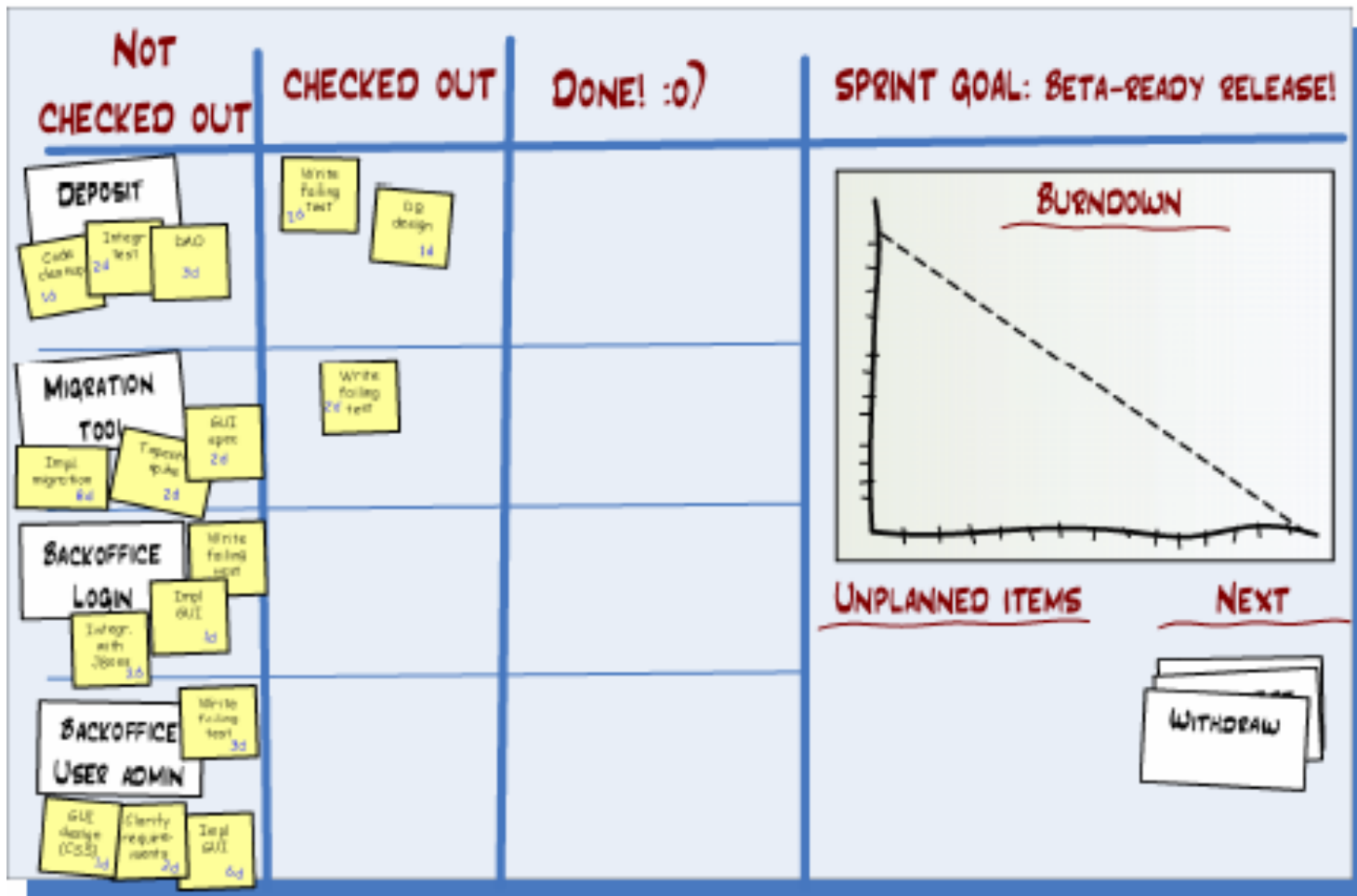
# Typical Visual Management with SCRUM

Sprint 4 – Day 0,5 of 30



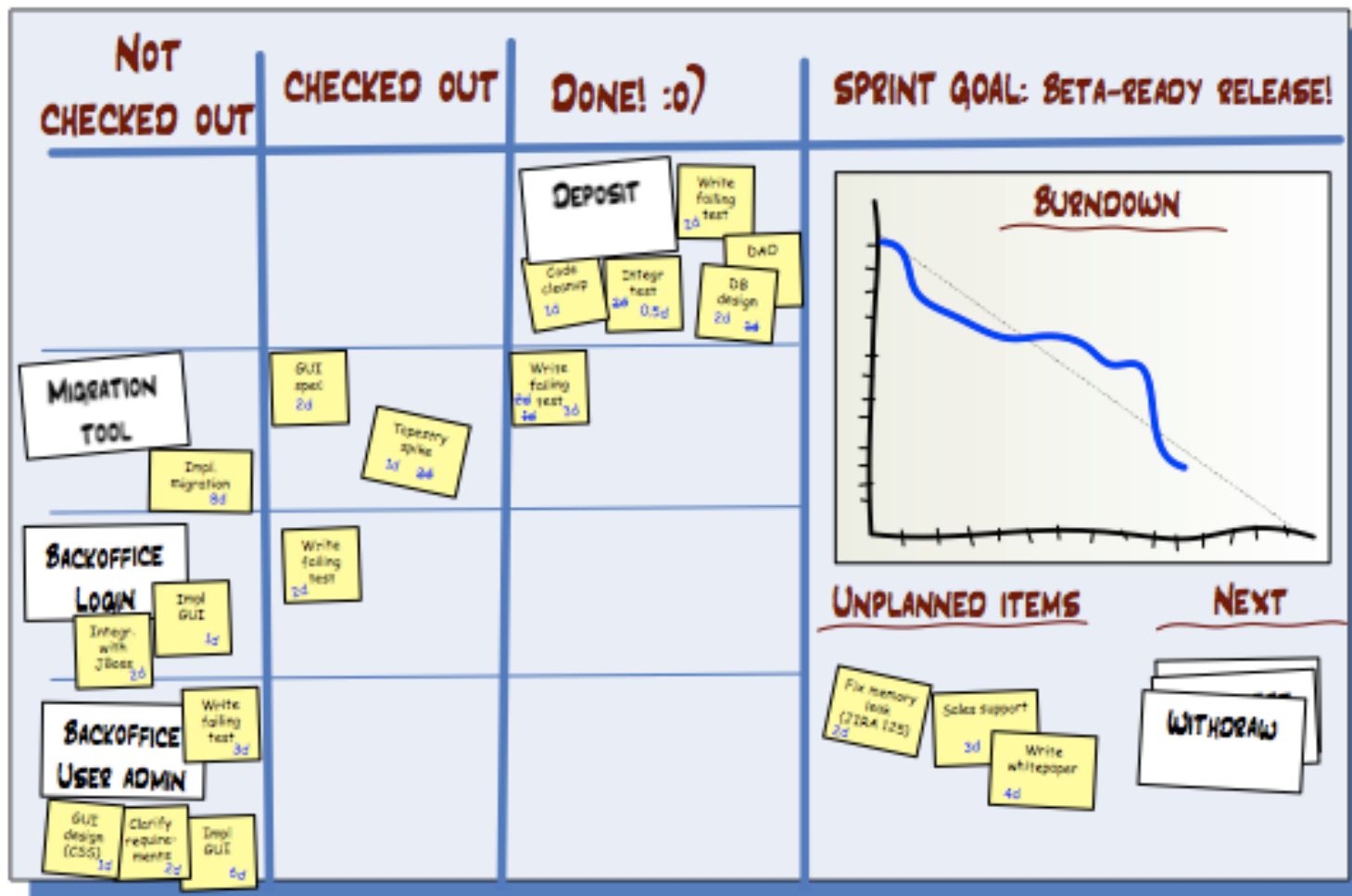
# Typical Visual Management with SCRUM

Sprint 4 – Day 01 of 30



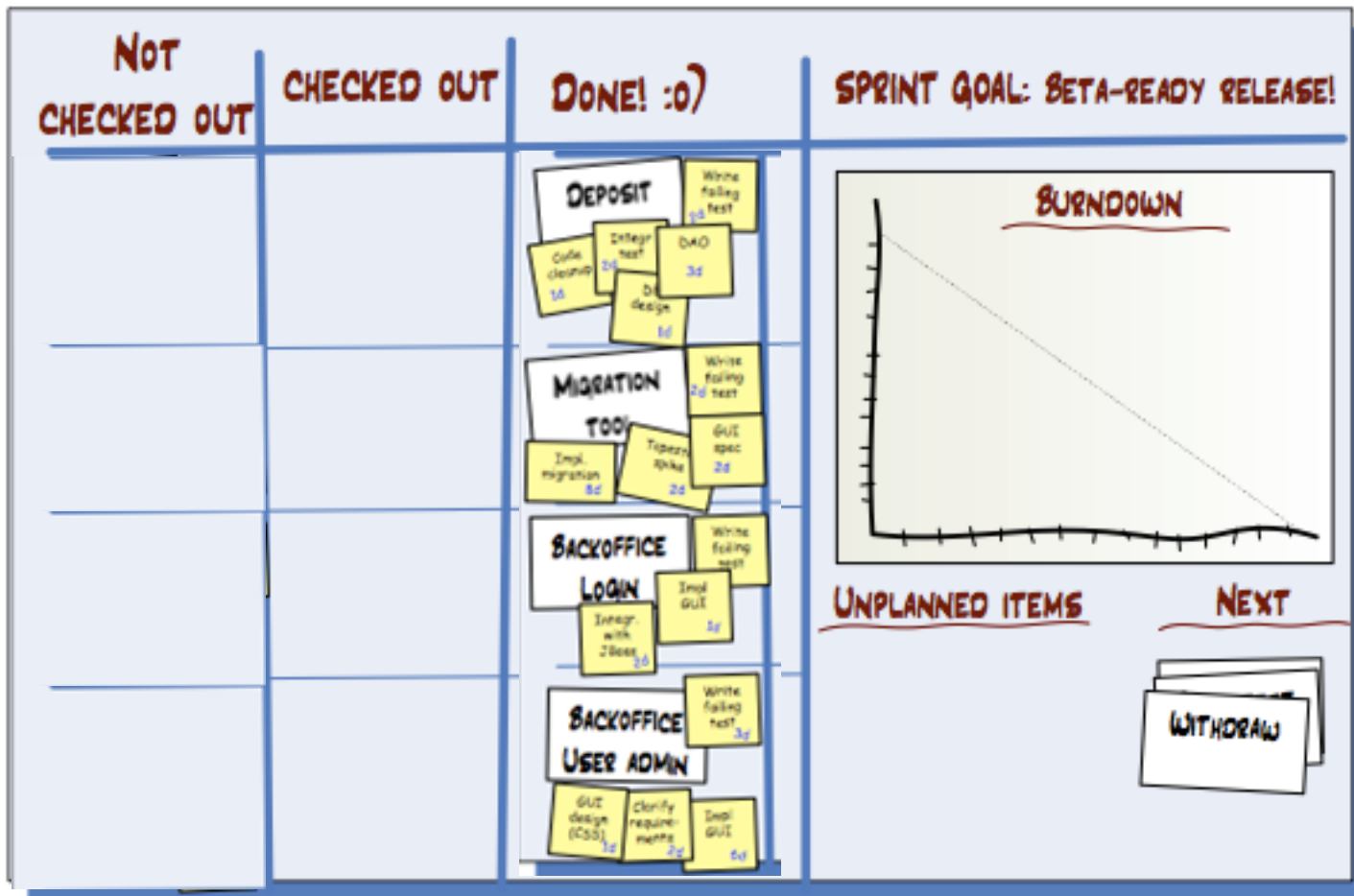
# Typical Visual Management with SCRUM

Sprint 4 – Day 12 of 30



# Typical Visual Management with SCRUM

Sprint 4 – Day 30 of 30 (Happy Scenario 😊 )

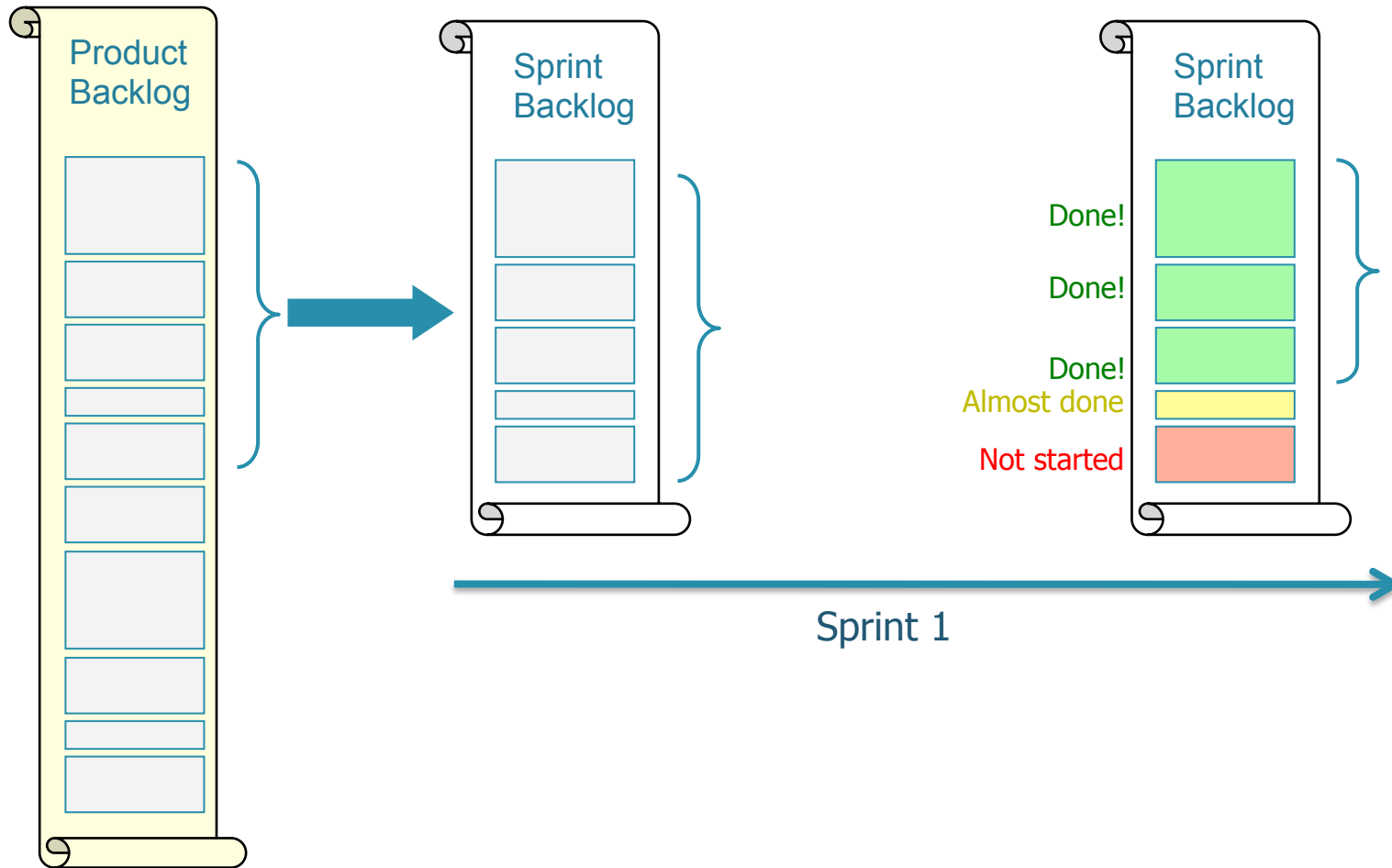




# Le Product Backlog et le Sprint Backlog

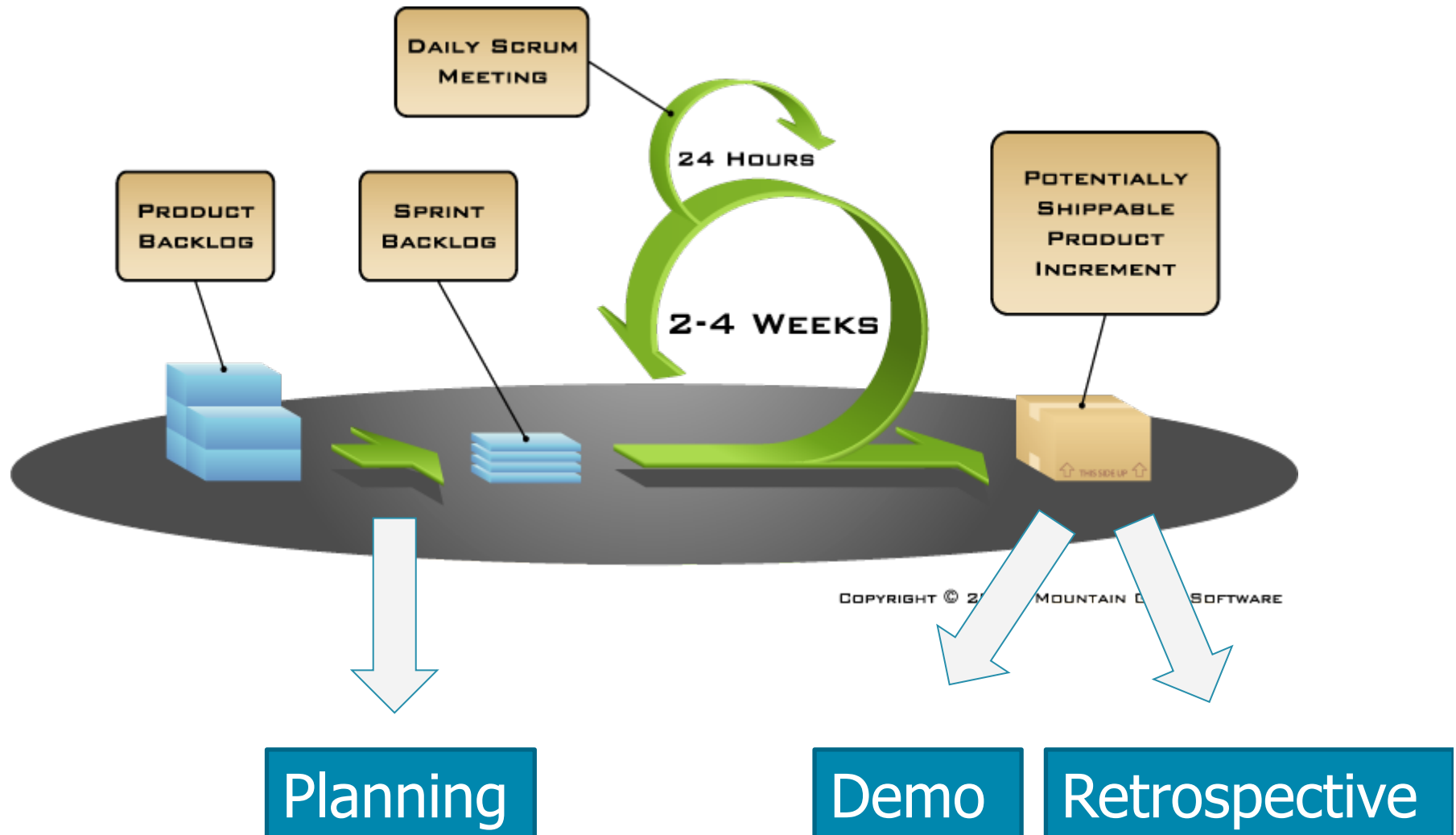
Début du sprint 1

Fin du sprint 1



**Fin d'un sprint**  
**SCRUM**

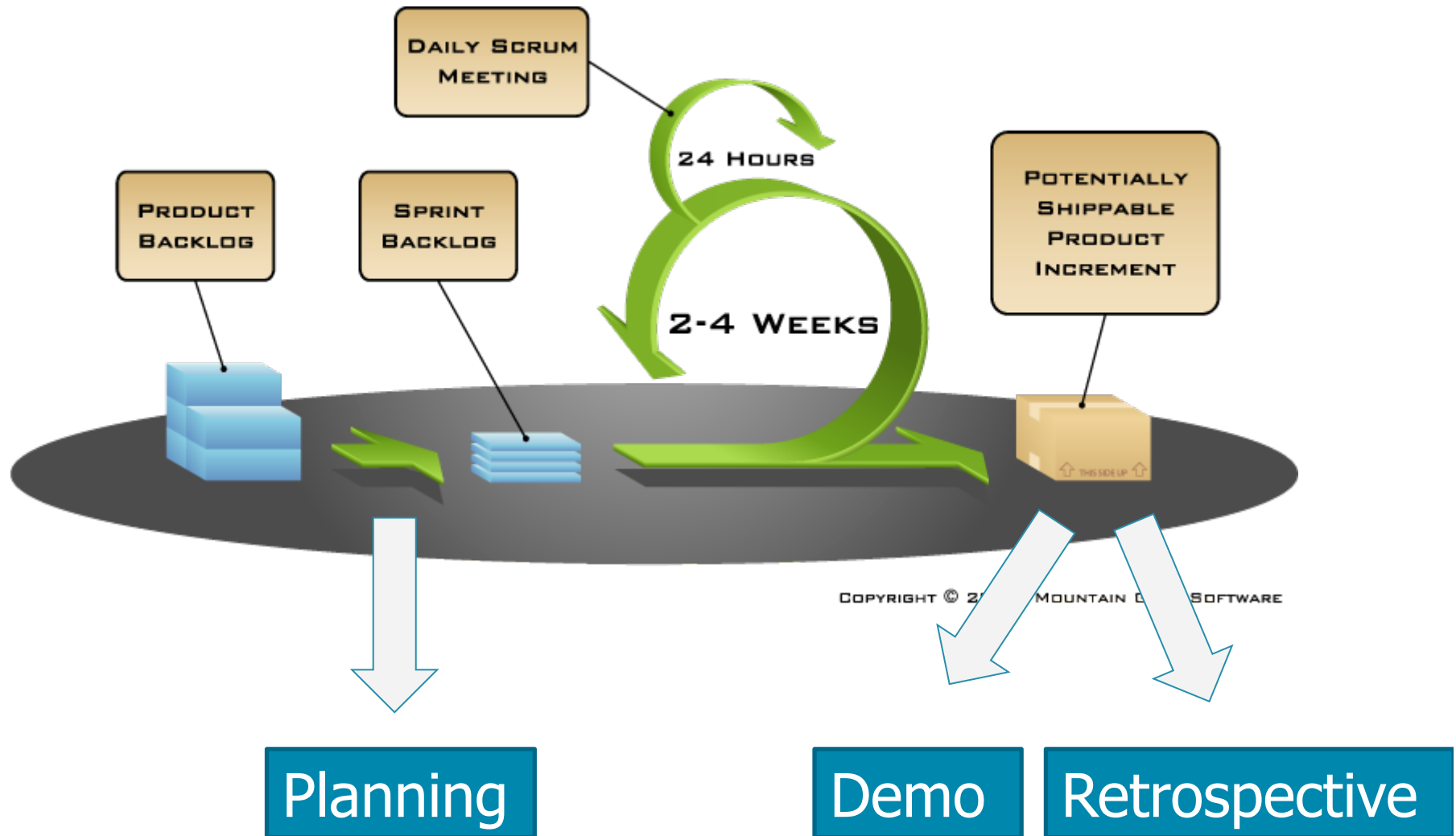
# Scrum en Bref: les différentes réunions et les différents documents



# Sprint Review (demo)

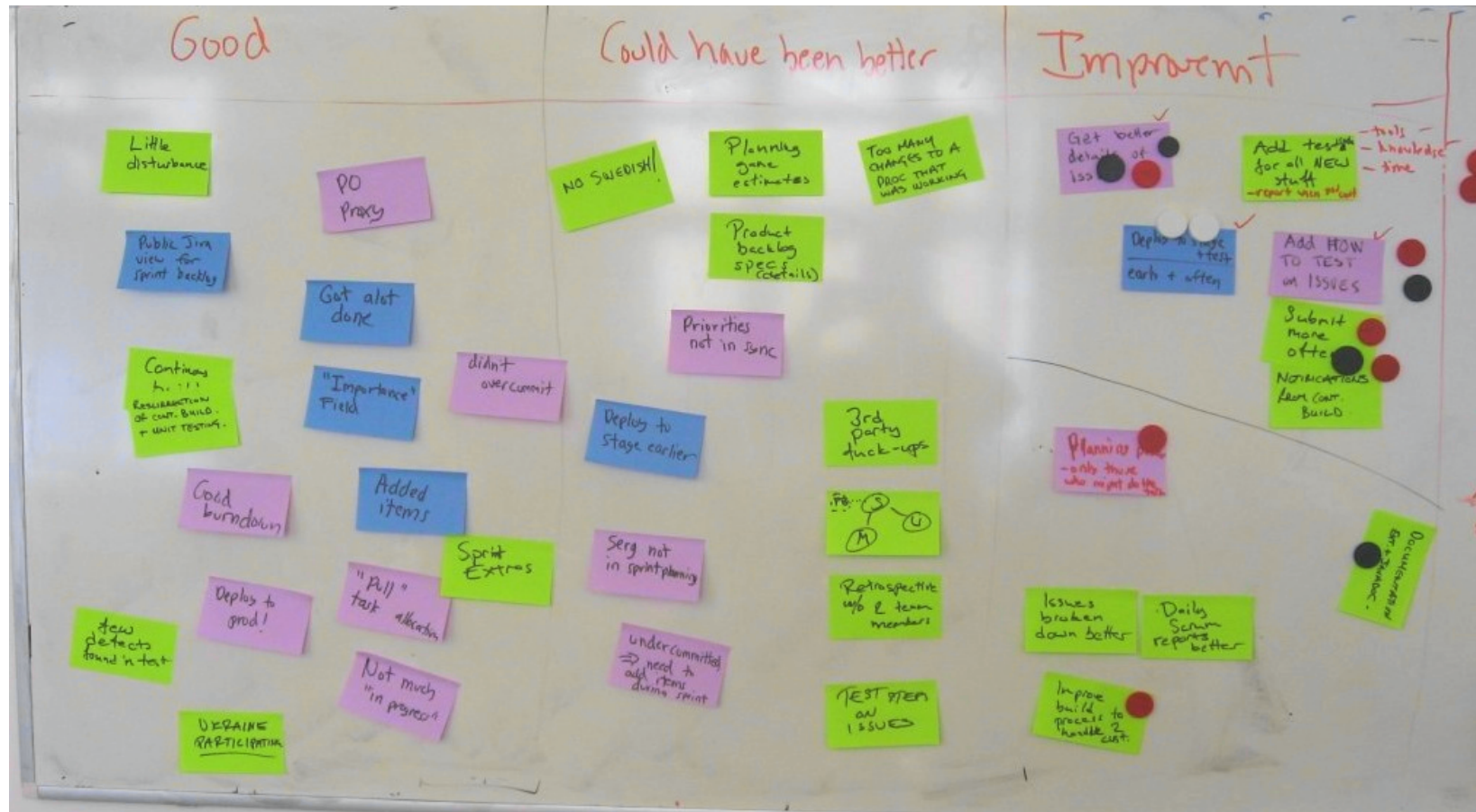
- Répond à la question: qu'avons-nous accompli ?
- L'équipe réalise la démo aux stakeholders
- Seulement ce qui est terminé à 100% est montré
- Direct feedback de la part des stakeholders
- Le Feedback est inclus dans la product backlog

# Scrum en Bref: les différentes réunions et les différents documents



# Sprint retrospective

Part 1: Que s'est-il passé ?  
Part 2: Qu'est-ce qu'on veut faire de différent pour le prochain sprint ?





**Release Planning**

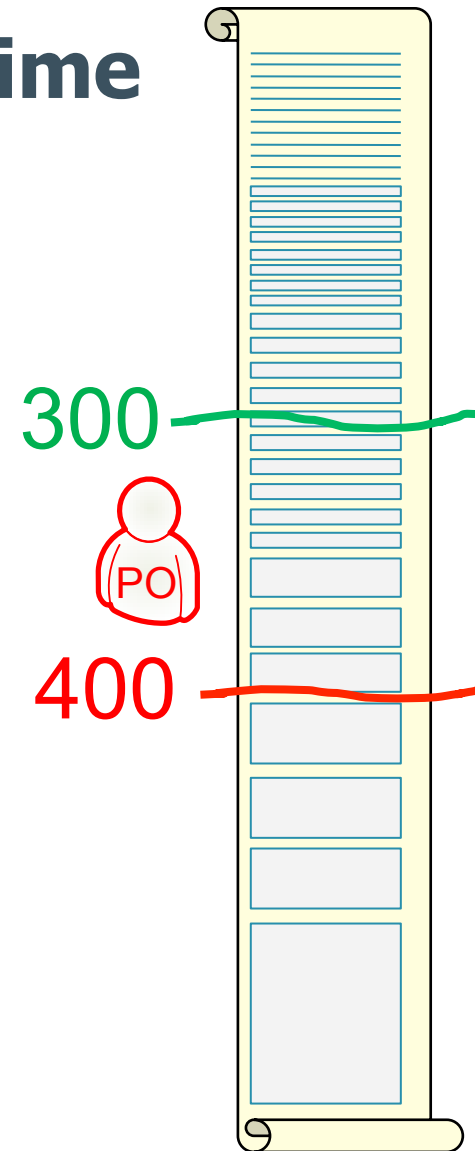
**SCRUM**

# Release planning – fixed time

- Today is Aug 6
- Sprint length = 2 weeks
- Velocity = 30 - 40

What will be done  
by X-mas?

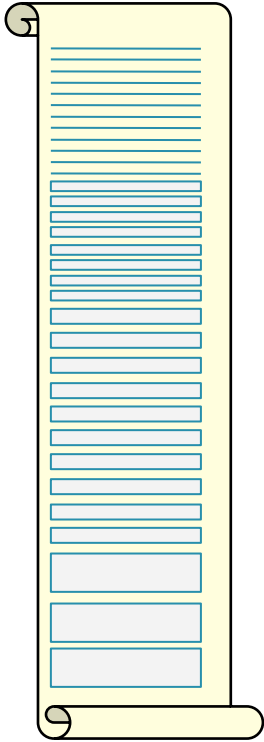
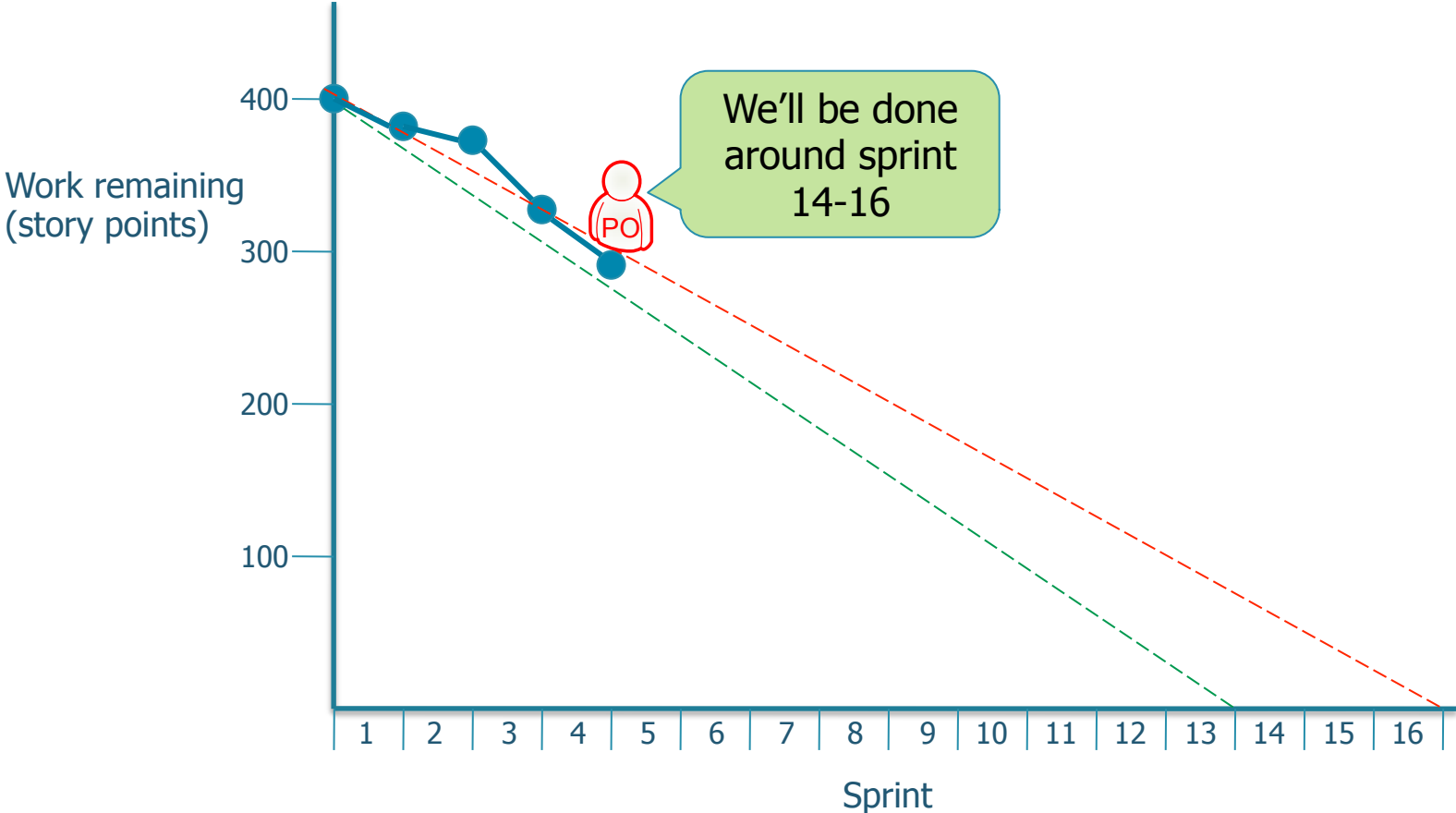
(10 sprints)



# Release planning – fixed scope

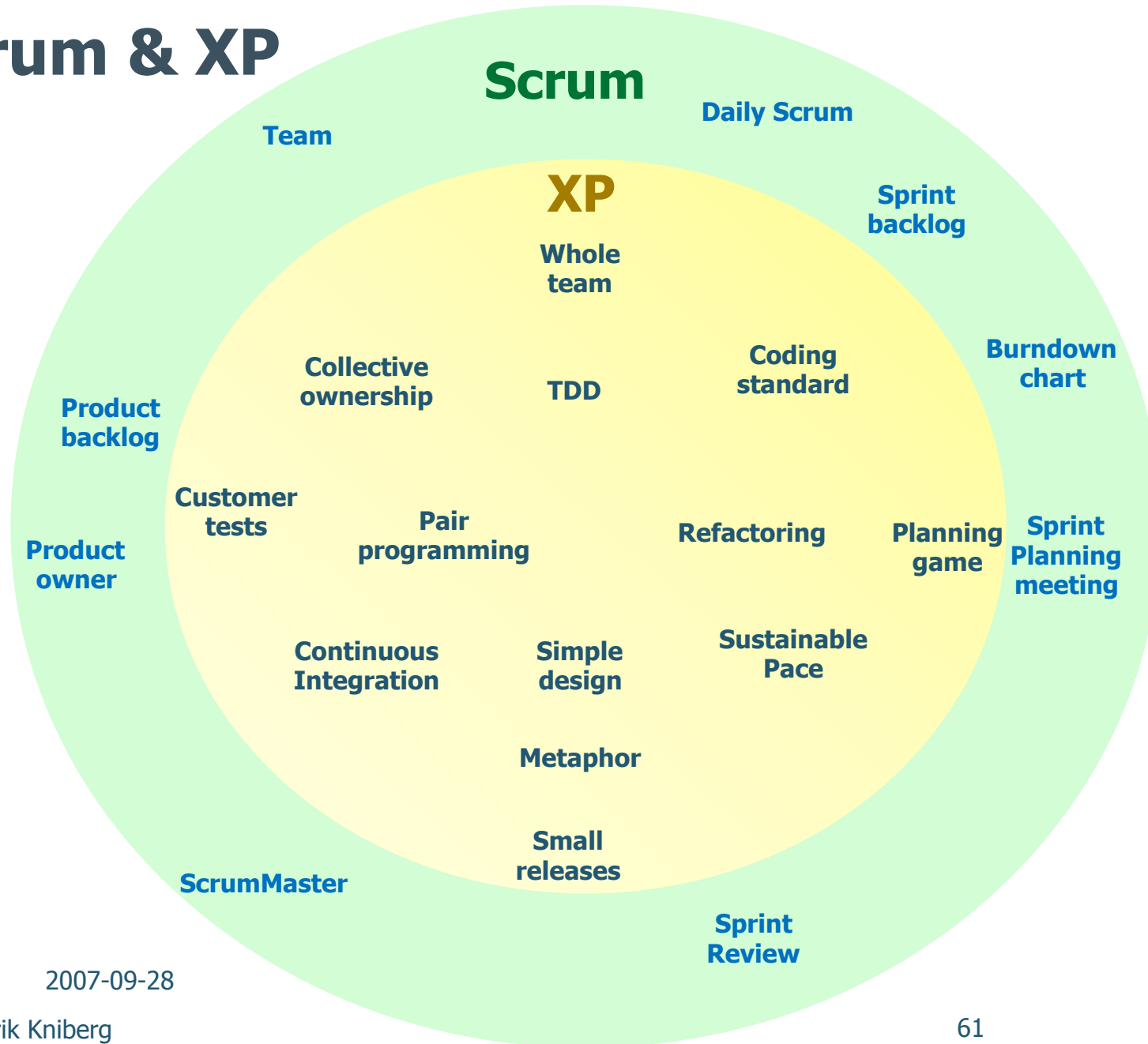
When will all of this be done?

Release burndown chart



# **TECHNIQUES D'INGÉNIERIE ET D'ARCHITECTURE**

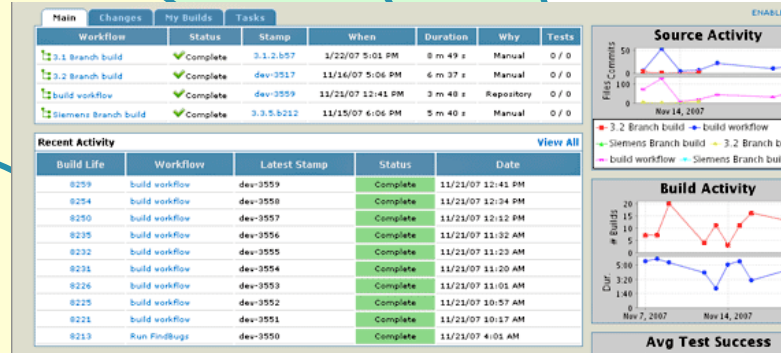
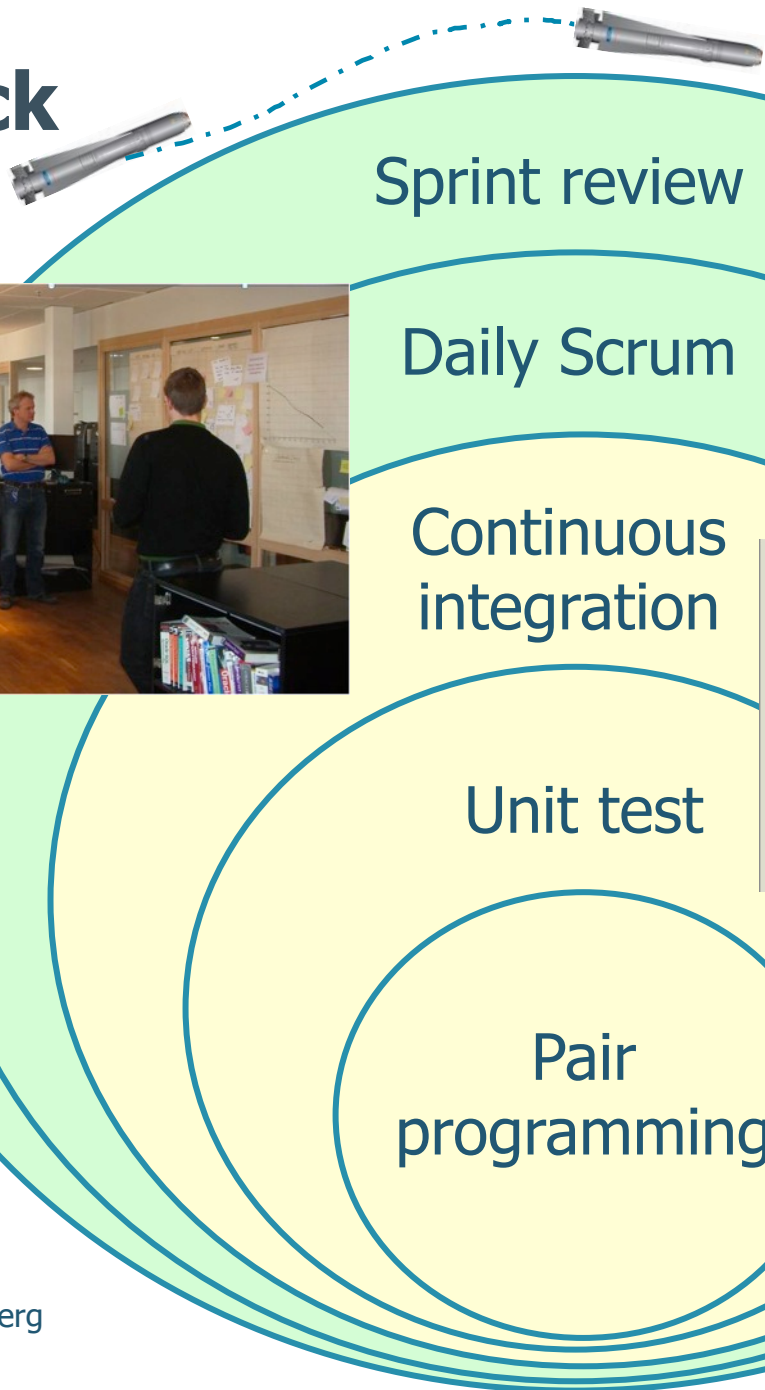
# Scrum & XP



2007-09-28

Henrik Kniberg

# Feedback loops



Henrik Kniberg



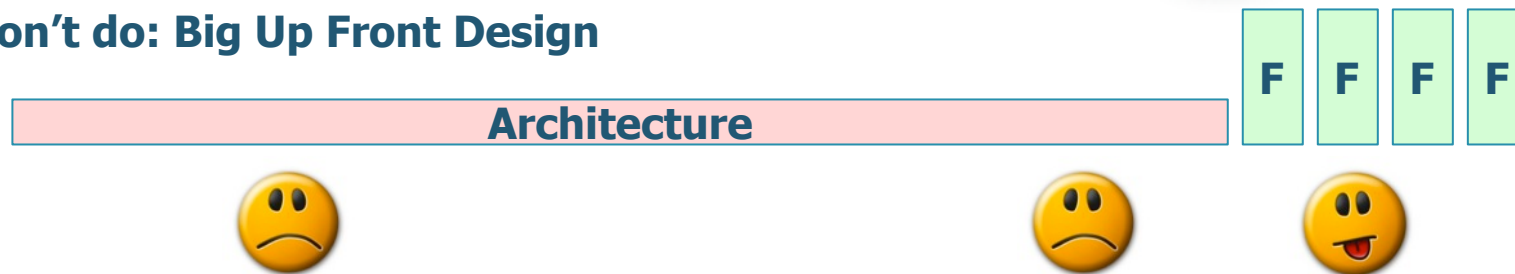
# Agile architecture



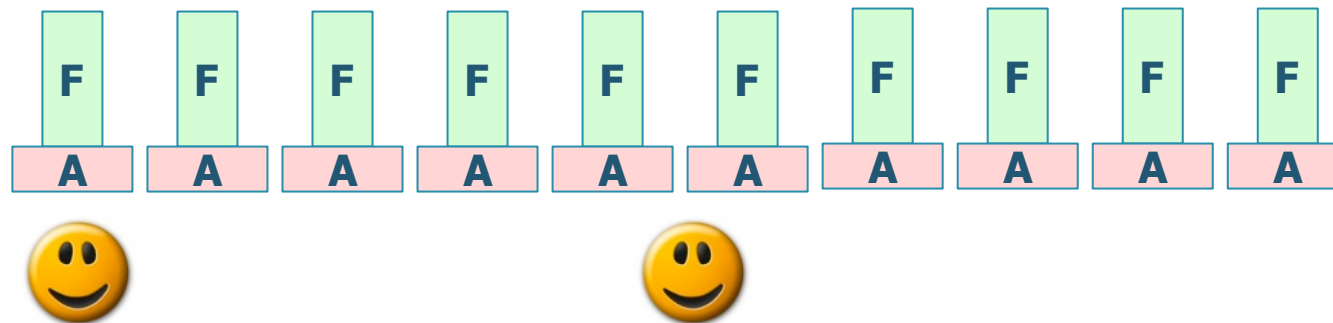
Don't do: Quick 'n dirty features => Slow 'n dirty => Entropy death? Big bang rewrite?



Don't do: Big Up Front Design



Do: Find a balance



# Clean & simple code

Simple code:

- 1. Passes all tests
- 2. No duplication
- 3. Readable
- 4. Minimal

Simple is hard!

~~Code is an asset~~  
All code is cost!  
Some code is value.

```
public class Dog {  
    private final String name;  
    private int woofCount = 0;  
  
    public Dog(String name) {  
        this.name = name;  
    }  
  
    public void woof() {  
        ++woofCount;  
    }  
}
```



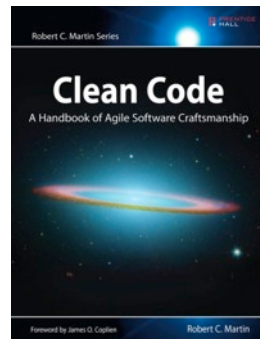
```
import java.sql.Connection;  
import java.util.concurrent.ExecutorService;  
import java.util.concurrent.Executors;  
  
public class Dog {  
    private Executor executor = Executors.newFixedThreadPool(18);  
    private int CACHE_SIZE = 50;  
  
    public Dog()  
    {  
        try  
        {  
            Class.forName("oracle.jdbc ThinDriver");  
            connection = DriverManager.getConnection("jdbc:oracle:thin:@prod", "admin", "beefhead");  
            statement = connection.prepareStatement("insert into Dog values (?, ?, ?)");  
        } catch (ClassNotFoundException e) {}  
  
        new Thread().start();  
    }  
  
    public void woof(Person woofCaller) {  
        Connection connection = null;  
        PreparedStatement statement = null;  
        try {  
            connection = DriverManager.getConnection("jdbc:oracle:thin:@prod", "admin", "beefhead");  
            statement = connection.prepareStatement("insert into Dog values (?, ?, ?)");  
            statement.setLong(1, System.currentTimeMillis());  
            statement.setString(2, person.getName());  
            statement.setString(3, person.getPhoneNumber().getNumber());  
            statement.executeUpdate();  
        }  
    }  
}  
  
Connection a = DriverManager.getConnection("jdbc:oracle:thin:@prod", "admin", "beefhead");  
b = a.prepareStatement("select * from Dog where name = '" + name + "'");  
c = b.executeQuery();  
if (c.next()) {  
    String foundName = c.getString("name");  
    PhoneNumber phoneNumber = new PhoneNumber(c.getString("woofCount"));  
    Person person = new Person(foundName, phoneNumber);  
    return person;  
} else {  
    return new Person("", null);  
}  
  
} catch (SQLException e) {  
    return null;  
} catch (IllegalArgumentException x) {  
    throw x;  
}
```

```
All() {  
    DriverManager.getConnection("jdbc:oracle:thin:@prod", "admin", "beefhead");  
    tion.prepareStatement("insert into Dog values (?, ?, ?)");  
    l, System.currentTimeMillis());  
    {  
        tString("name");  
        umber = new PhoneNumber(c.getString("woofCount"));  
        w Person(foundName, phoneNumber);  
    }  
}
```



Embrace Change!

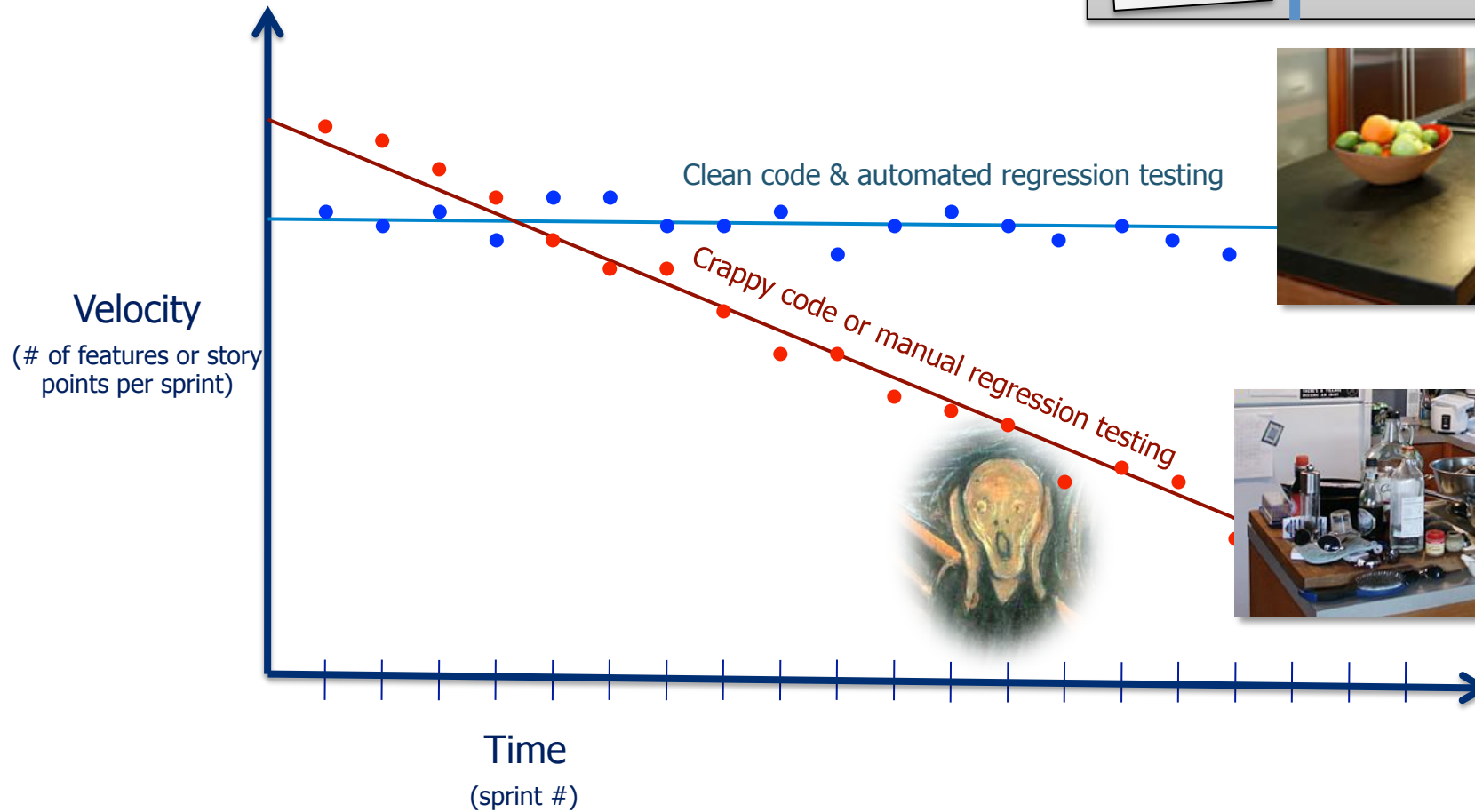
Kent Beck  
Henrik Kniberg



Robert C Martin (Uncle Bob)

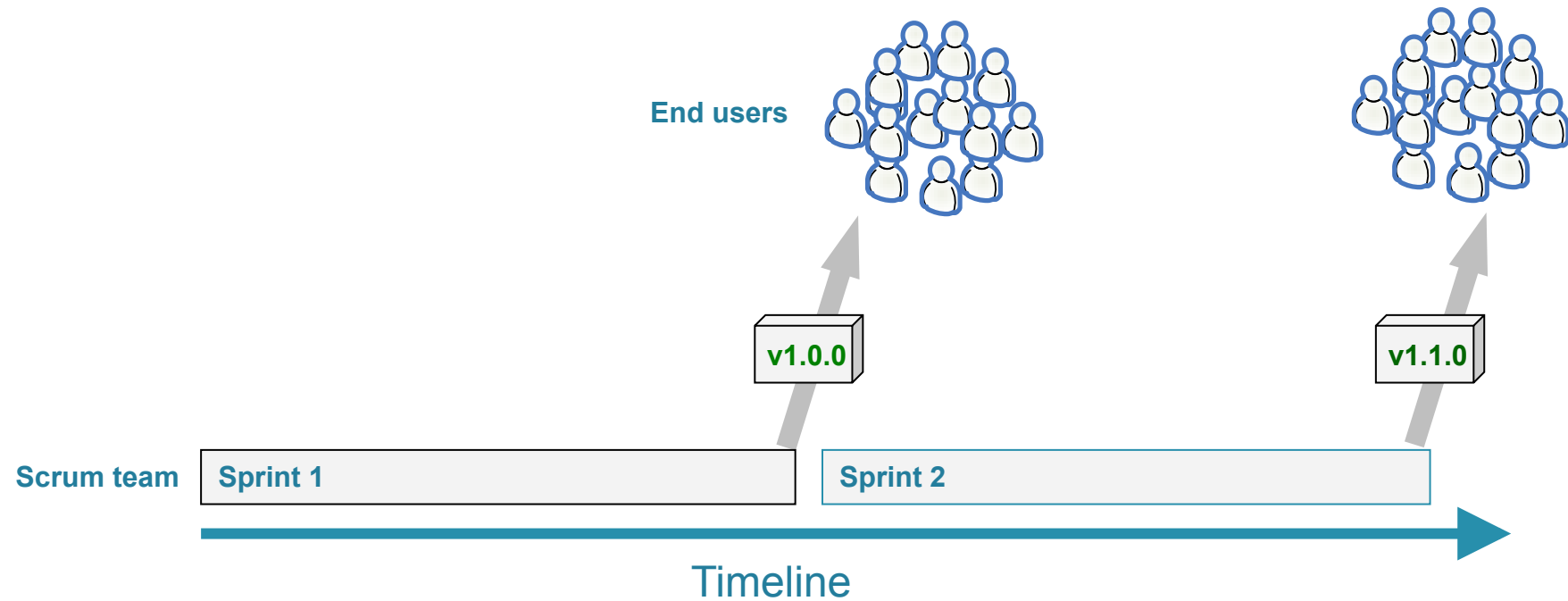
# "Done" includes clean code & regression testing

To do	Doing	Done
Withdraw Transfer	Deposit	Register



# Testing

# Testing – ideal case



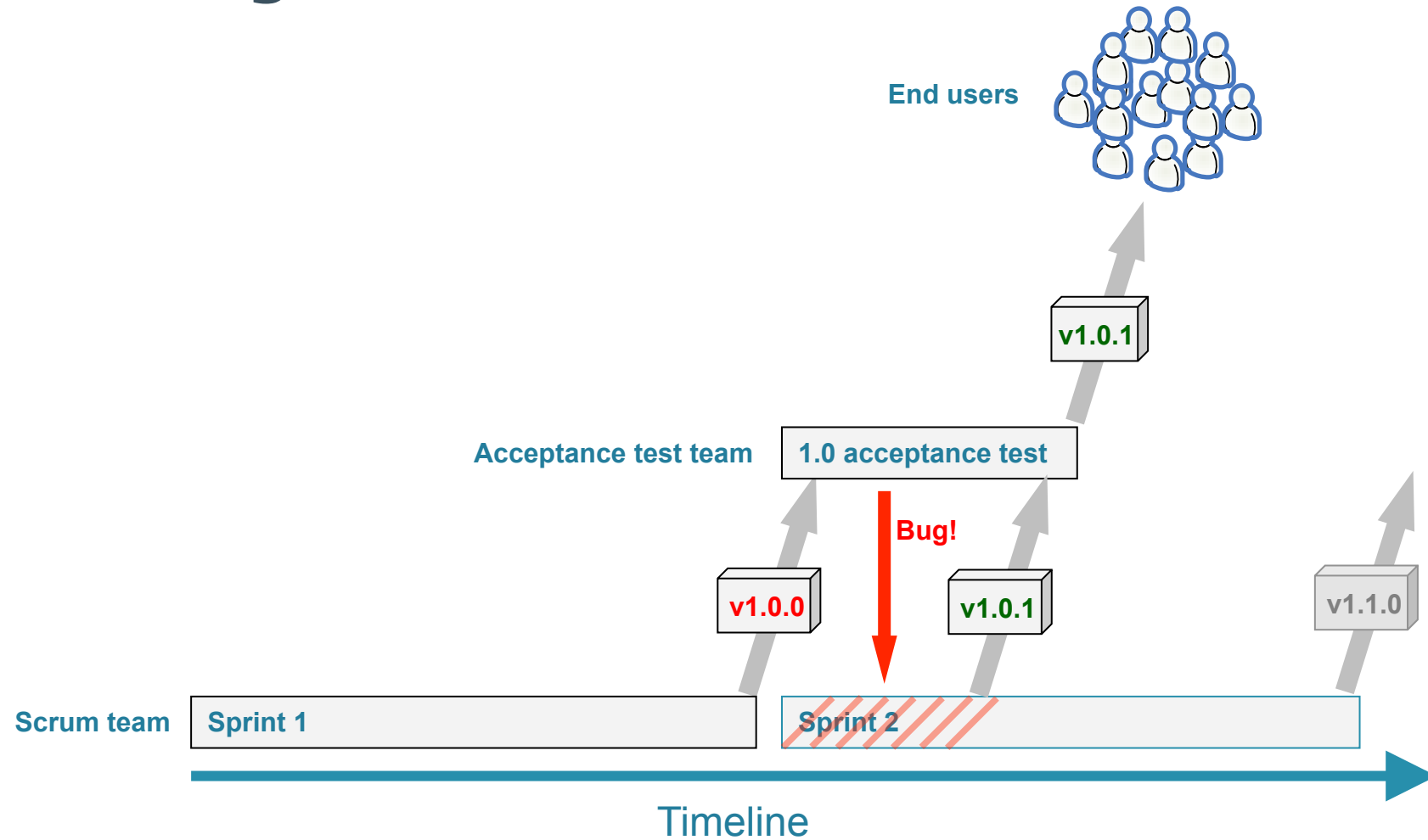
24/06/13

Henrik Kniberg

67



# Testing – common alternative



24/06/13

Henrik Kniberg

68

# Et après ?

- **C'est quoi nos plus gros problèmes ?**
- **Veut-on essayer d'utiliser Agile pour les résoudre ?**
- **Formation, savoir-faire, coaching**

# Les raisons d'échecs

- **Ne pas vouloir voir (ou traiter) les problèmes**
- **Faire comme avant mais avec le vocabulaire "Agile"**
- **Customiser (trop tôt) Agile**



# Bénéfices des méthodes Agile

- Rapidement sur le marché
- Qualité
- Flexibilité
- Le juste produit
- Visibilité
- Contrôle des coûts
- Moins de risques
- plus motivant
- Revenus

# Le jeu des balles: ce que ça nous apprend

- **Le Processus Scrum basé sur le cycle de Deming:**  
[http://fr.wikipedia.org/wiki/Roue\\_de\\_Deming](http://fr.wikipedia.org/wiki/Roue_de_Deming)
- **Le principe "Inspect and Adapt" Faites qqch puis, debriefing sur l'expérience.**
- **On a tendance à s'ajouter des contraintes**
- **Principe du bottleneck (  
[http://fr.wikipedia.org/wiki/Th%C3%A9orie\\_des\\_contraintes](http://fr.wikipedia.org/wiki/Th%C3%A9orie_des_contraintes))**
- **Une équipe de 15 personnes, C'est 15 cerveaux plein d'idées**

# Sources

- **The Essence of Agile (Keynote by Henrik Kniberg)**
- **<http://agileatlas.org/atlas/scrum>**