

Agile Business Day 2016

#abd2016 #venezia



Venezia

17 Settembre 2016

Autore: Maksym Rybak

<http://about.me/maksym.rybak>

Agile business day 2016

#abd2016 #venezia

<http://www.agilebusinessday.com>

Premessa

In questo documento (mini relazione) descrivo a testo libero com'è andata la conferenza Agile Business Day a Venezia il 17/9/2016. E' stato molto interessante partecipare come ascoltatore. Impressione tra 1h era "Cavolo, sono nel posto giusto".

Ho partecipato ai seguenti talk:

- Keynote di apertura
- Scrum e metodi Agile in una software factory a trazione ibrida
- Management 3.0: management art in today's turbulent world
- Business triathlon: come cambiare, innovare e produrre valore rapidamente, in ambienti sottoposti a costante turbolenza
- Pausa pranzo :)
- Keynote di pomeriggio
- Introdurre design thinking in un processo agile: perché, che impatto ha sul business, come fare
- Favorire i "feature teams" con architetture a microservizi

Ovviamente, chi leggerà questo documento, è libero a correggermi nelle mie imprecisioni/missunderstanding.

Tutti le informazioni per quanto riguarda la conferenza potete trovare sul sito menzionato prima.

Keynote di apertura

Speaker Massimo Messina (head of global ICT presso UniCredit S.p.A). Purtroppo siamo arrivati in ritardo per poter seguire completamente il keynote di apertura. Il motivo di ritardo è stato: 1 - non abbiamo considerato che il parcheggio multi piano a Venezia è di 8 piani e che ci mandano proprio



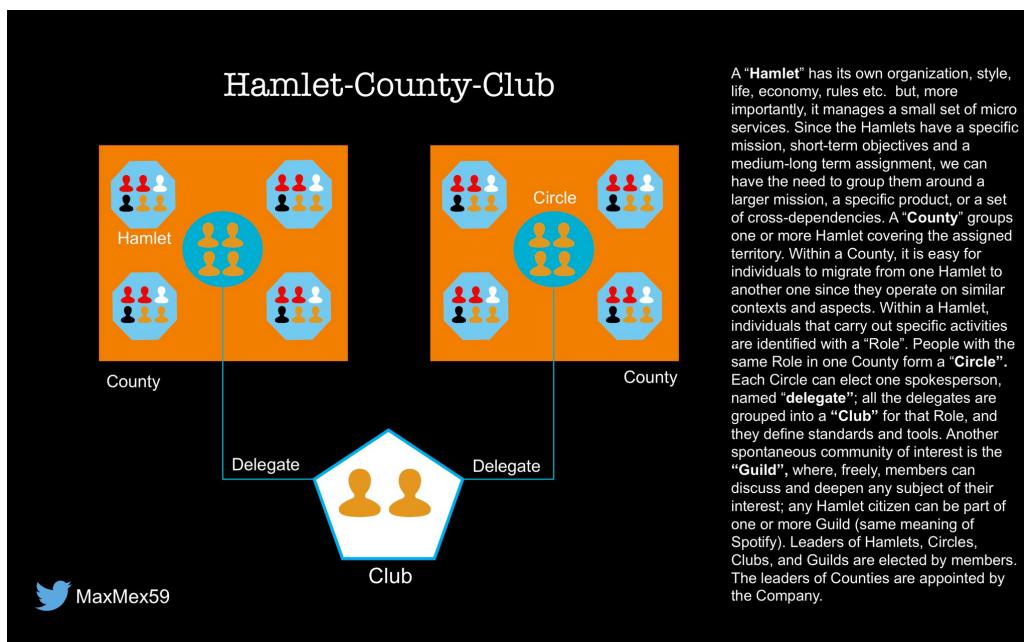
all’ottavo, 2 - probabilità di sbagliare la strada per arrivare all’università Ca’ Foscari di Venezia :)

I punti che siamo riusciti ad ascoltare erano:

- Autonomia - lasciare massima autonomia ai gruppi di lavoro
- Trust personale e controlli autonomi - fiducia alle persone, nessun controllo evidente da parte dei “manager”, le persone che lavorano devono essere in grado di fare i controlli autonomi sul lavoro fatto
- Persone multidisciplinari - si mette una persona multidisciplinare in un gruppo di lavoro e si aspetta che diventa il leader, è stato ribadito da più speaker che è molto difficile trovare le persone multidisciplinari
- Keep simple and fast - non complicare, massima facilità nella collaborazione e lavoro giornaliero

Il video del talk: <https://www.youtube.com/watch?v=TWICc32UJms&feature=youtu.be&t=1h10m40s>

Qualche immagine interessante pescata da twitter:



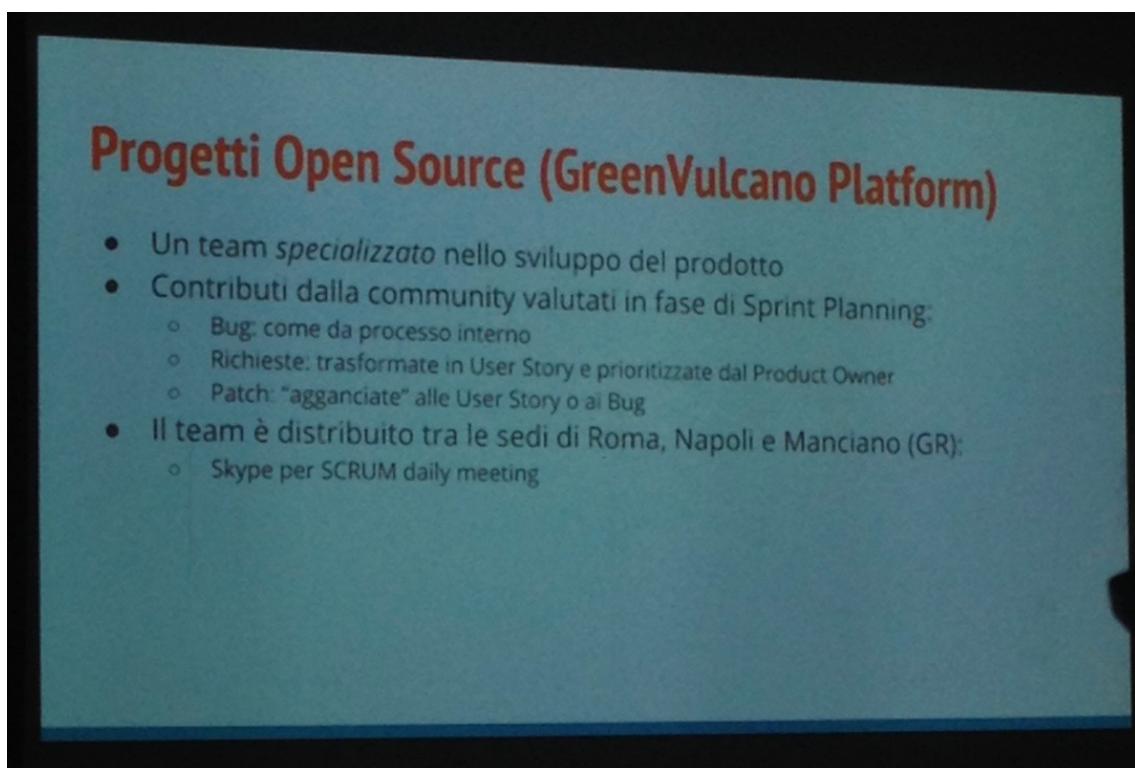
Scrum e metodi agili in una software factory a trazione ibrida

Speaker Domenico Barra. Ha parlato di come adottano i principi Agile nella propria azienda. Azienda di Roma che opera nel settore IT, attività di integrazione (CI) nelle grandi aziende. Azienda ha tre sedi, chiamiamoli *Sede1*, *Sede2*, *Sede3*. In *Sede1* ci sono principalmente i back-end developer. In *Sede2*, esperti database. E in *Sede3*, esperti di grafica e front-end.

Vediamo come sono riusciti ad organizzare il loro lavoro.

Il talk era suddiviso in tre scenari:

1. Progetti opensource - guadagno economico arriva dal pagamento del supporto e manutenzione di prodotti opensource. Per ogni prodotto esiste un team specializzato. Il team può essere formato anche dalle persone che fisicamente sono nelle sedi diverse. Lo strumento principale per daily meeting è Skype. Usano trello come strumento di collaborazione (<https://trello.com/>). Ogni tre settimane fanno la dimostrazione al cliente finale/partner trusted (partner con quale collaborano da molti anni).



2. Progetti enterprise - esiste priorità all'attrattività, rispetto funzionalità.
Hanno lavorato sia con aziende piccole che con delle aziende grandi.
Aziende piccole hanno il budget inferiore ma sono più dinamiche. Aziende

grandi, viceversa, budget più grande ma sono meno flessibili. Azienda piccola può non avere il product owner, al contrario di una azienda grande.

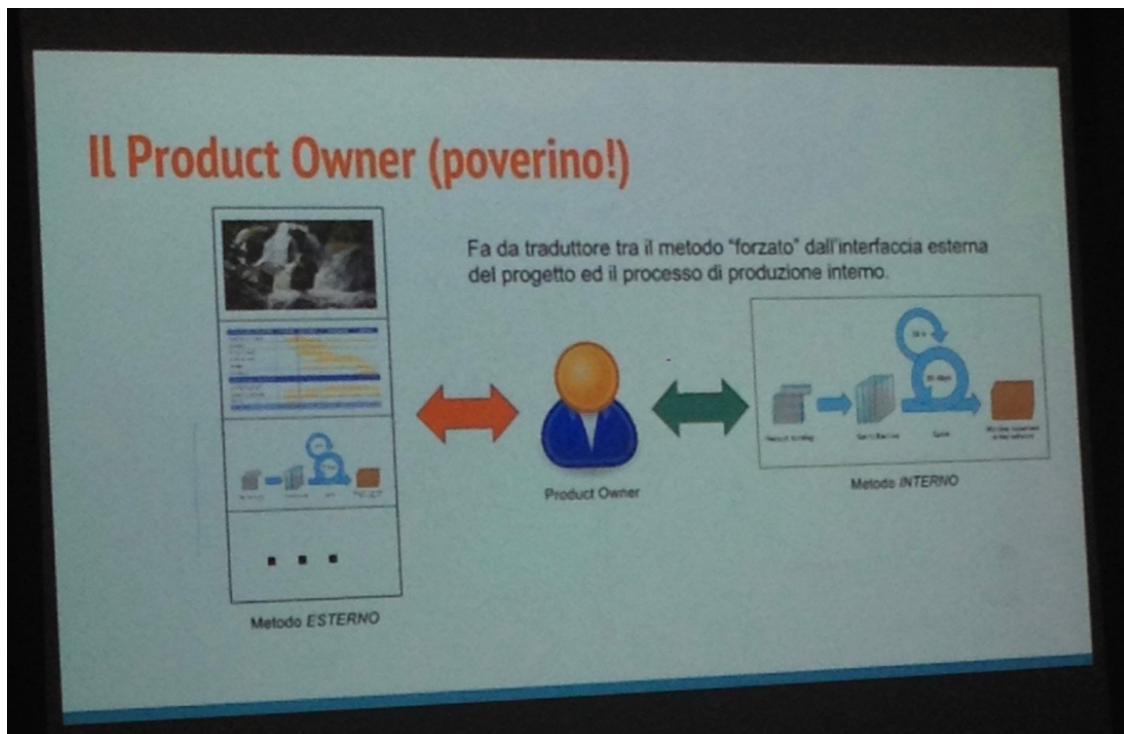
Progetti (enterprise) su commissione

- Assioma 1: $P(\text{attrattività}) - P(\text{funzionalità}) = P(\text{attrattività})$
i.e. si valuta negativamente il **brutto**, anche se **funziona**
- Assioma 2: Il **brutto** è (spesso) **soggettivo**

... quindi ?

- I designer si aggiungono al famoso "Sprint Zero"
- Validazione e "battaglia" dei mock-up prima di coinvolgere il team

Il product owner serve per soddisfare le esigenze burocratiche in una grande azienda.

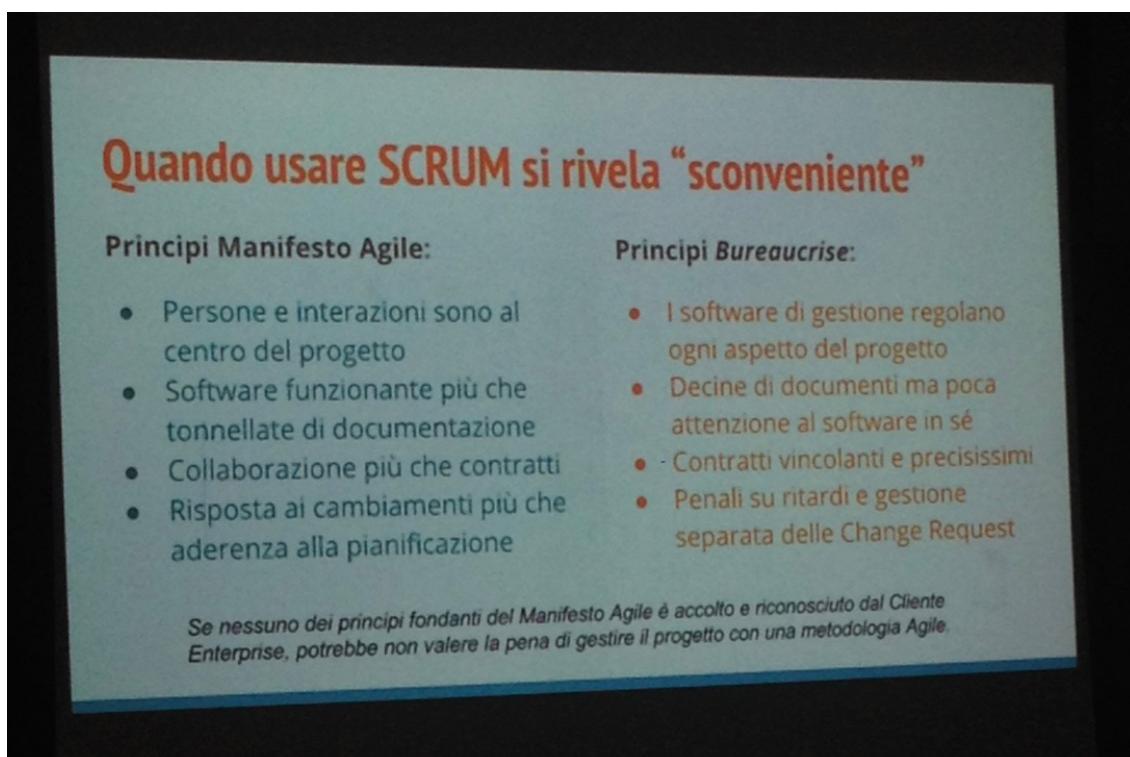


3. Progetti interni - sviluppo di strumenti interni per migliorare operatività di diversi team. Si dà la priorità alle funzionalità, comunicazione facile tra gli stakeholders (utenti finali). Riunioni meno ufficiali, parliamo tra i colleghi.

Come viene formato un team: esiste un user expert, un front-end developer e due back-end. Team possono essere distribuiti tra città differenti. Ogni sede ha la sua Agile board. Ed esiste una board comune. Ogni mattina viene fatto stand-up meeting.

La difficoltà principale è spostare un task da una sede all'altra.

Quando usare Scrum si rivela sconveniente:

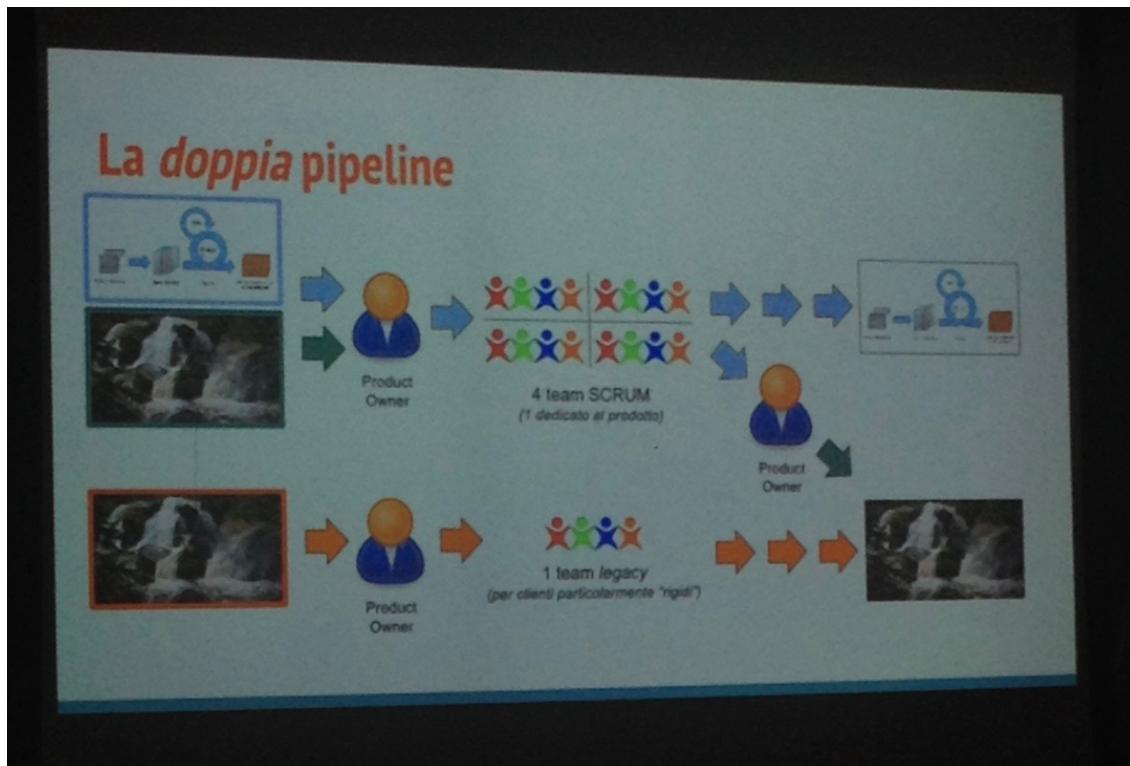


La parte destra è molto simile alla situazione che abbiamo ora in Credem.

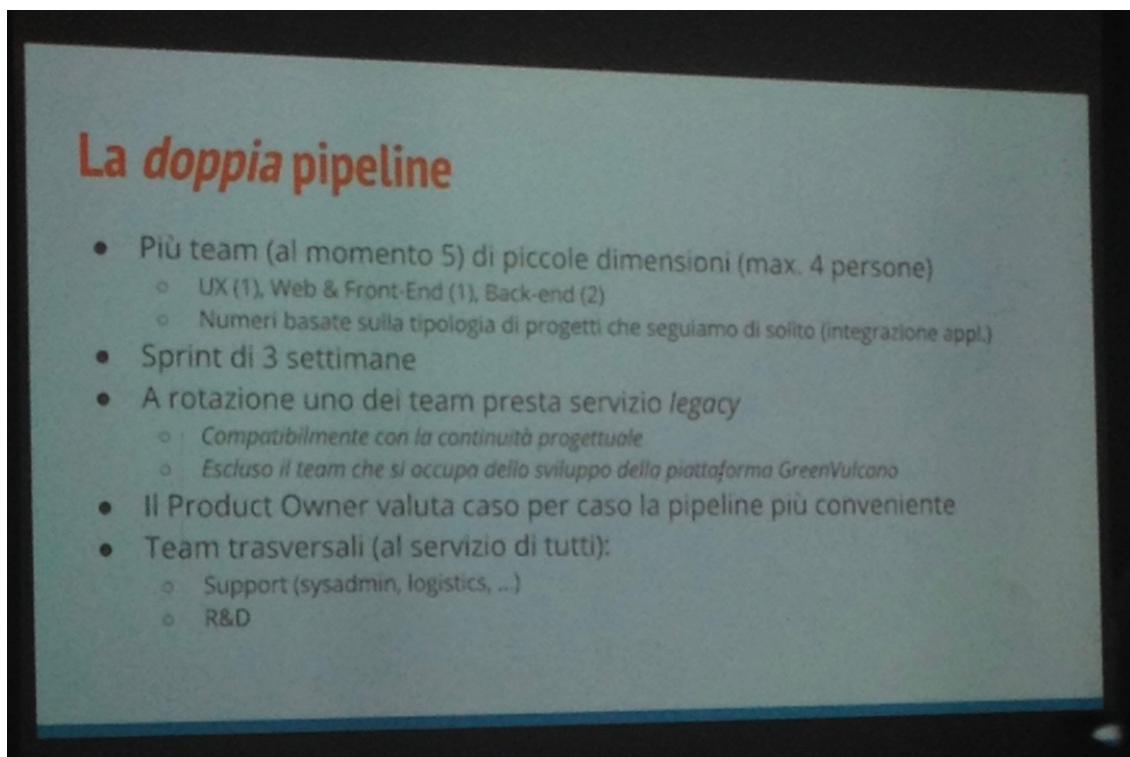
Il talk di Emiliano Soldi (vedi sotto) può essere una soluzione?

La doppia pipeline.

I membri dei team Scrum possono cambiare i prodotti (team).



Come è stato specificato nella slide sotto, sprint è di 3 settimane, hanno ribadito



che diminuire (a due settimane) o ad aumentare (a 4 settimane) non è conveniente. Se si diminuisce, si presenta troppo lavoro per i product owner, per esempio per rimappare tutte le attività allo sprint successivo.

A rotazione uno dei team presta servizio legacy. Non è quello che volevamo fare anche in Credem? :) per le attività di AM.

Arrivata una domanda molto interessante da un ascoltatore: “come si gestisce il passaggio da un team Scrum ad un’altro da parte di un membro, oppure il passaggio di una persona dal team legacy al team Scrum? La difficoltà di adattarsi, e più che altro, la velocità da parte di una persona di cambiare il team ed entrare su un nuovo prodotto”. Direi che non esiste una risposta a questa domanda. Le persone sono diverse ed apprendimento di uno può essere più veloce di un’altro. Non dimentichiamo la resistenza al cambiamento.

Management 3.0: management art in today's turbulent world

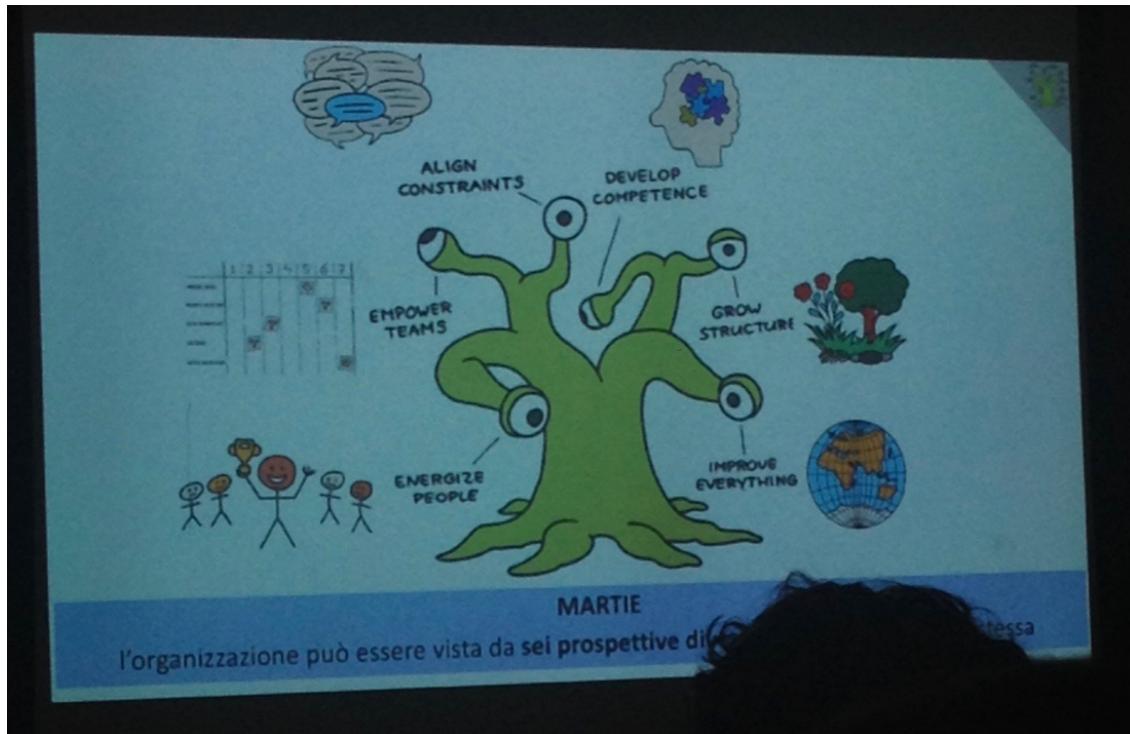
Speaker Felice Pescatore.

Ha parlato delle difficoltà di predire l’evoluzione e il comportamento di una azienda. Importanza del fatto che i manager devono imparare a delegare.

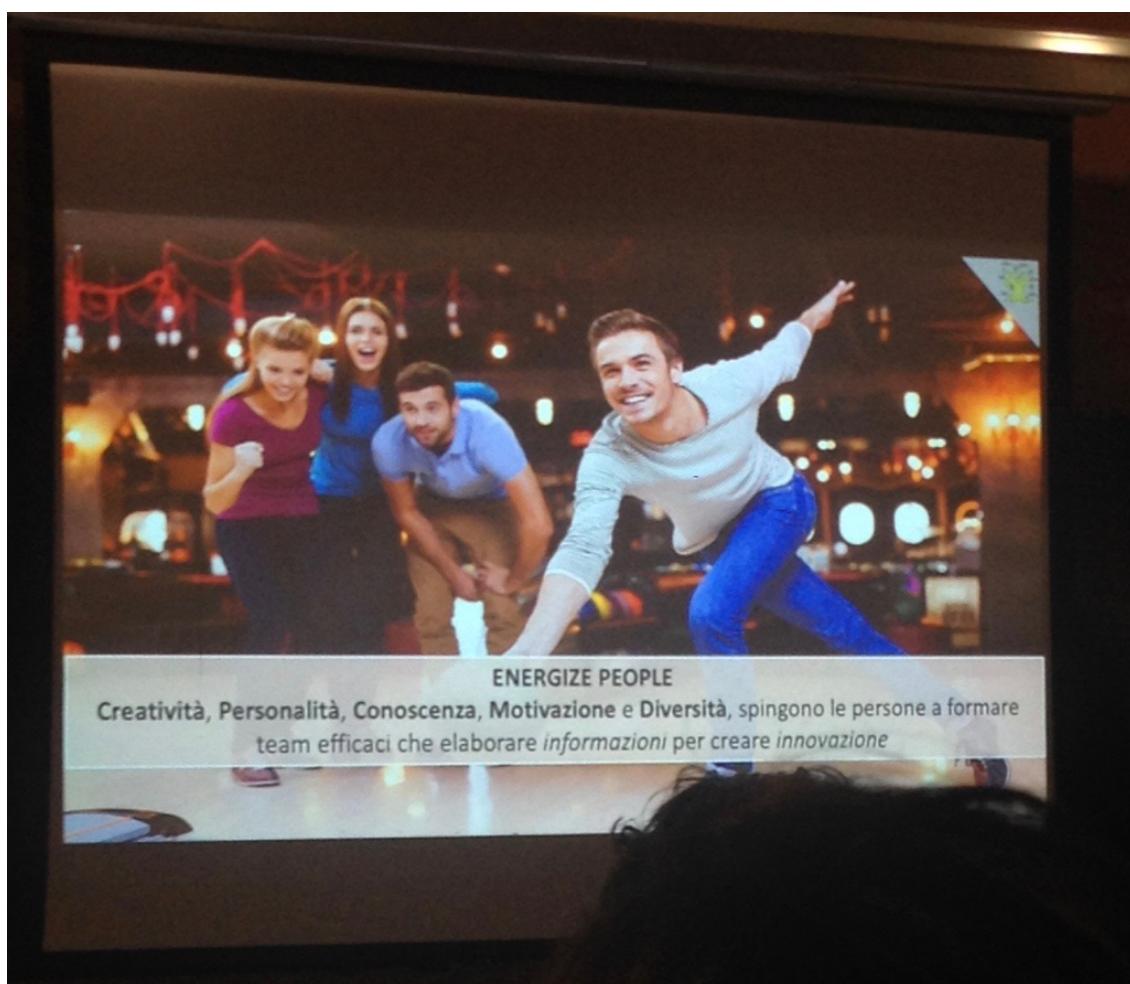
Un team deve essere stabile e non cambiare nel tempo. Il team può proporre le proprie regole, che verranno condivise con manager. Se un manager ha paura o non vuole delegare, può essere “spinto” o convinto da un consulente esterno.

Speaker consiglia di provare ad eliminare la parola manager nel momento di transizione di una azienda verso l’approccio Agile.

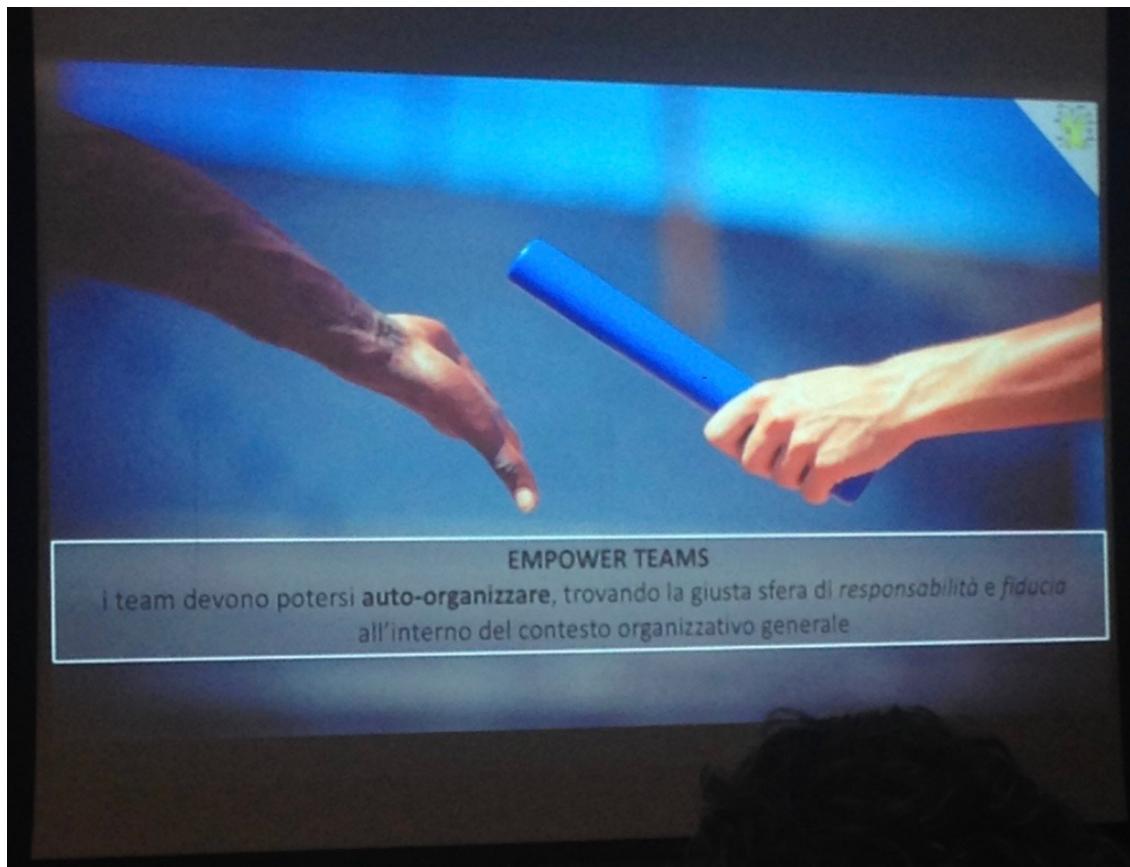
Efficacia non è sempre un fattore vincolante nell’adozione di approccio Agile. Nella foto successiva viene specificato come può essere vista un’organizzazione, 6 prospetti sono:



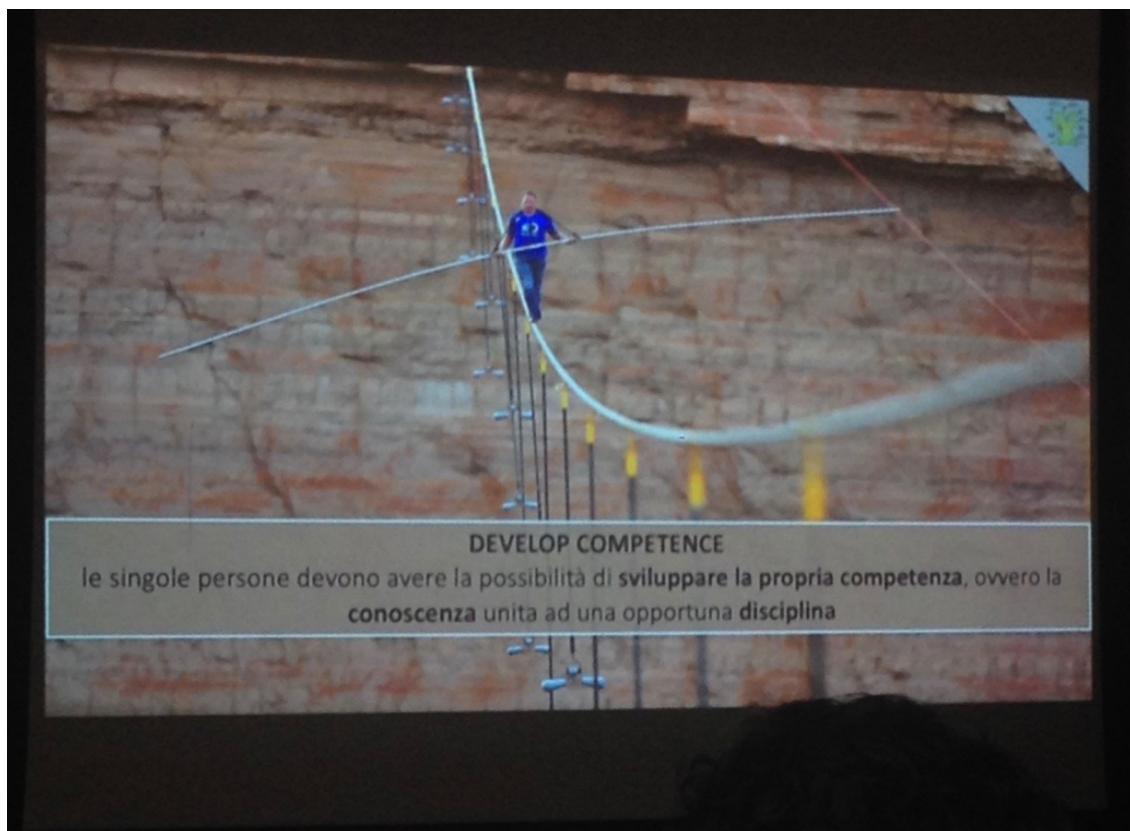
- Energize people - creatività, personalità, conoscenza, motivazione e diversità spingono le persone a formare team efficaci.

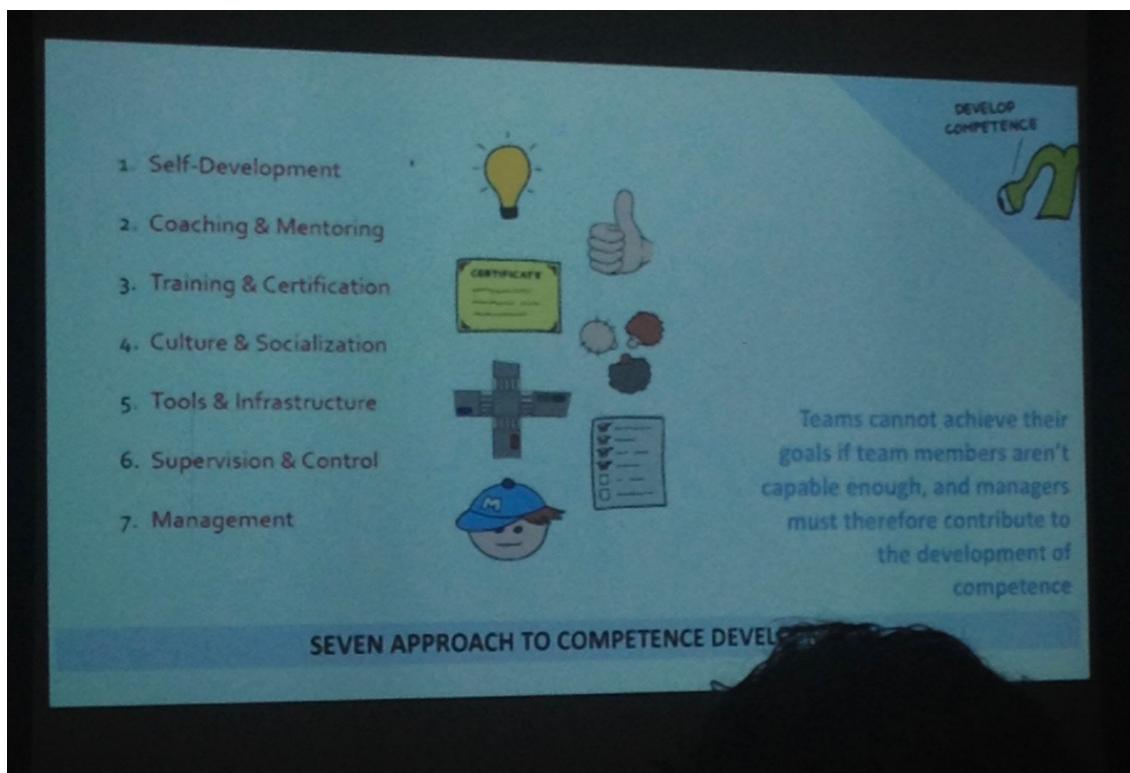


- Empower teams - autorganizzazione di un team

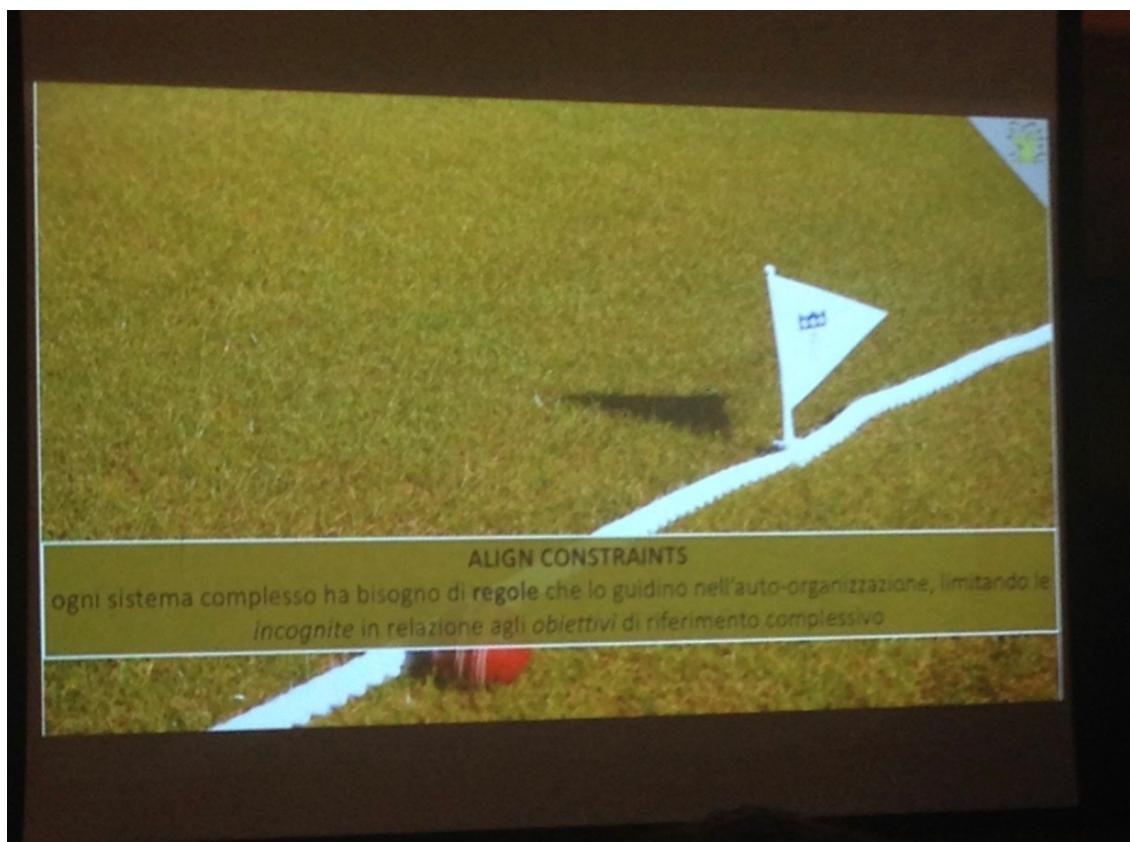


- Develop competence - sviluppo competenze di singole persone

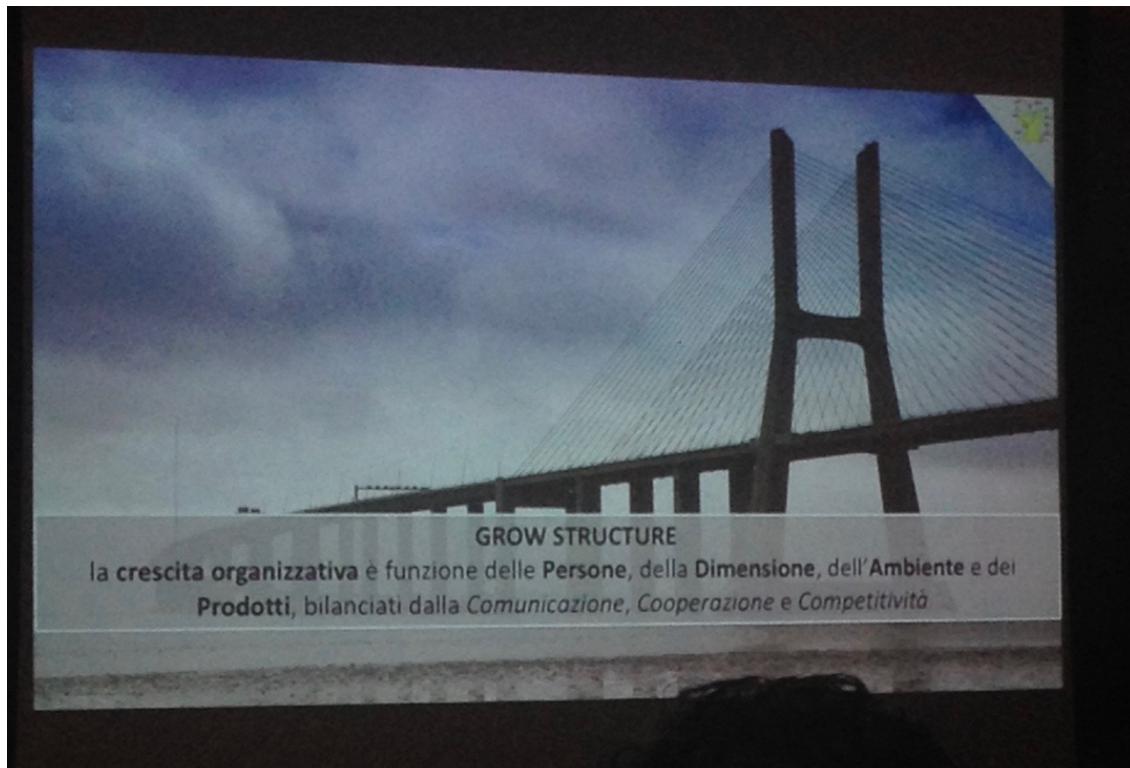




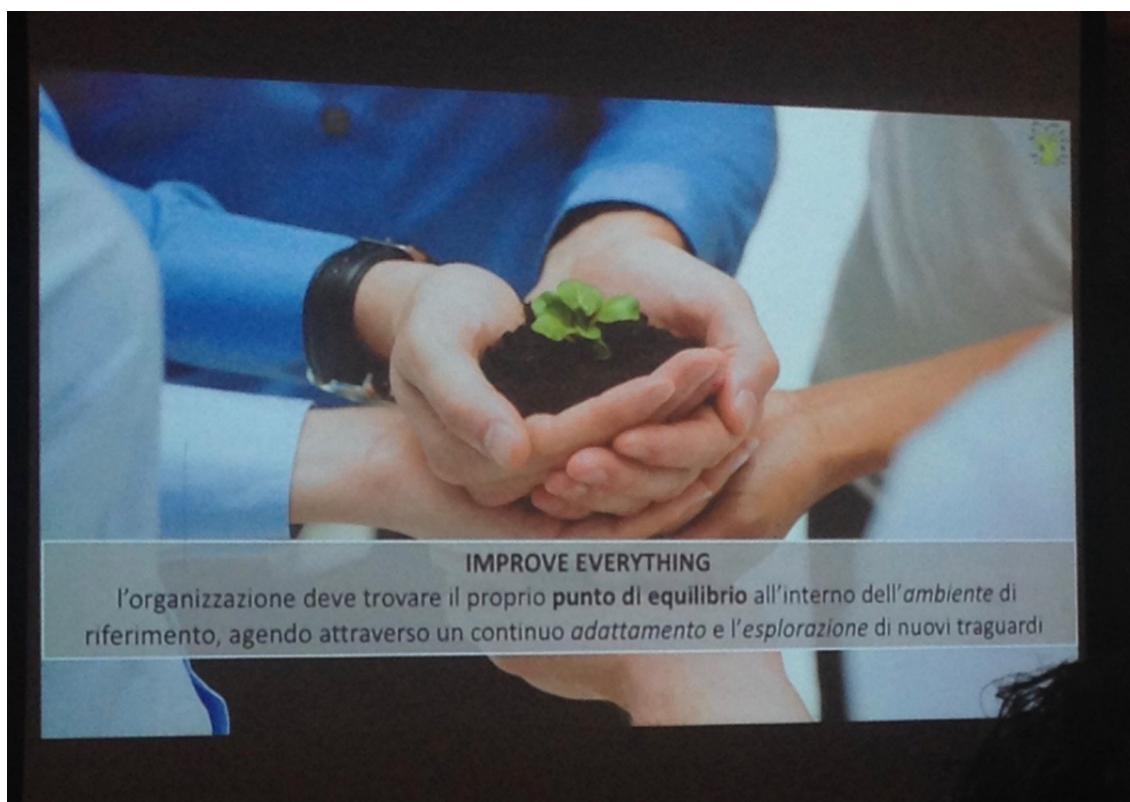
- Align constraints - limitazioni di una persona all'interno di un contesto

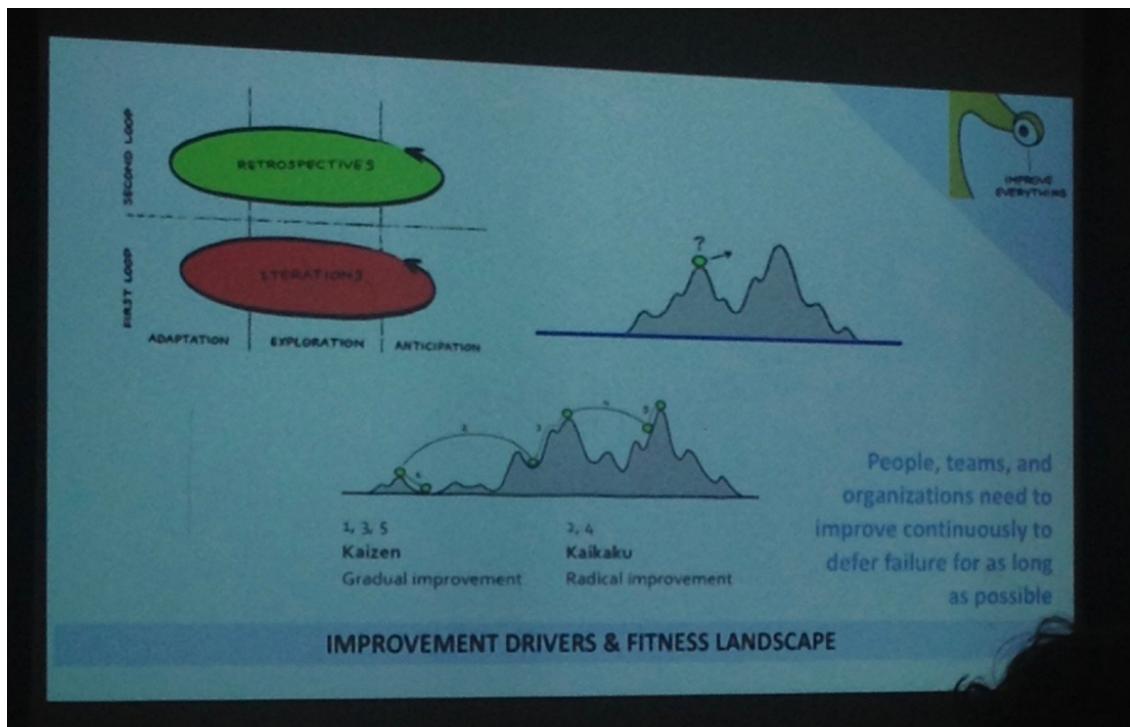


- Grow structure - crescita costante di una azienda

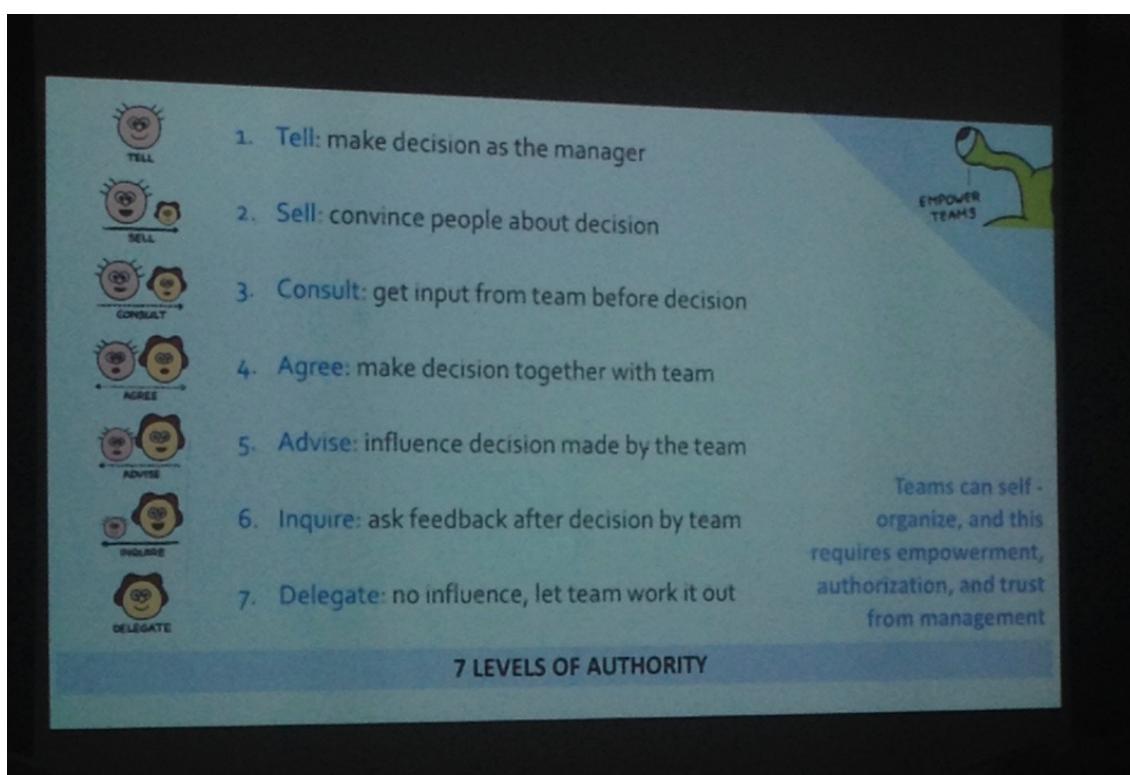


- Improve everything - cercare di migliorare qualsiasi aspetto all'interno di una azienda

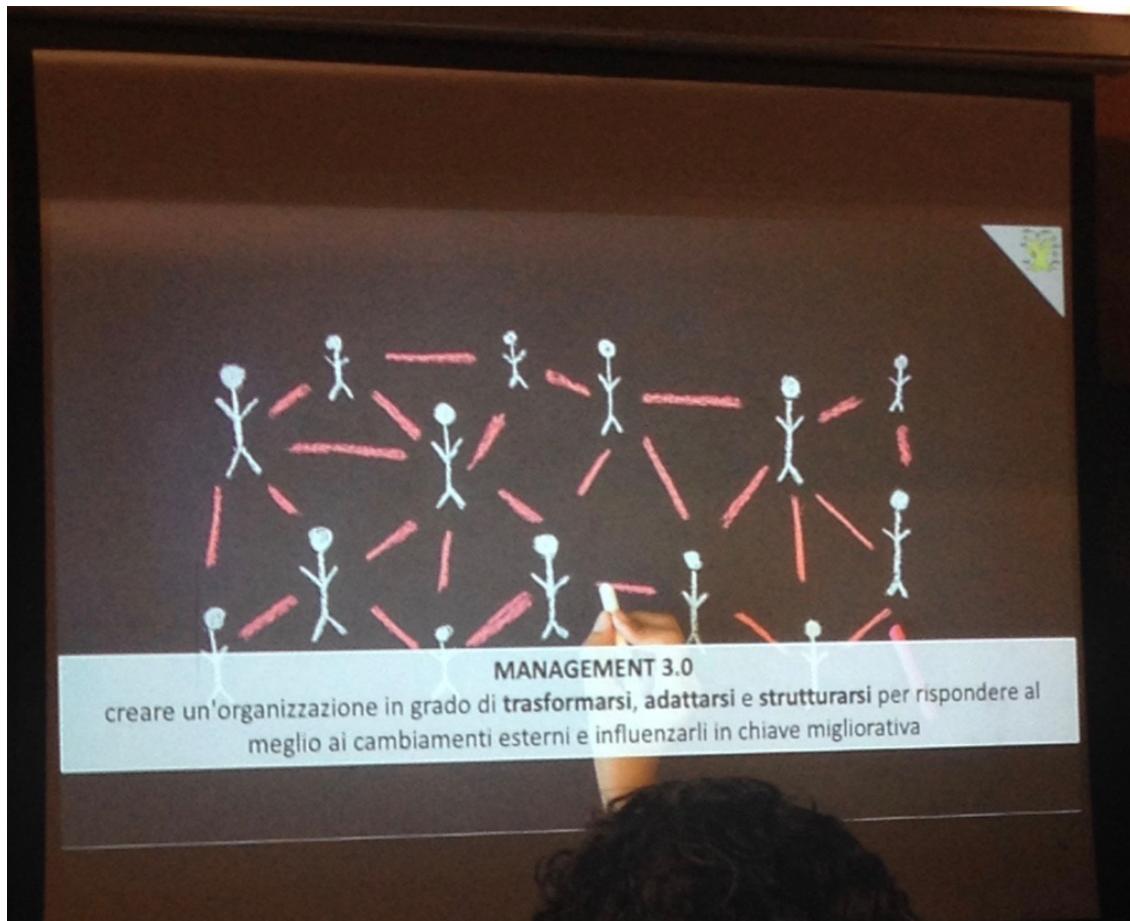




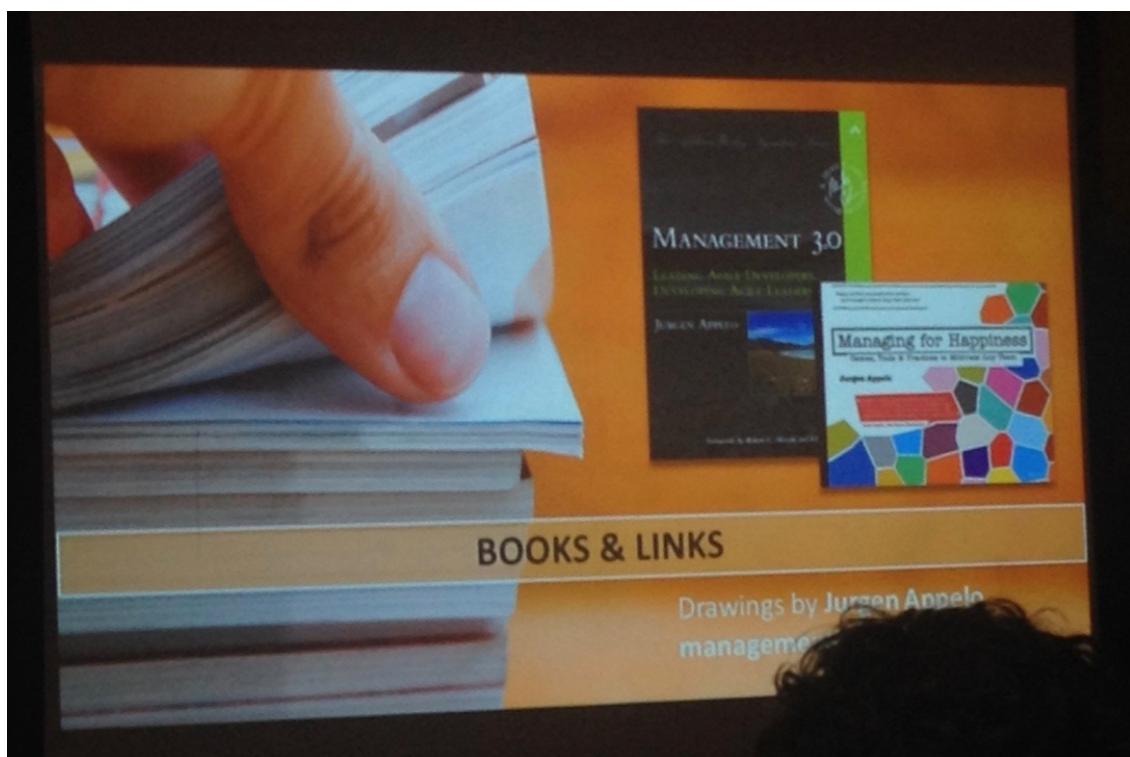
7 livelli di autorità:



Management 3.0:



Libri consigliati:



Business triathlon

Speaker Emiliano Soldi. Ha parlato di come cambiare, innovare, e produrre valore rapidamente, in ambienti sottoposti a costante turbolenza.

Come esempio ha fatto il triathlon (cercate “triathlon iron man” su internet).

Triathlon è composto da tre prove:

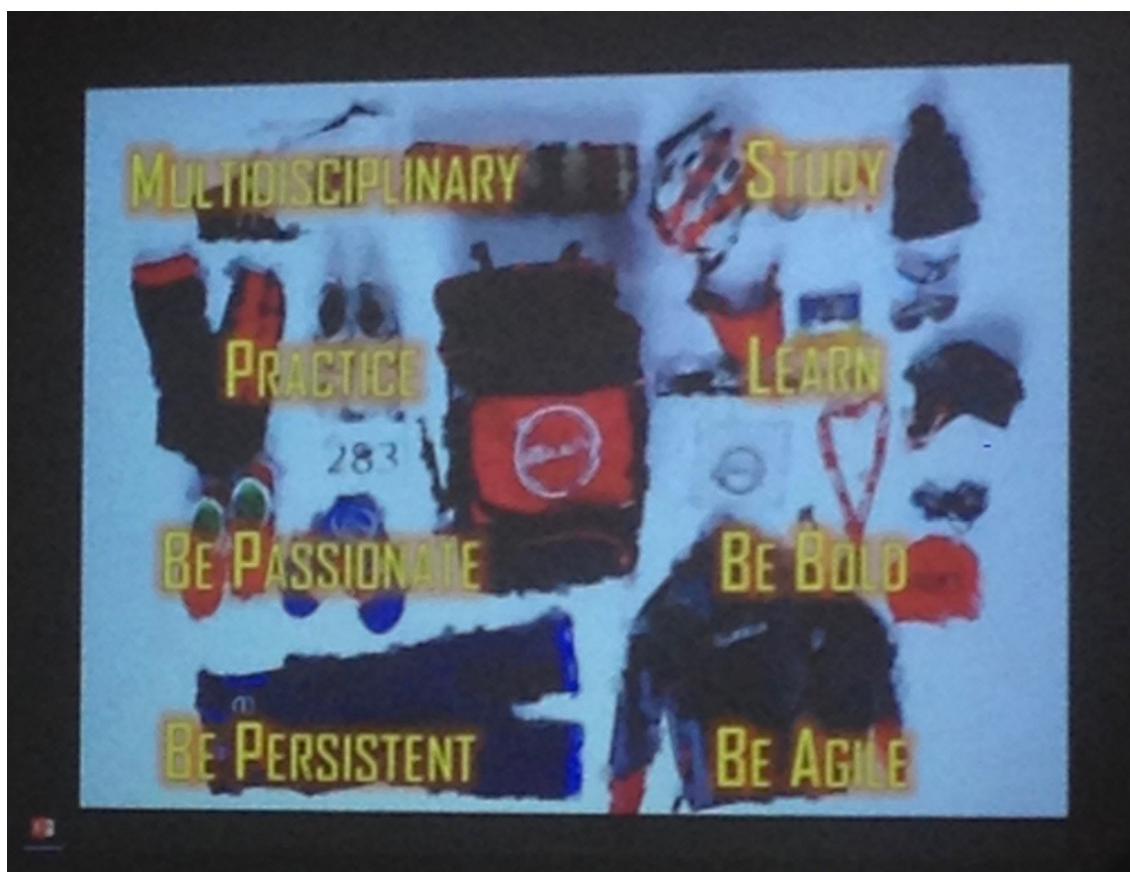
1. Nuoto - in questo momento è importante sviluppare la strategia
2. Ciclismo - qui la strategia cambia di continuo
3. Corsa - metti giù il pragmatismo e crea il valore

Ma, la più importante è la quarta prova - Transizione, il momento di passaggio dal nuoto al ciclismo, dal ciclismo alla corsa. Qui cambia il corpo. E' una fase di adattamento/trasformazione.

In triathlon serve passiamo, coraggio e allenamento.

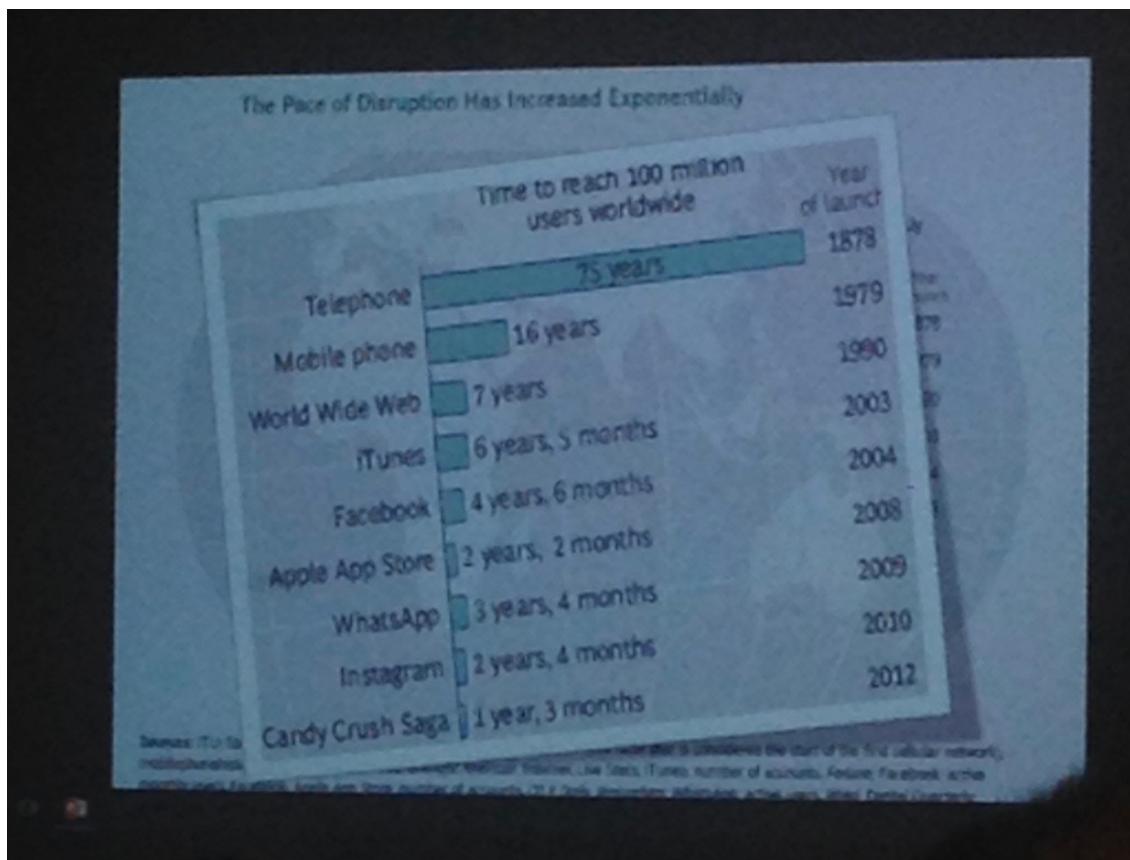
Coraggio di tenere a bada l'ansia. Allenamento è importantissimo.

Adesso pensaci un'attimo come tutte queste caratteristiche si combaciano con il passaggio di una azienda ai processi Agile.



Aziende, grande realtà (quotate anche in borsa) di solito chiedono “Come faccio ad applicare agile e non fallire?”. Prima si studia la strategia. Dopo si studiano le iniziative. Qui importante “mettersi nelle scarpe” dei clienti. Cosa dobbiamo inventare - visione ad alto livello.

I ritmi sono aumentati in modo esponenziale:



Le domande sono:

THE PROBLEM

How can we strongly link
STRATEGY vs **EXECUTION**

How can we match
DEMAND vs **CAPACITY**

How much effort should we dedicate
RUN BIZ vs **CHANGE BIZ**

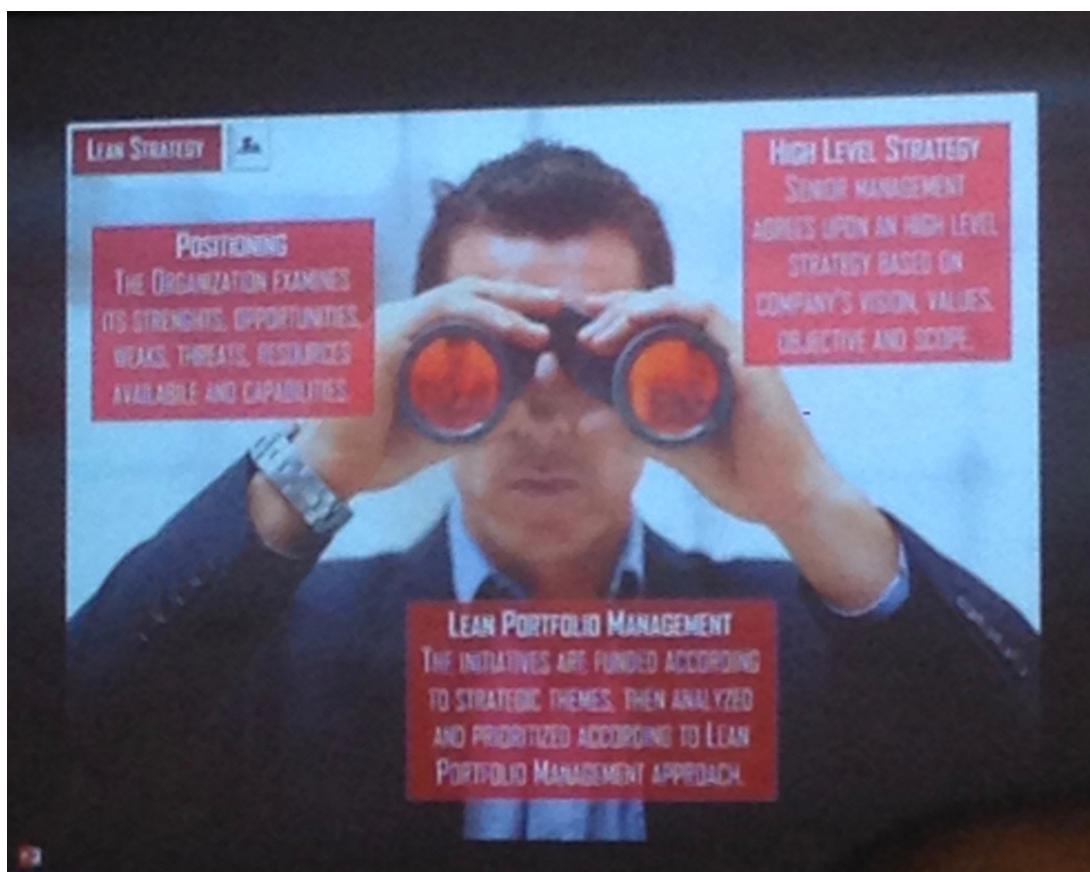
THE SOLUTION

Can the **LEARNING** from **TRIATHLON**,
HELP to deliver better **BUSINESS**
RESULTS in this **TURBULENT** world?

Paragoniamo il triathlon con il business a approccio da adottare:

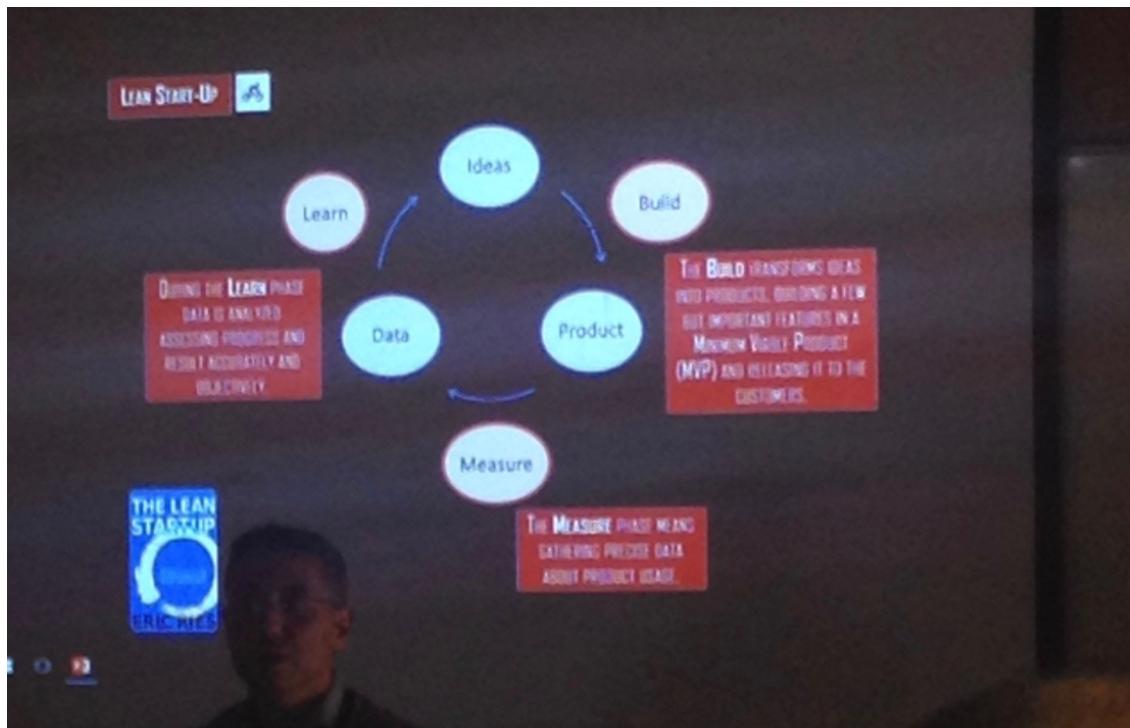


Positioning, hight level strategy e lean portfolio management:



Non vi ricorda niente l'immagine precedente? :)

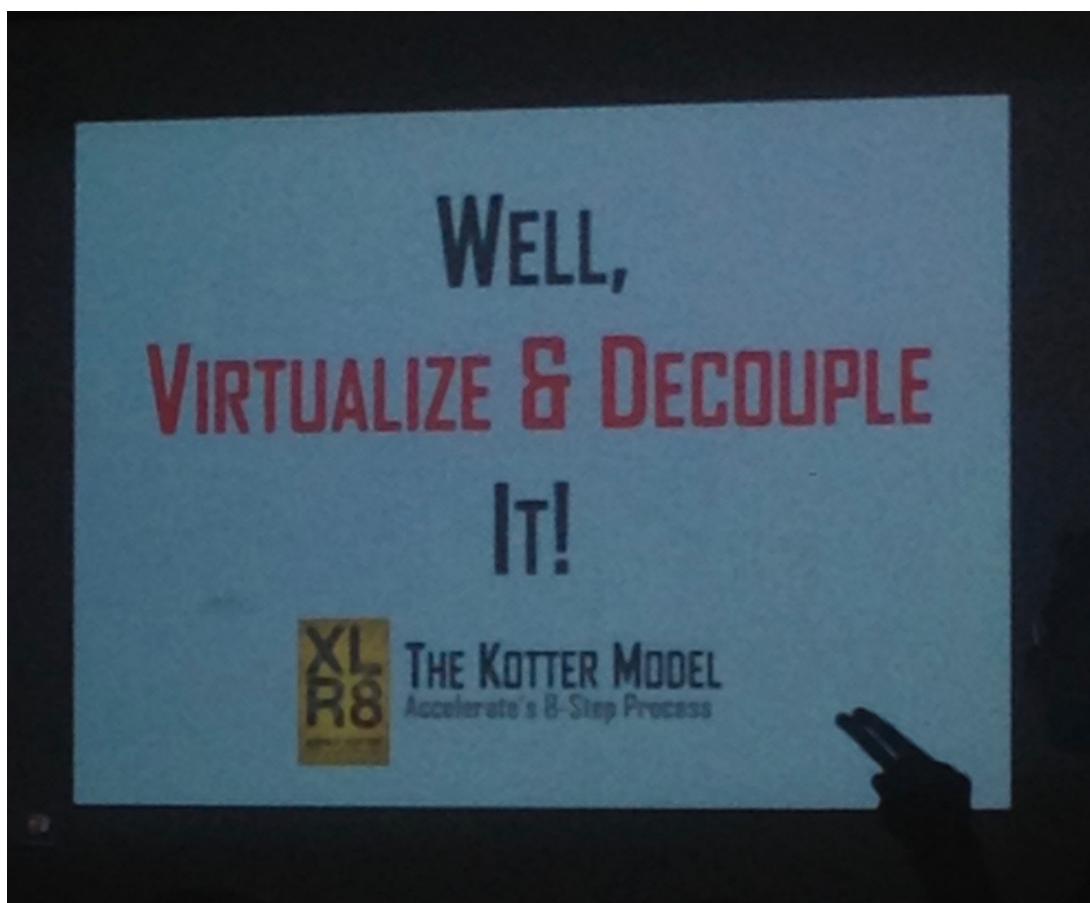
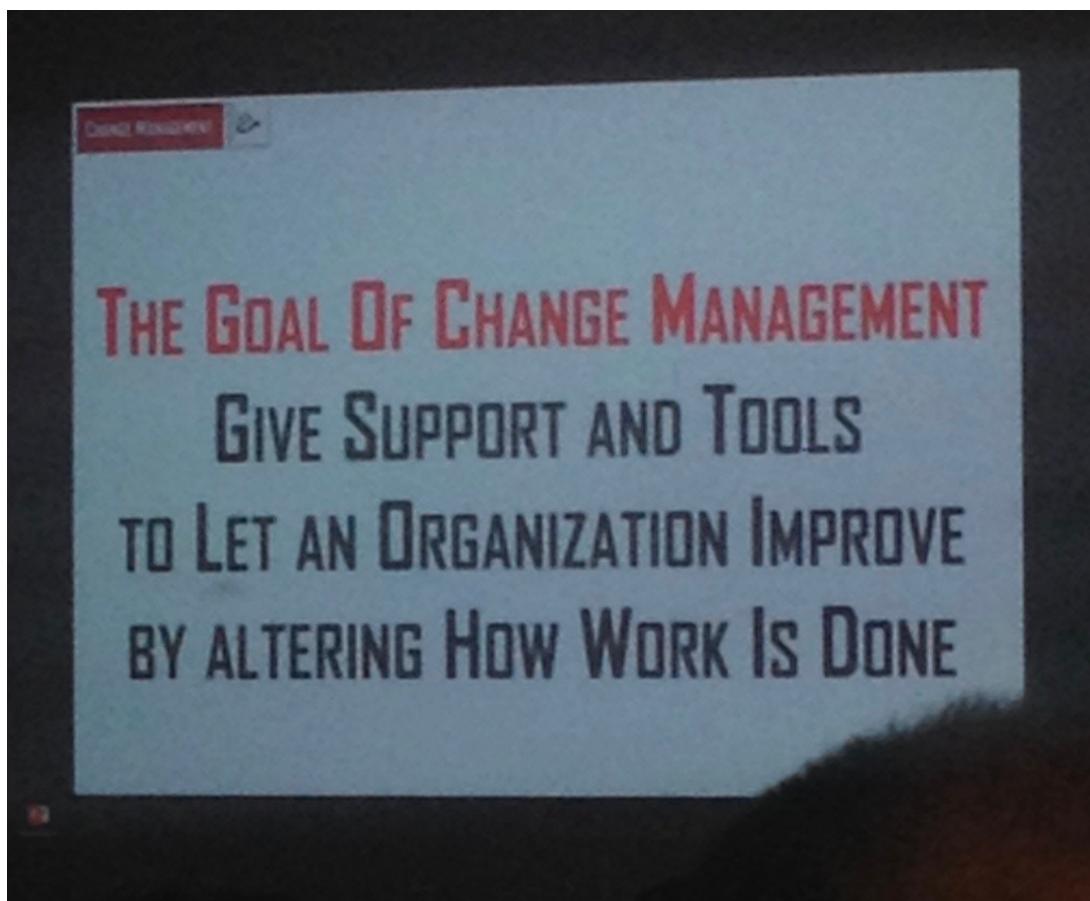
Il flusso:



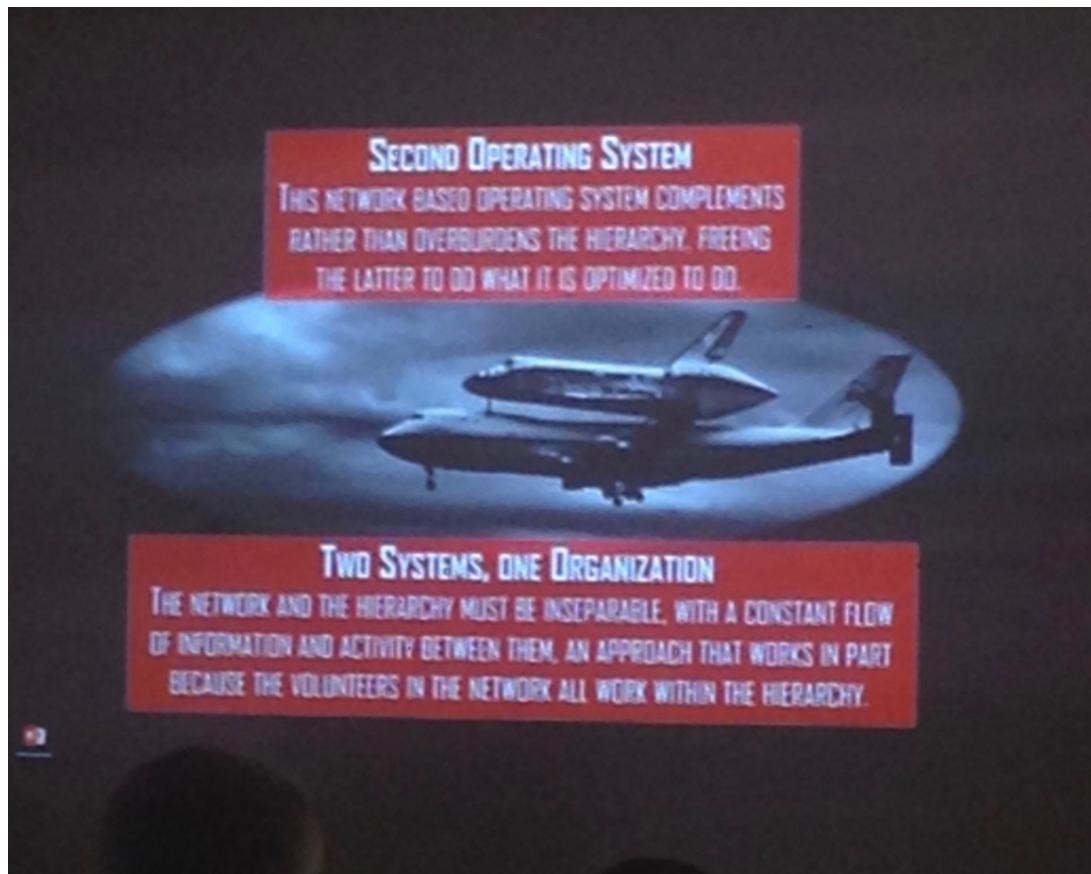
Sviluppo del prodotto iterativo e incrementale e i valori:



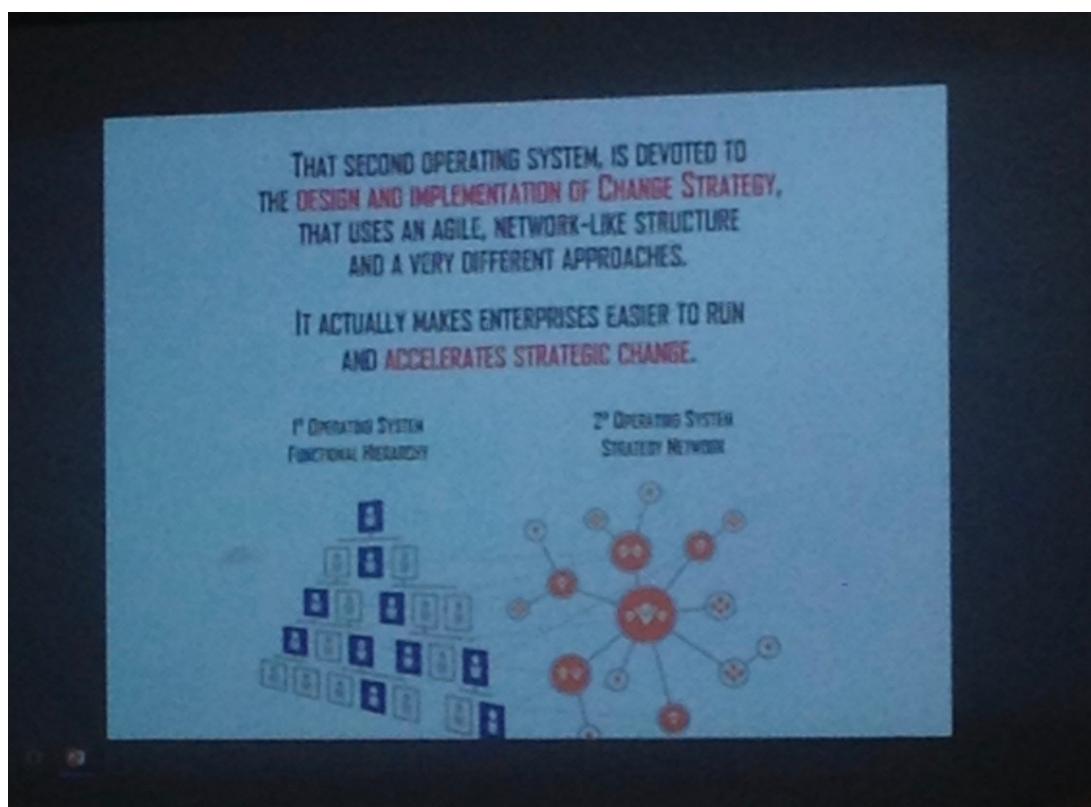
Obiettivo di change management:



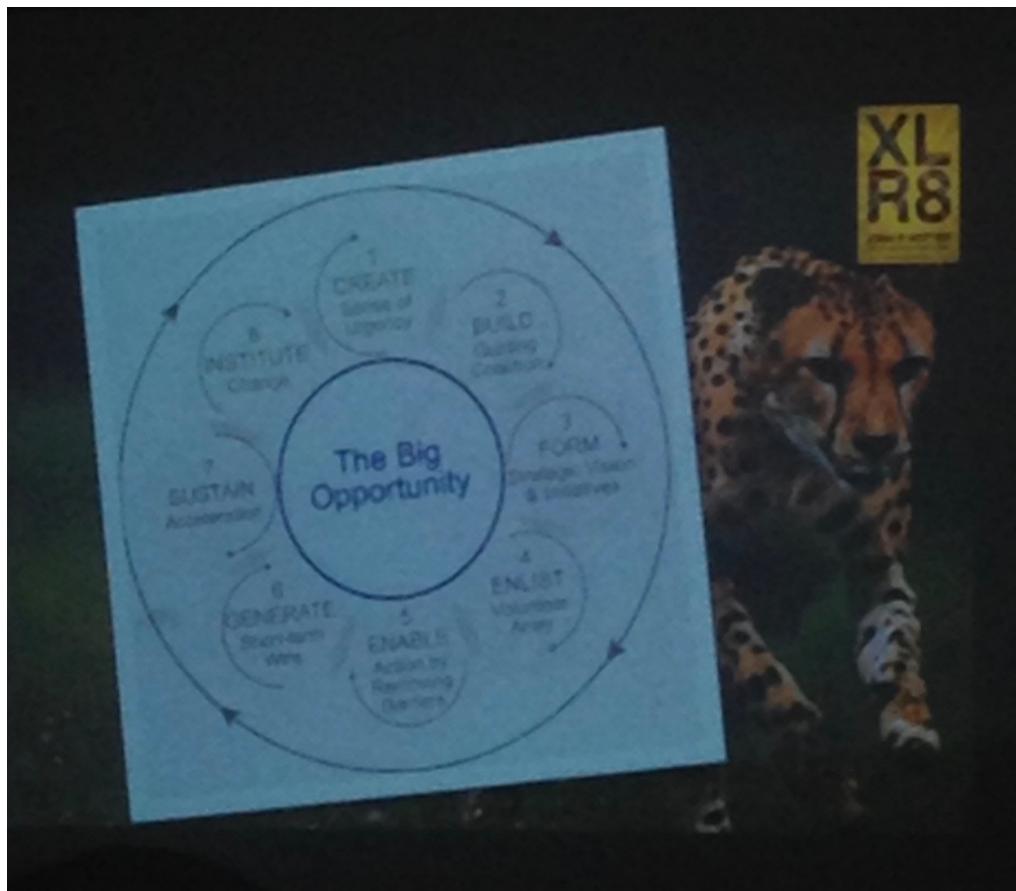
Secondo sistema operativo - come possibile soluzione per portare una grande azienda verso l'approccio Agile. Molto interessante!



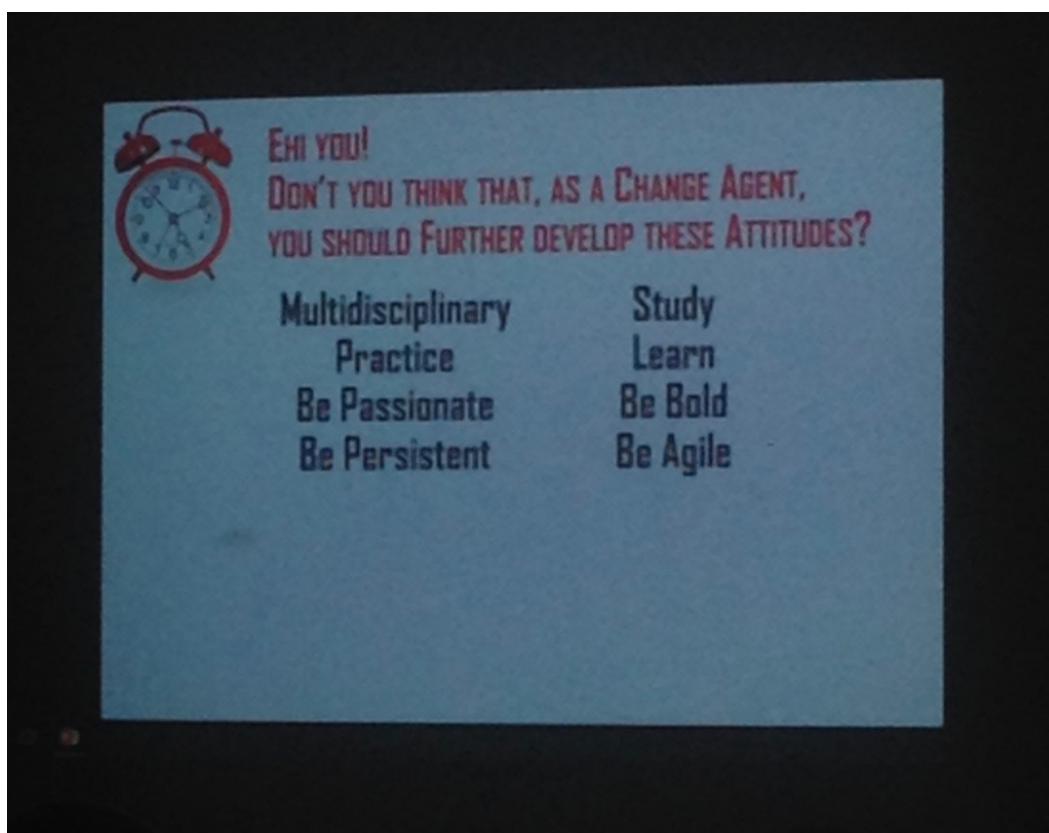
Che cos'è un secondo sistema operativo:



The big opportunity:



Skill di una persona che porta il cambiamento in azienda:



Si è parlato molto di lean startup. Si studia l'idea. Dopo si studia il passaggio dall'idea al prodotto. MVP - minimum valuable product (prima di questa conferenza conoscevo questa sigla come Most Valuable Player). MVP viene fatto per creare il valore. Per vedere i numeri concreti.

Il secondo sistema operativo menzionato prima è una rete esterna che vuole cambiare l'azienda. Si appoggia sul primo sistema operativo, quello principale. Il secondo sistema funziona/opera sotto il continuo monitoraggio.

Pausa pranzo



Grazie agli organizzatori per il lunch!

Keynote di pomeriggio

Speaker Andrea Provaglio.

Ecco il video:

<https://www.youtube.com/watch?v=TWICc32UJms&feature=youtu.be&t=5h9m47s>

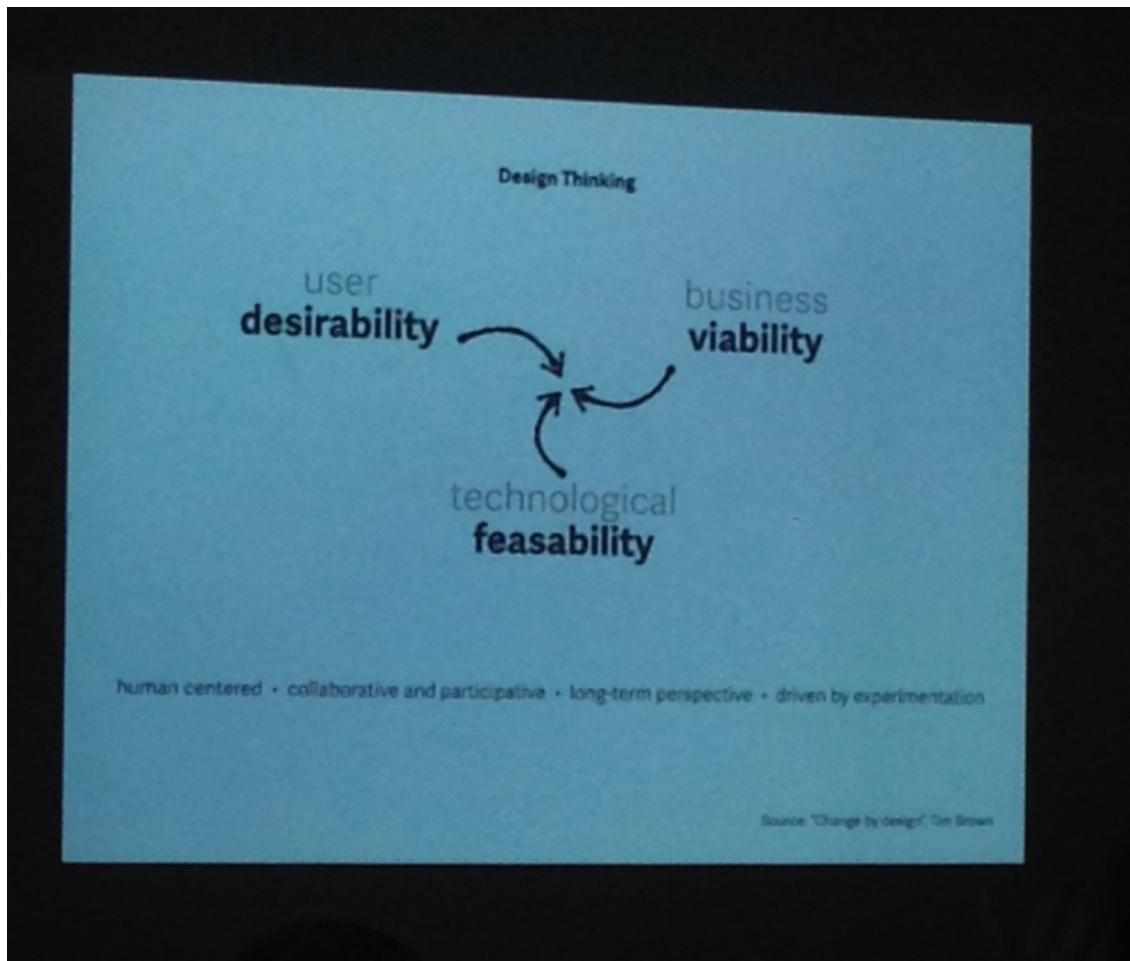
Introdurre design thinking in un processo agile: perché, che
impatto ha sul business, come fare

Speaker Ilaria Mauric.

E' un designer in una azienda che cerca di seguire l'approccio agile nel proprio lavoro. Amica di Manuel Spezzani :) (@manuel, abbiamo visto una tua foto nella sua presentazione, non ho fatto in tempo a fare la foto con il cell).

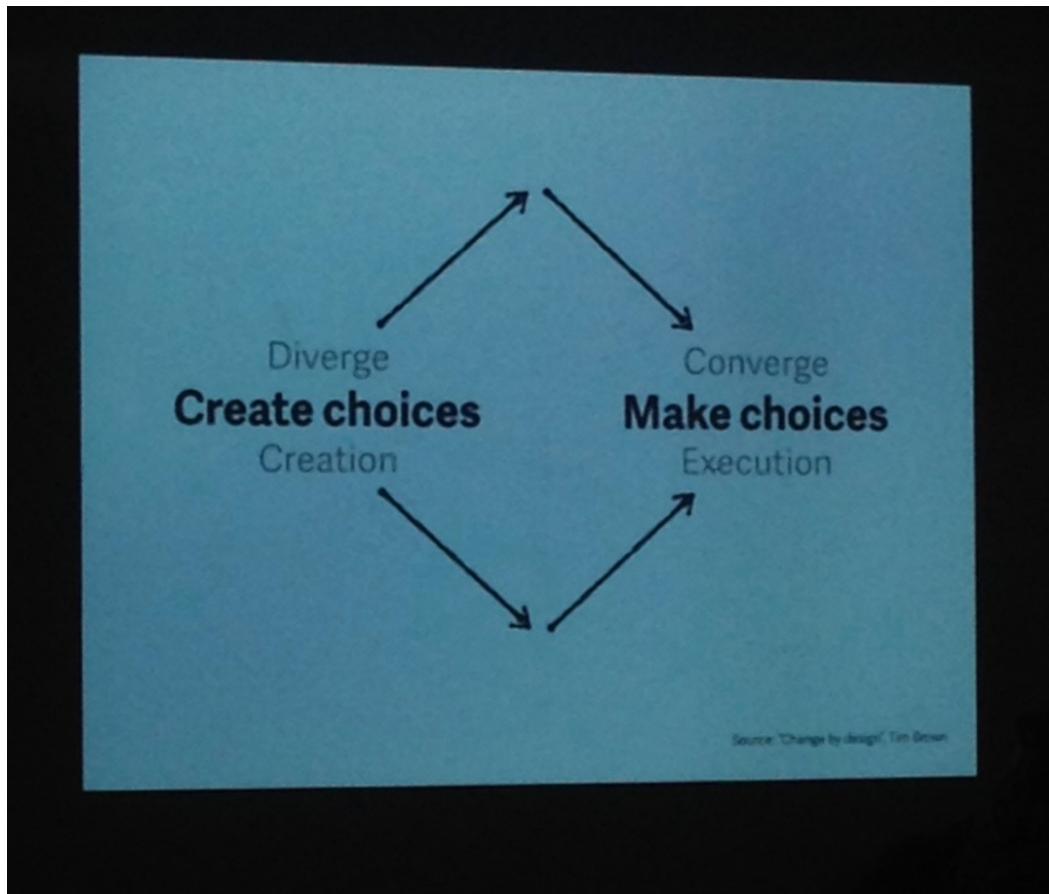
La cosa importantissima è di pensare prima di fare qualsiasi cosa.

Framing come divisione del contesto, dividere qualcosa di grande in parti più piccoli.

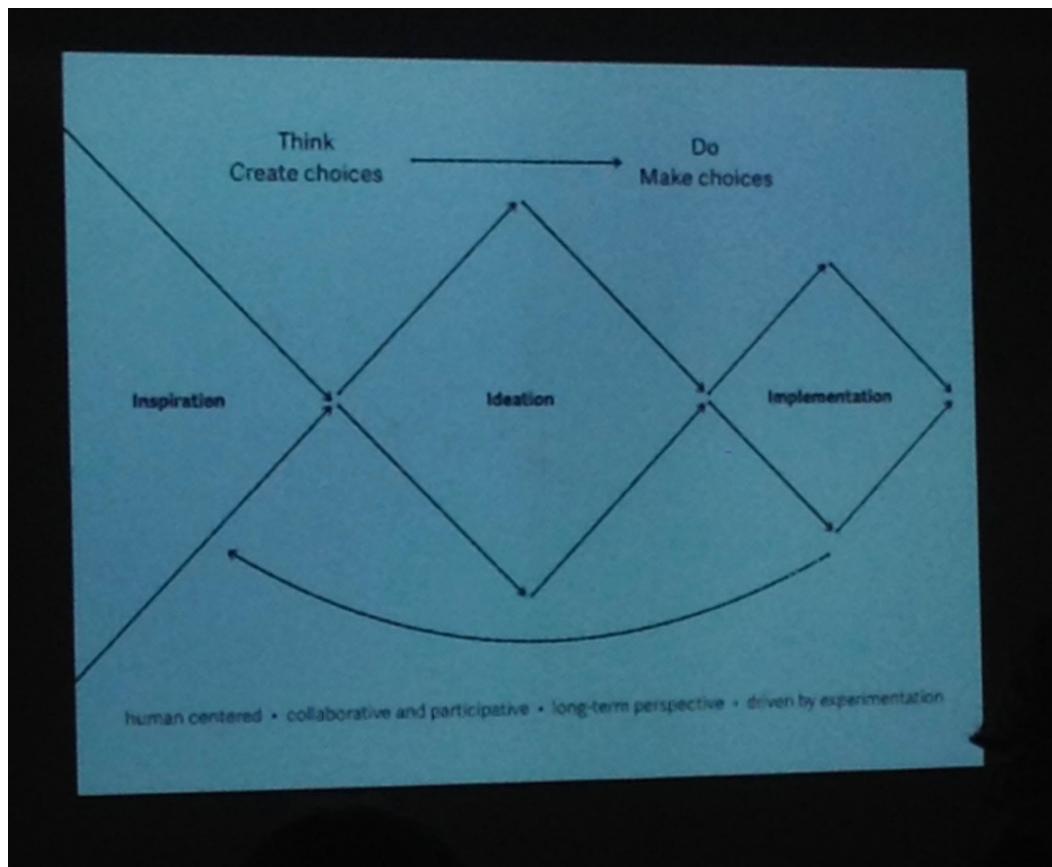


Nella foto precedente sono tre cose da valutare nel momento di analisi di un idea.

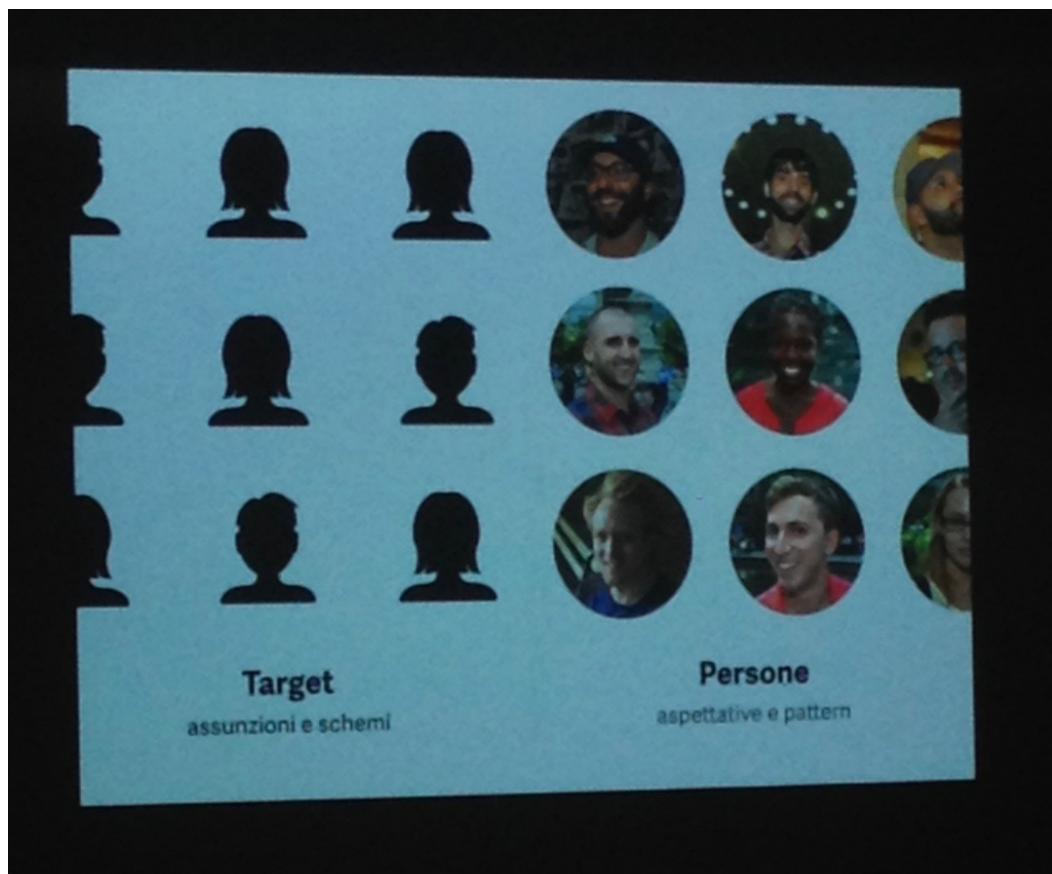
Divergenza, per creare le possibilità di scelta, convergenza, per prendere le decisioni.



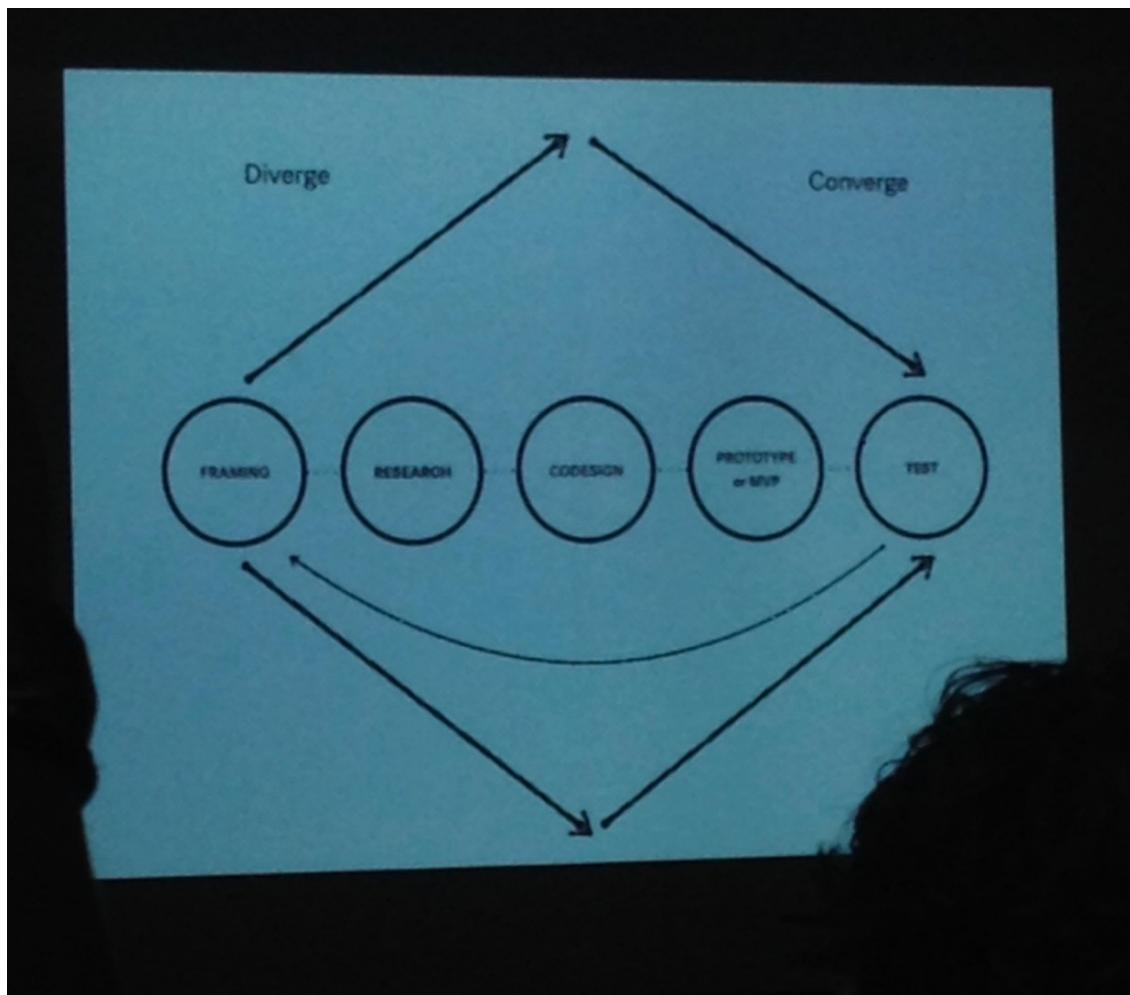
All'inizio di un idea abbiamo ispirazione, dopo avviene ideazione e chiude l'implementazione.



Importante conoscere le aspettative e pattern delle persone interessati:



Le fasi principali sono: framing (divisione del contesto), research (ricerca, può



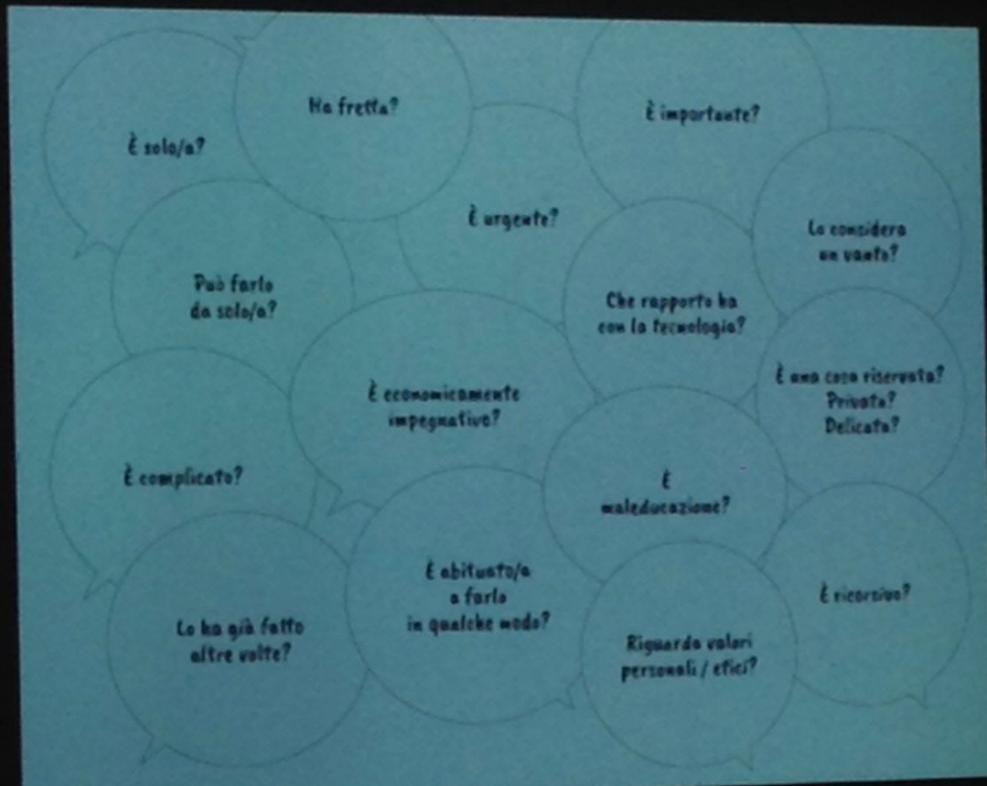
essere posticipata alla fine), codesign, prototype or MVP, Test.

Lo schema dipende molto dal cliente, qualche fase può saltare (per esempio ricerca).

Le domande alle quali dobbiamo rispondere:

Who?
What?
Why?
When?
Where?

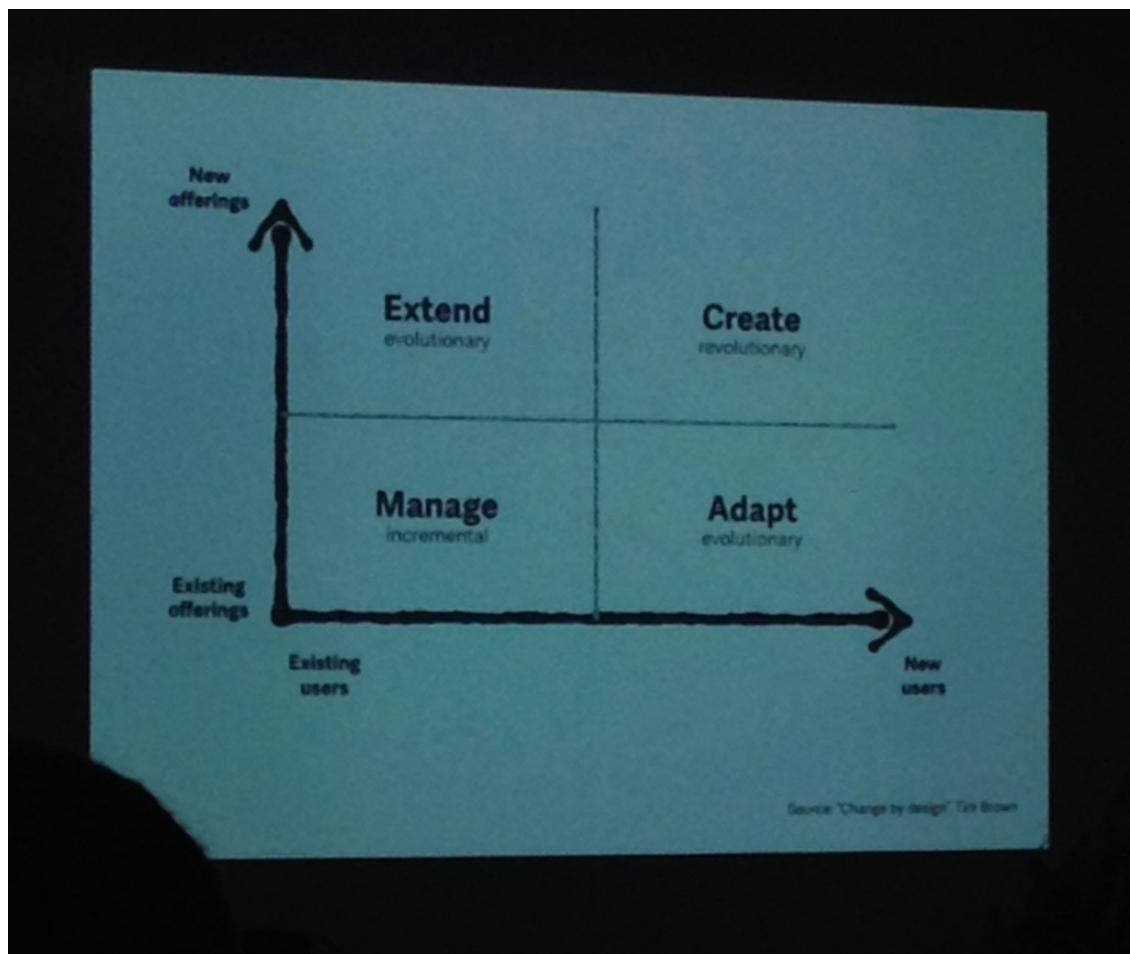
How?



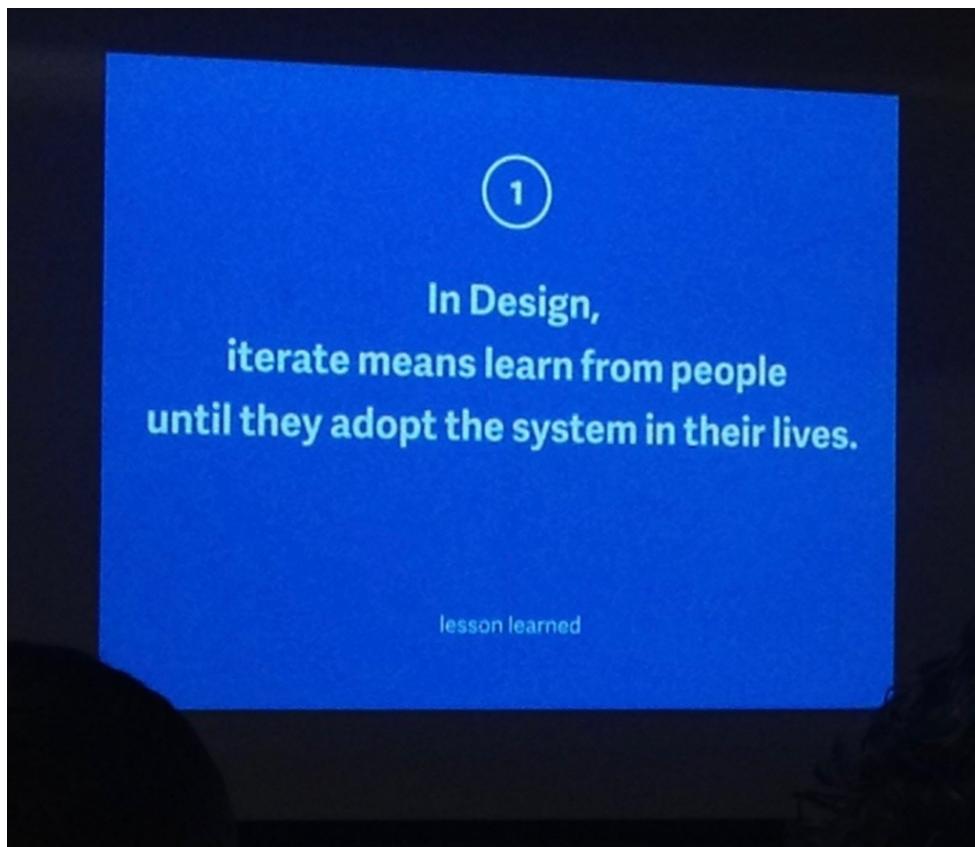
Speaker ha fatto seguenti esempi, esperienza che hanno avuto durante loro attività professionale.

- Sono stati chiamati alla fine di uno sviluppo - molto difficile apportare i cambiamenti
- Sono stati chiamati all'inizio del progetto con tempi molto stretti - 2 mesi per fare i template. Come prima cosa sono stati rinegoziati i tempi. Il primo mese è stato un periodo di ricerca. Lo scopo principale era capire come lavorano gli utenti. Qui importante avere delle risposte NON condizionate alle domande. Era un punto di vendita, dovevano intercettare le persone sul campo per capire come stanno operando attualmente.
- Sono stati chiamati in una azienda che lavora 100% agile (ex azienda di Manuel :)). L'approccio ha funzionato completamente.
- Quando chiamano all'inizio di una idea - complicato fare la pianificazione a lungo termine. Di solito si chiede al cliente cosa si aspetta tra 6-9 mesi. Dopo questo periodo si tenta di avere un idea per prossimi 2-3 anni.

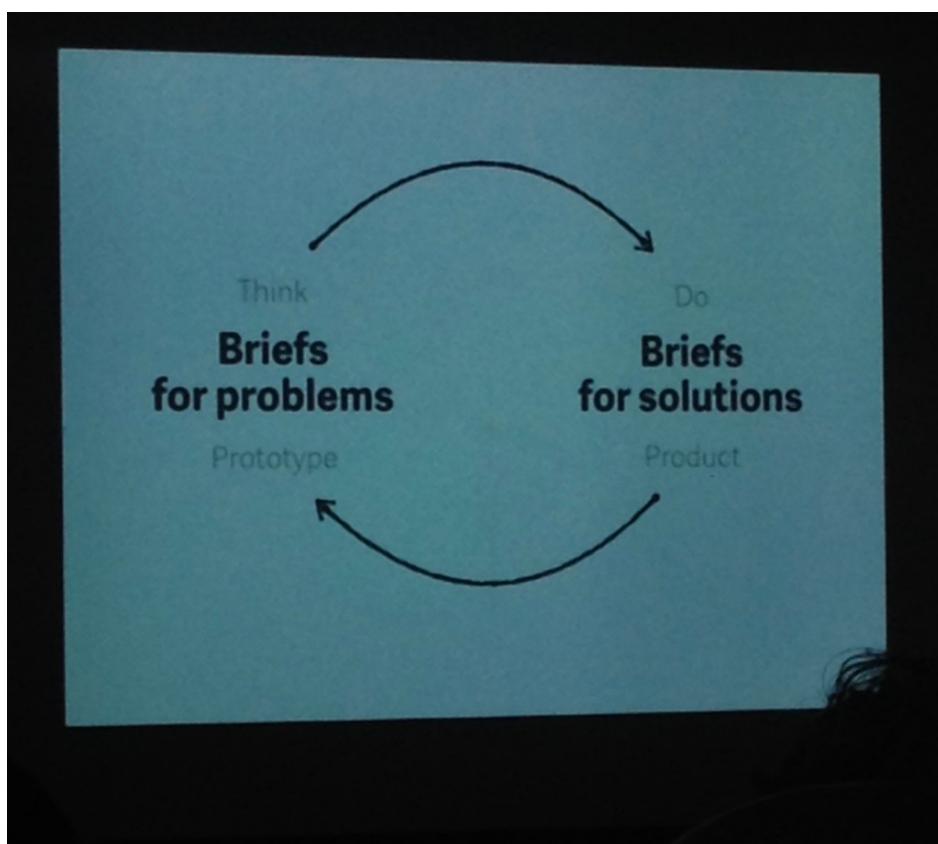
Uno schema che mostra il legame tra gli utenti e cambiamenti:



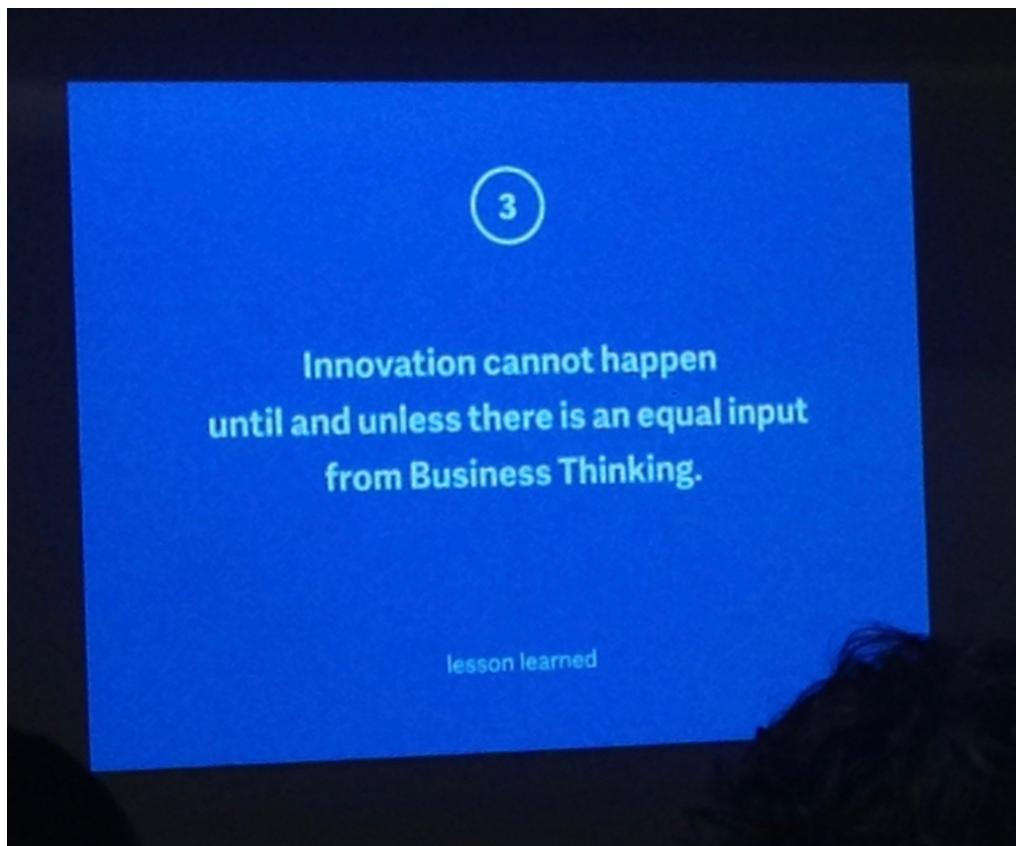
Significo di iterazioni in design:



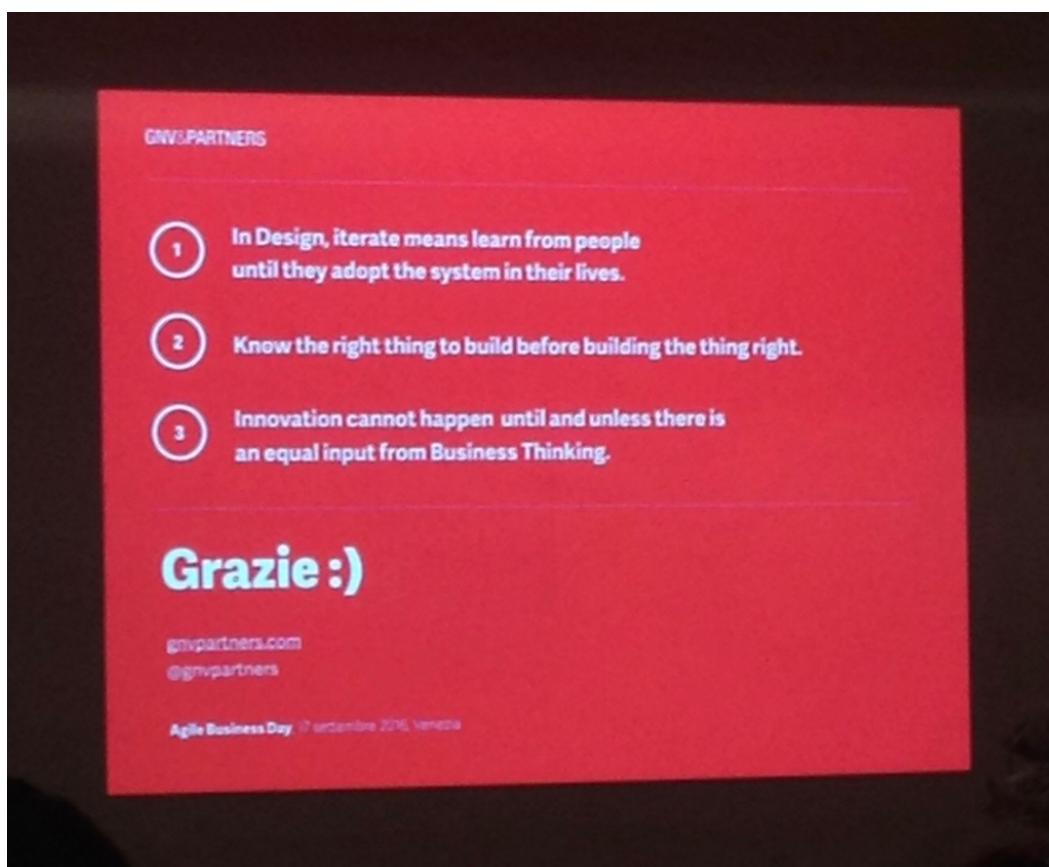
Pensa - Fai:



NOTA:



Riepilogo:

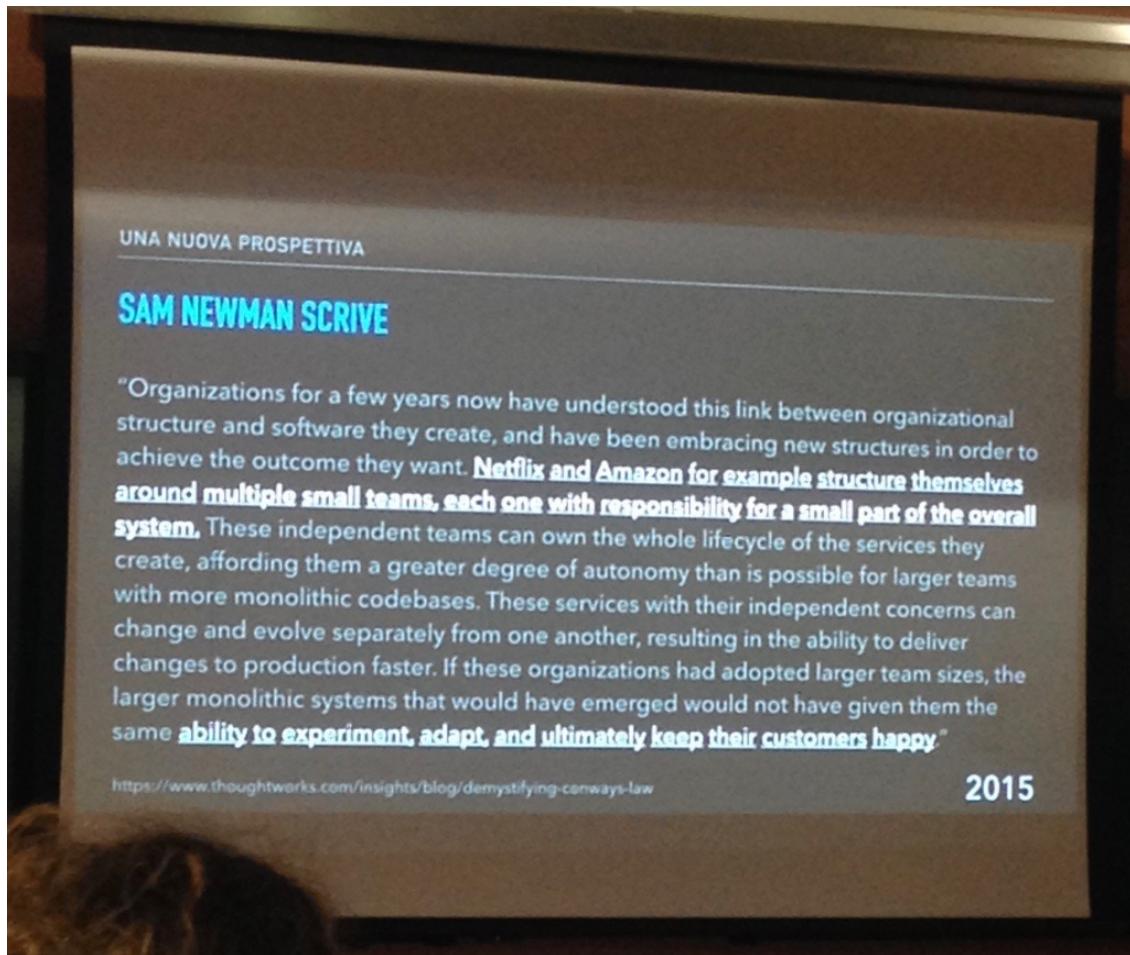


Favorire i “feature teams” con architetture a microservizi

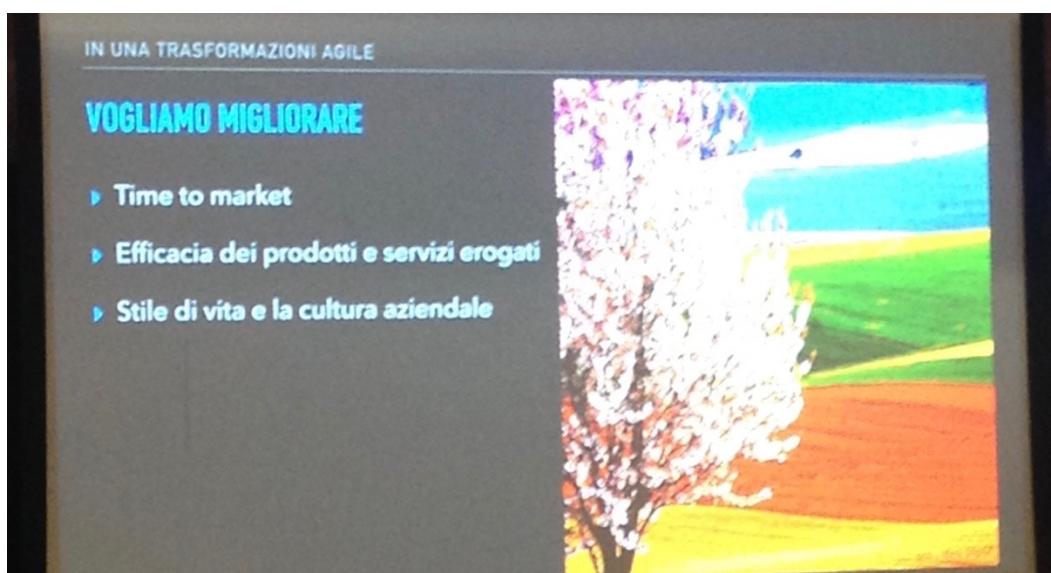
Speaker Giulio Roggero.

Ha parlato di come strutturare un team di lavoro. Molto interessante come idea!

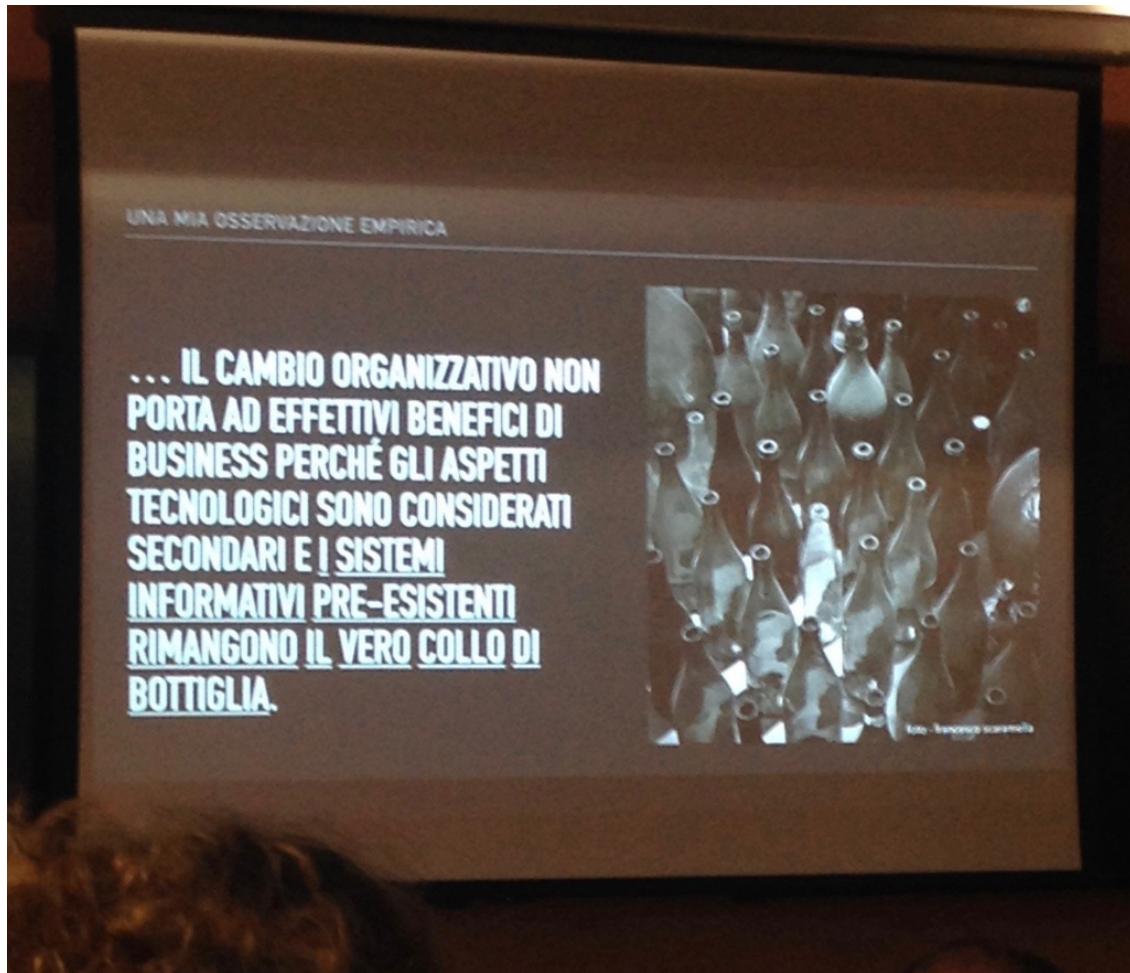
Favorire i team piccoli:



Cosa vogliamo:

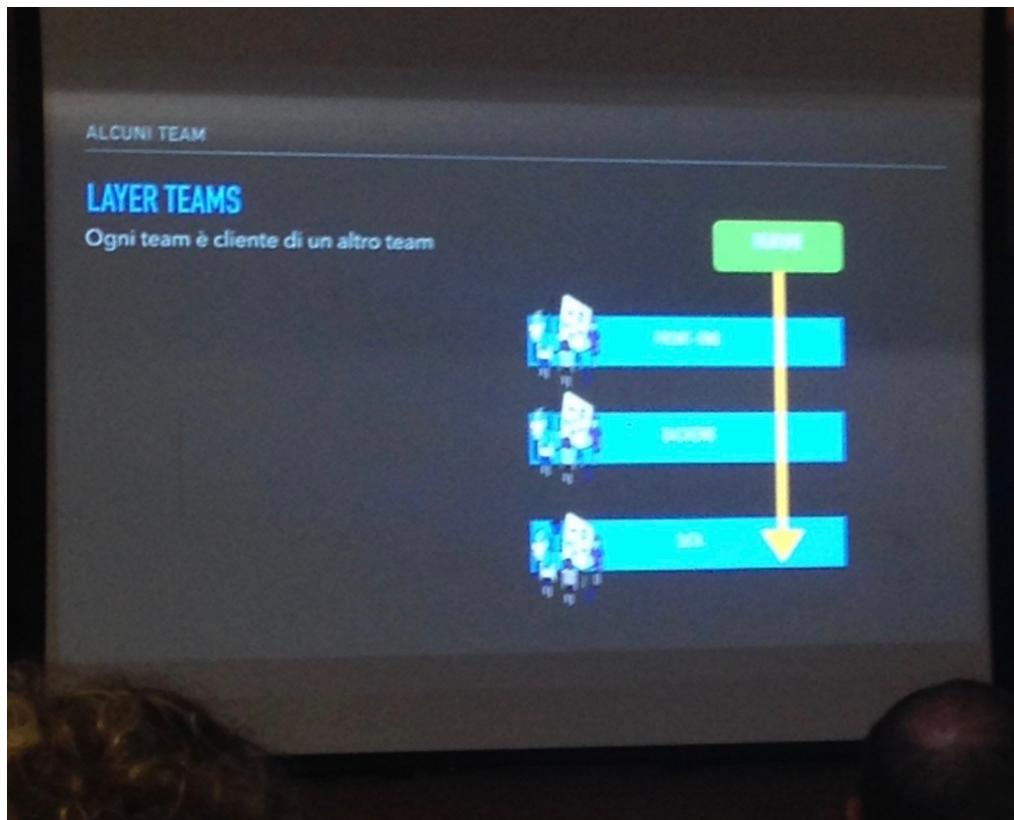


Il problema vero:



Possiamo avere i seguenti team:

- Layer teams

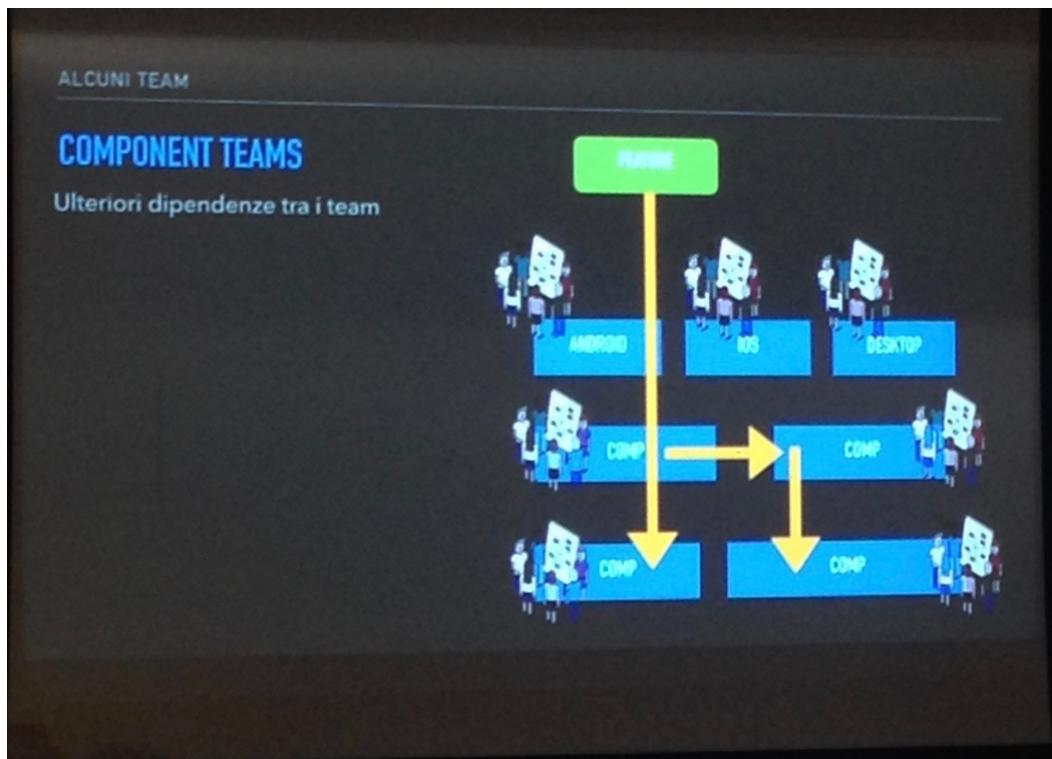


Nella foto non si vede bene, i layer sono: front-end, back-end, dati. In verde è Feature.

Qui ogni team è il cliente di un altro team. Tre diversi team devono collaborare per implementare una nuova funzionalità. Come conseguenza abbiamo la lentezza nella collaborazione. Se nella nuova funzionalità emergono i problemi i team potranno scaricare la responsabilità sugli altri team.

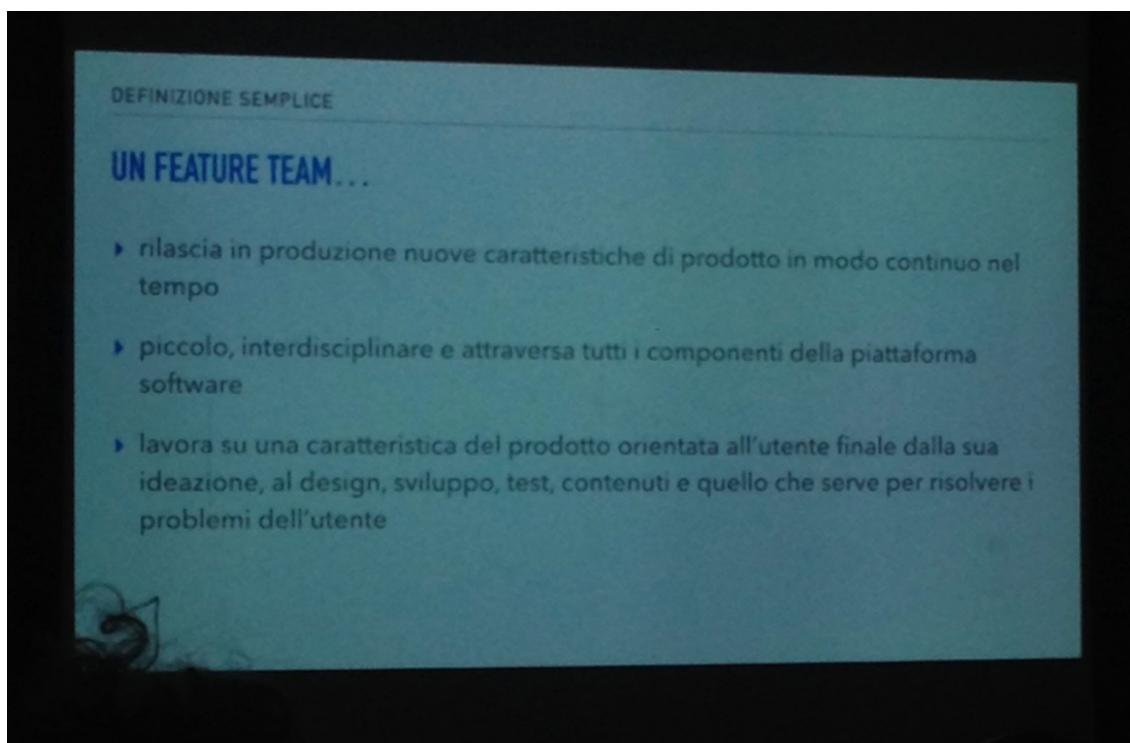
- Component teams

Troppi dipendenze tra i team. Se una nuova funzionalità deve “toccare” più componenti, dobbiamo coinvolgere più team.

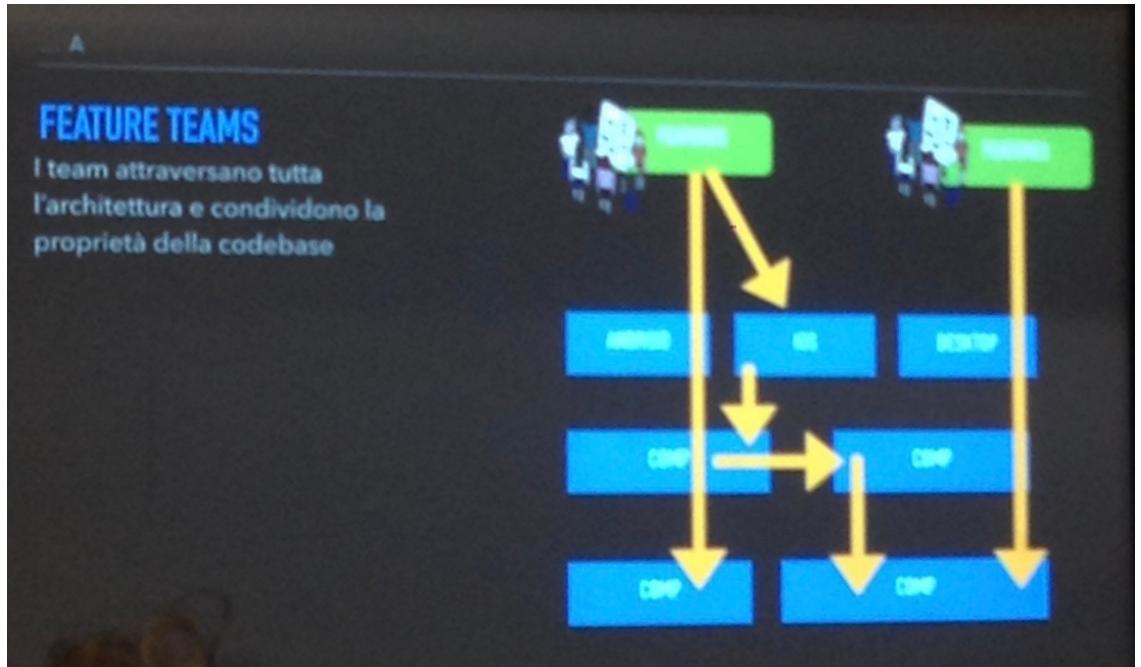


- Feature teams

Team per funzionalità. Benefici sono:



Feature team è orientato al cliente finale. Il team è responsabile di quello che rilascia.



Che cos'è un architettura a microservices:

DEFINIZIONE

UN'ARCHITETTURA A MICROSERVICES ...

- ▶ è un uno stile architetturale
- ▶ è un'applicazione composta da piccoli e indipendenti processi che comunicano tra di loro utilizzando API indipendenti dal linguaggio
- ▶ ogni microservice è fortemente disaccoppiato dagli altri e svolge un compito preciso



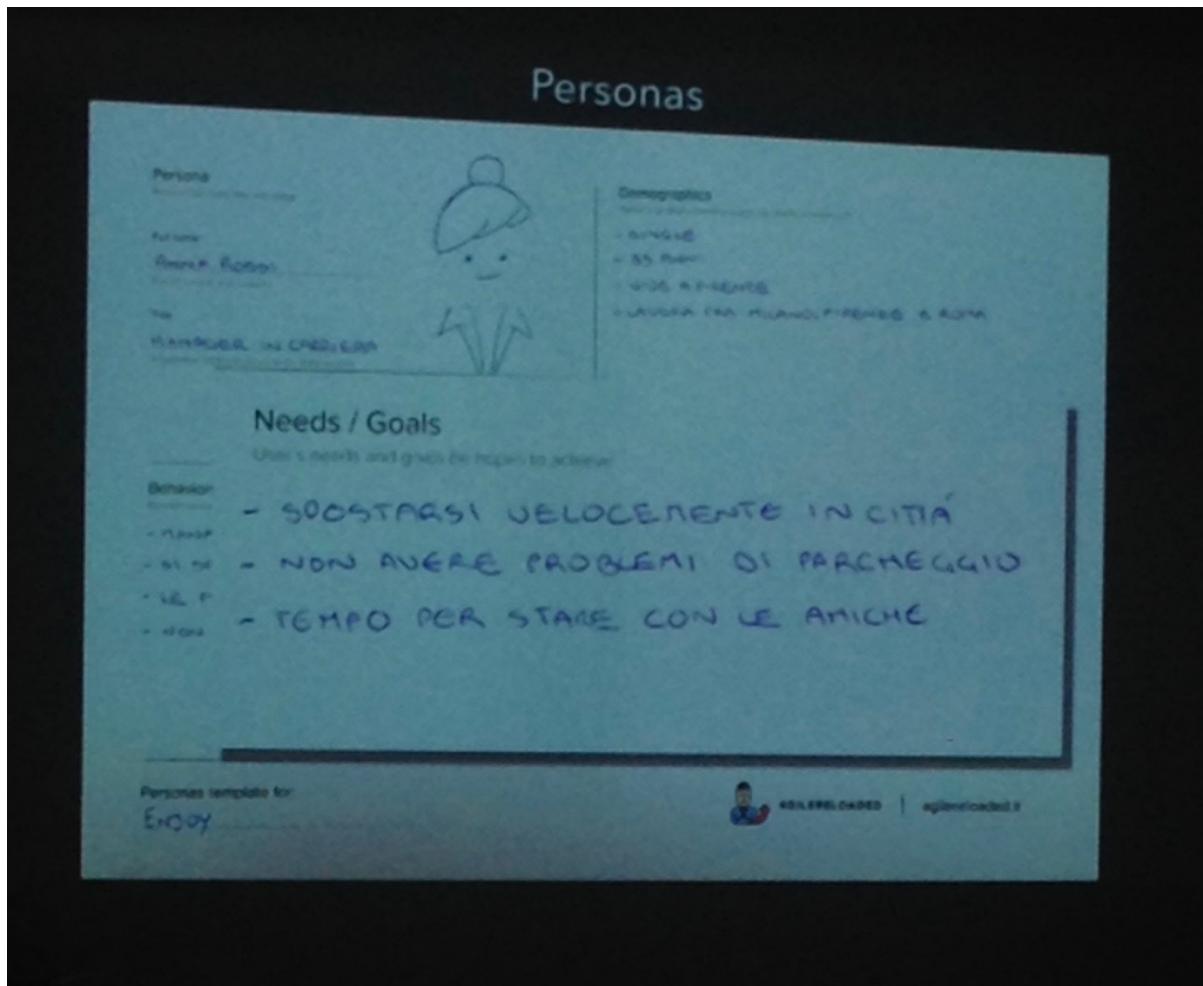
PORTANO BENEFICI MA ANCHE QUALCHE PROBLEMA

BENEFICI E PROBLEMI

- ▶ pro: contesti ben definiti
contro: i sistemi distribuiti sono più difficili da programmare
- ▶ pro: posso fare deploy in modo indipendente
contro: possibili inconsistenze
- ▶ pro: posso essere poliglotta
contro: è necessario un team di esperienza per orientarsi

Come organizziamo lo sviluppo di una nuova funzionalità?

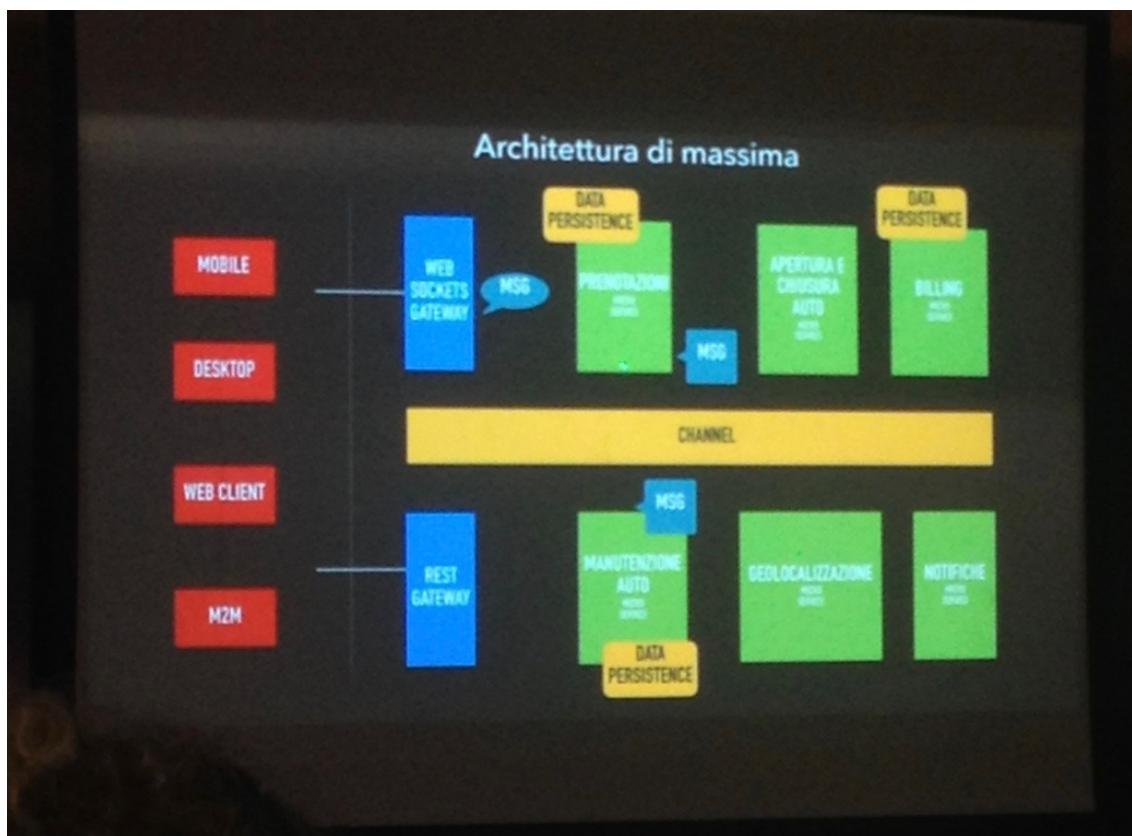
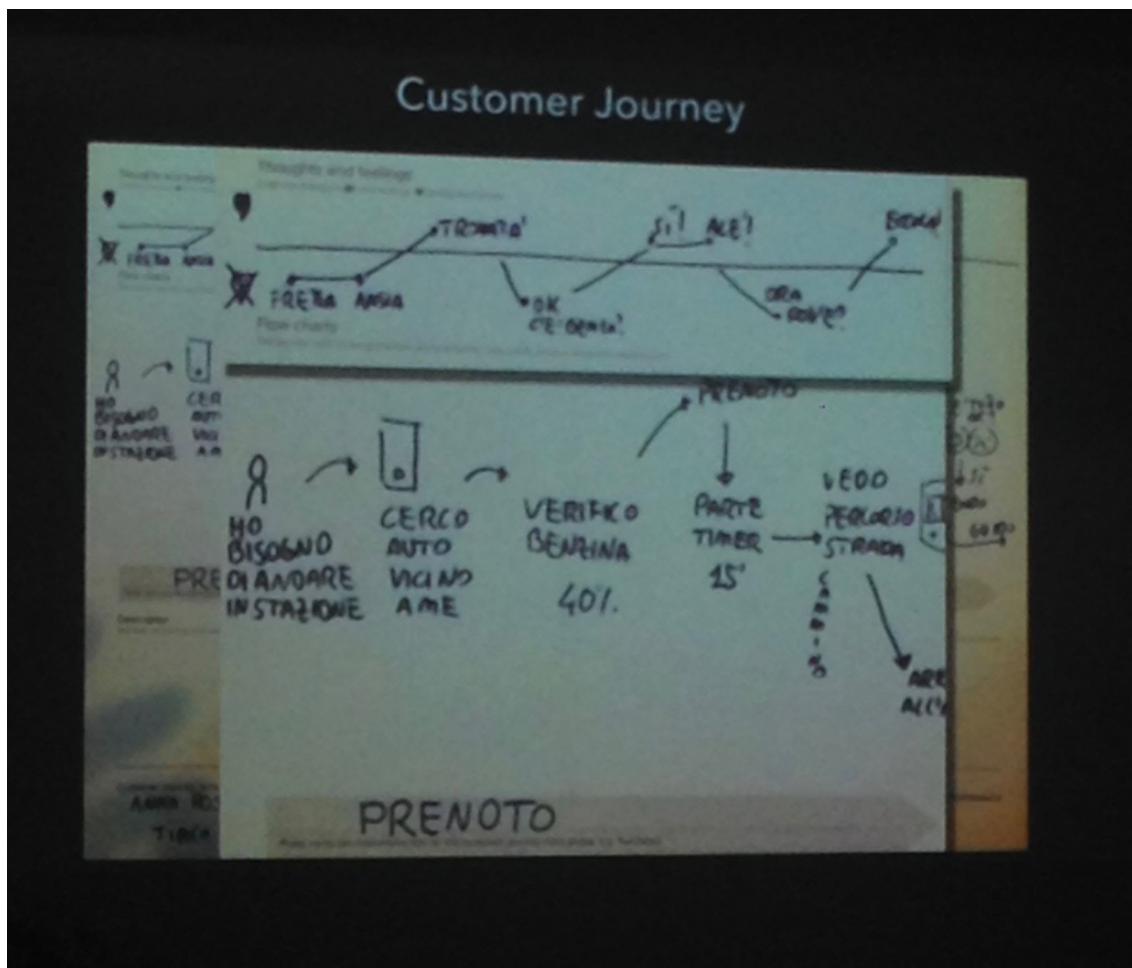
Come prima cosa, definiamo gli obiettivi, esigenze, necessità:



Definiamo user story.

Speaker propone di combinare il flusso di azioni che esegue un utente con lo stato emotivo, in quale si trova in un certo momento.

In alto abbiamo gli stati emotivi, in basso, gli stati comportamentali.



Nell'immagine precedente vediamo un architettura di massima.

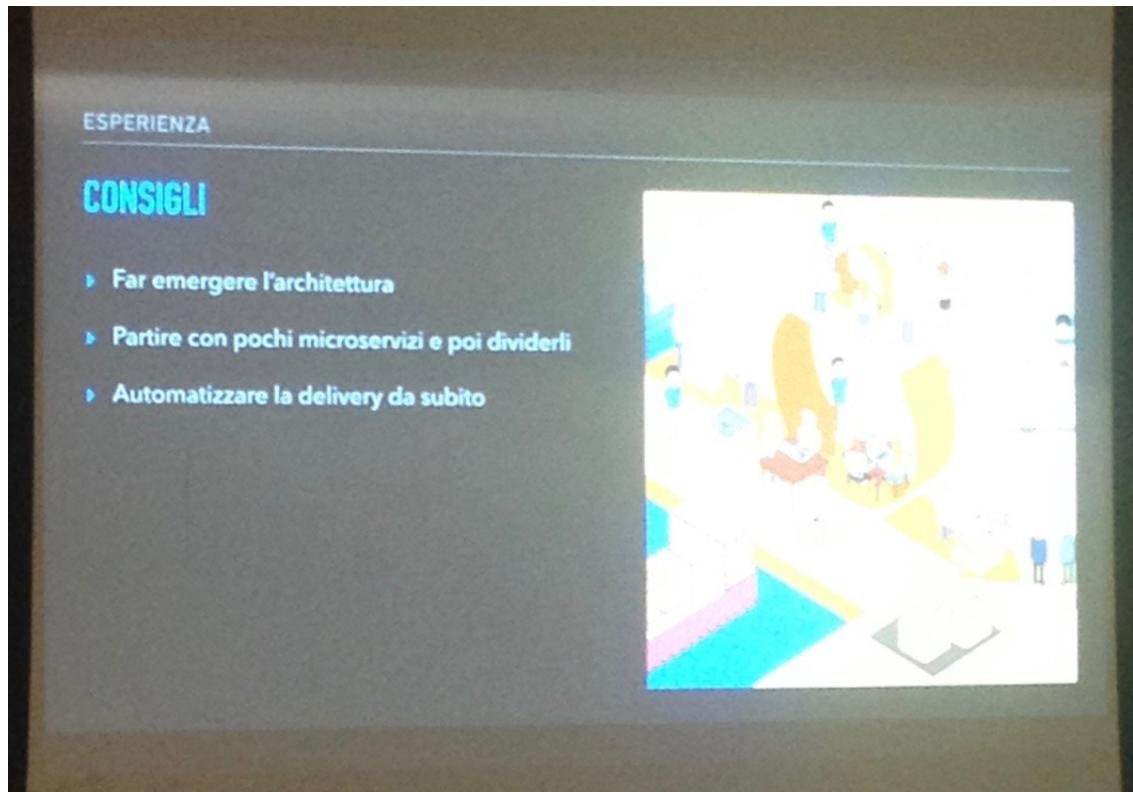
In rosso sono tutti i client utilizzando quali possiamo accedere al nostro sistema.

In blu abbiamo l'interfaccia tra i client e sistema in formato di servizi REST o Web Socket.

In verde è una singola funzionalità.

In giallo, dove c'è scritto Channel, abbiamo il canale di comunicazione tra diverse funzionalità. Può essere un service bus (ESB), una coda di messaggi, etc.

Riepilogo:



Account twitter degli speaker:

@MaxMex59

@EmilianoSoldi

@felicepescatore

@andreaprotaglio

@ilariamauric

@giulioroggero

Libri interessanti:

- Accelerate: Building Strategic Agility for a Faster-Moving World https://www.amazon.it/gp/product/1625271743/ref=oh_aui_detailpage_o00_s00?ie=UTF8&psc=1
- Holacracy: The New Management System for a Rapidly Changing World https://www.amazon.it/gp/product/1627794875/ref=oh_aui_detailpage_o02_s00?ie=UTF8&psc=1