
Metodologie: design per il Web

Summary

- Principi e metodologie per il design delle applicazioni
- Interazione e interfacce
- Metodologie e strumenti per il design delle interfacce

Principi e metodologie per il design delle applicazioni

Metodologie

- Esistono diverse metodologie che possono essere adottate nel design e sviluppo di software
- Approfondimento e dettagli nei corsi di Project Management e di Paradigmi di programmazione e sviluppo
- Noi introduciamo:
 - Il principio del ***debito tecnico***
 - Metodologie ***Waterfall*** e ***AGILE***
 - Coinvolgimento degli utenti: ***User Centered Design***, ***Design Partecipativo*** e ***Open Innovation***

Debito Tecnico

- Metafora inventata da Ward Cunningham
- Descrive le possibili complicazioni che subentrano in un progetto (tipicamente di sviluppo software), qualora non vengano adottate adeguate azioni volte a mantenerne bassa la complessità

Debito Tecnico

- Analogamente a quanto avviene nel mondo finanziario, in cui per sanare un debito occorre pagarne anche gli interessi, *lo sforzo per recuperare un progetto sviluppato senza una corretta metodologia può aumentare anche considerevolmente nel tempo*, se non si interviene tempestivamente o se *non si affronta la fase di design in modo opportuno*

Debito Tecnico

- “*Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite... The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt. Entire engineering organizations can be brought to a stand- still under the debt load of an unconsolidated implementation, object-oriented or otherwise”*

Ward Cunningham, 1992

https://en.wikipedia.org/wiki/Ward_Cunningham

Debito Tecnico

- *“As an evolving program is continually changed, its complexity, reflecting deteriorating structure, increases unless work is done to maintain or reduce it”*

Meir Manny Lehman, 1980

[https://en.wikipedia.org/wiki/Manny_Lehman_\(computer_scientist\)](https://en.wikipedia.org/wiki/Manny_Lehman_(computer_scientist))

Cause del debito tecnico

- *Esigenze di mercato*: urgenza di avere un prodotto da vendere prima possibile, quindi rilasciato prima che le dovute modifiche siano complete
- *Mancanza di conoscenza del processo*: chi prende le decisioni è ignaro delle implicazioni sottostanti (o decide di ignorarle)
- *Mancanza di disaccoppiamento*: lo sviluppo delle componenti del programma avviene ignorando il paradigma di programmazione modulare o comunque senza l'intento di mantenere basso il legame di dipendenza tra i sottosistemi

Cause del debito tecnico

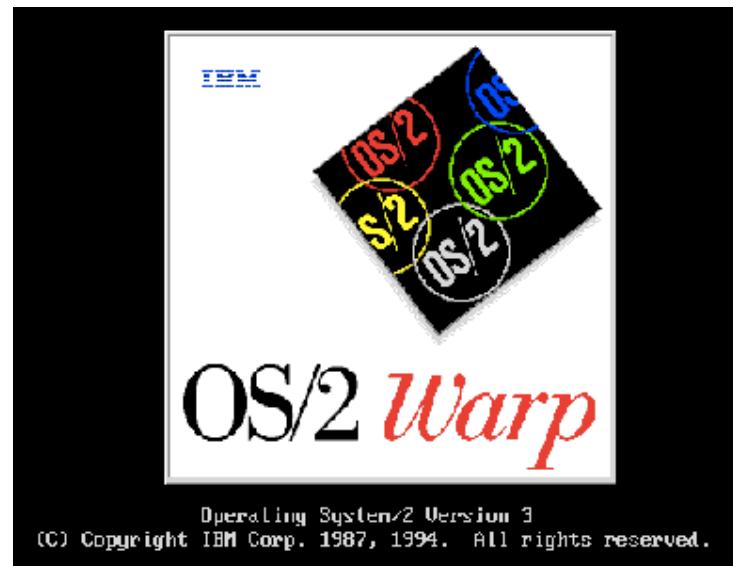
- *Assenza di procedure di test:* incoraggia correzioni di bug "al volo" che non contemplano possibili effetti secondari
- *Assenza di documentazione:* il codice viene sviluppato "a braccio", senza documentazione/specificazione dei requisiti
 - Il lavoro per produrre la documentazione a posteriori, e la necessaria verifica di corrispondenza con quanto già codificato, rappresenta un debito che occorre prima o poi saldare

Cause del debito tecnico

- *Mancanza di collaborazione*: causa degrado dell'efficienza del processo di sviluppo
 - Un'altra motivazione potrebbe essere dovuta ad un affiancamento di sviluppatori inesperti/neoassunti non adeguato (insufficiente o assente) da parte delle figure professionali con maggior esperienza
- *Mancanza di conoscenza*: nel caso in cui lo sviluppatore non abbia competenze adeguate per scrivere codice di qualità

Date di scadenza

- Il refactoring del codice non basta:
 - Il software scade (sistemi operativi, applicazioni, pensate al millennium bug!, ecc)
 - L'hardware scade (es: lettori CD e porte VGA nei portatili)
 - Le infrastrutture scadono



Metodologie

- 1  Agile – collaborating to iteratively deliver whatever works
- 2  Scrum – enabling a small, cross-functional, self-managing team to deliver fast
- 3  Kanban – improving speed and quality of delivery by increasing visibility of work in progress and limiting multi-tasking
- 4  Scrumban – limiting work in progress like Kanban, with a daily stand up like Scrum
- 5  Lean – streamlining and eliminating waste to deliver more with less
- 6  eXtreme Programming (XP) – doing development robustly to ensure quality
- 7  Waterfall – planning projects fully, then executing through phases
- 8  PRINCE2 – controlled project management that leaves nothing to chance
- 9  PMI's PMBOK – applying universal standards to Waterfall project management

Metodologie

- 1  Agile – collaborating to iteratively deliver whatever works
- 2  Scrum – enabling a small, cross-functional, self-managing team to deliver fast
- 3  Kanban – improving speed and quality of delivery by increasing visibility of work in progress and limiting multi-tasking
- 4  Scrumban – limiting work in progress like Kanban, with a daily stand up like Scrum
- 5  Lean – streamlining and eliminating waste to deliver more with less
- 6  eXtreme Programming (XP) – doing development robustly to ensure quality
- 7  Waterfall – planning projects fully, then executing through phases
- 8  PRINCE2 – controlled project management that leaves nothing to chance
- 9  PMI's PMBOK – applying universal standards to Waterfall project management

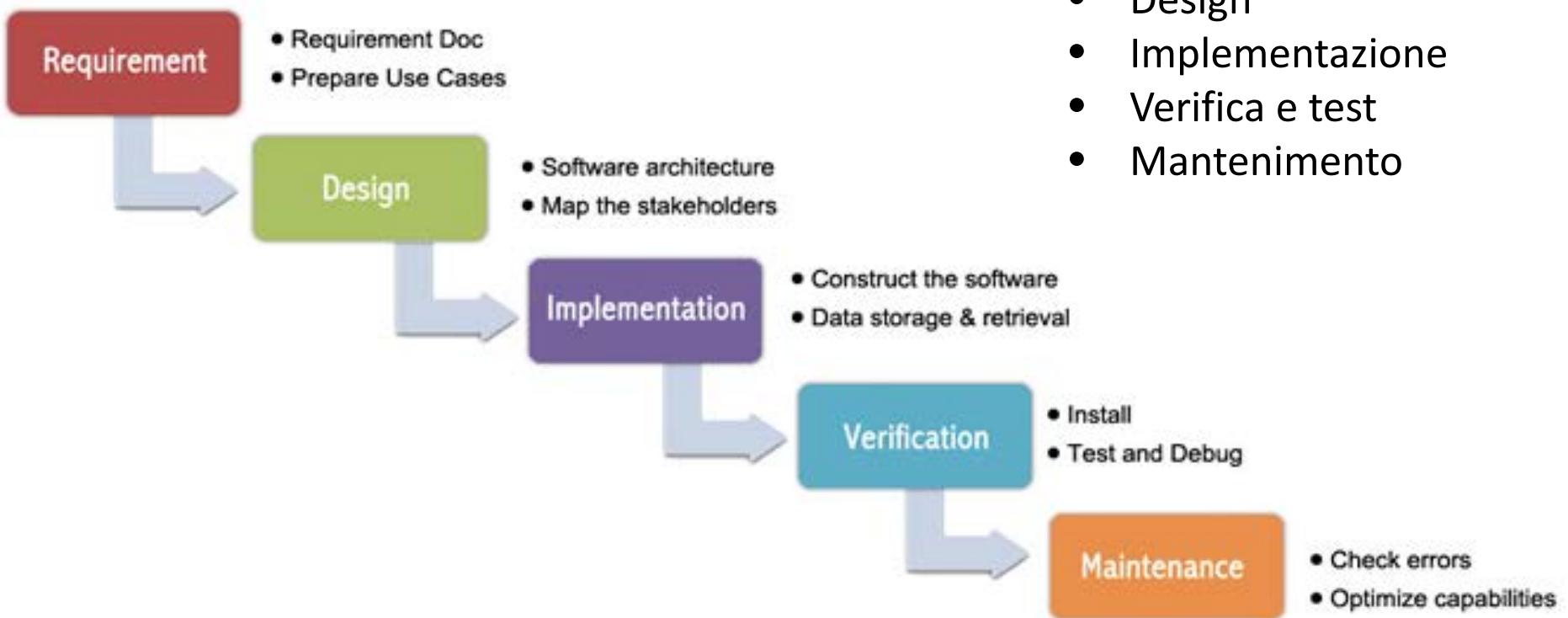
Waterfall

- *Planning projects fully, then executing through phases*
- Primo modello di ciclo di vita del software, risale agli anni '60, quando per la prima volta lo sviluppo del software viene inteso come un processo industriale
- Questa metodologia prevede una sequenza di sottoattività (ognuna con documentazioni e controllo), anziché come attività "artigianale" (approccio *code and fix*)
- Associata alla programmazione procedurale e strutturata
- Ogni fase (o sottoattività) produce un output, che rappresenta l'input della fase successiva
- Non è prevista sovrapposizione tra le fasi

Waterfall

- Consiste in un'unica pianificazione, seguita dalle fasi sequenziali che vengono eseguite una sola volta
- Una volta approvata la pianificazione, è possibile adattarla solo se strettamente necessario
- Una volta arrivati alla fase di verifica e test è molto difficile tornare indietro e modificare qualcosa che non era stato opportunamente progettato all'inizio
- Potenzialmente rischioso, dato che non prevede coinvolgimento del cliente/committente e degli utenti (tranne nell'analisi dei requisiti e al termine del processo)
- Potrebbe essere efficace e predicibile se i requisiti sono ben documentati, fissati e chiari, se la tecnologia è matura e conosciuta dagli sviluppatori, se il progetto è a breve termine, ecc

Waterfall



AGILE

- Insieme di metodi di sviluppo del software fondati su insieme di principi comuni derivati dai principi del *Manifesto for Agile Software Development* ("Manifesto Agile", <http://agilemanifesto.org/>)
- I *metodi agili* si contrappongono al ***modello a cascata*** e altri processi software tradizionali, proponendo un approccio meno strutturato e focalizzato sull'obiettivo di consegnare software funzionante e di qualità in tempi brevi e frequentemente

AGILE

- I *metodi agili* tentano di ridurre il rischio di fallimento sviluppando il software in finestre di tempo limitate (iterazioni)
- Ogni iterazione è un progetto a sé stante e deve contenere tutto ciò che è necessario per rilasciare un piccolo incremento nelle funzionalità del software
- Il risultato di ogni singola iterazione non può essere considerato completo, ma il susseguirsi delle iterazioni si avvicina sempre di più a quanto richiesto dal committente
- Al termine di ogni iterazione si rivalutano le priorità del progetto
- La comunicazione gioca un ruolo fondamentale: deve avvenire in tempo reale, possibilmente di persona
- Spesso nel team di progetto sono inclusi i clienti/committenti e idealmente anche target user

AGILE: i principi

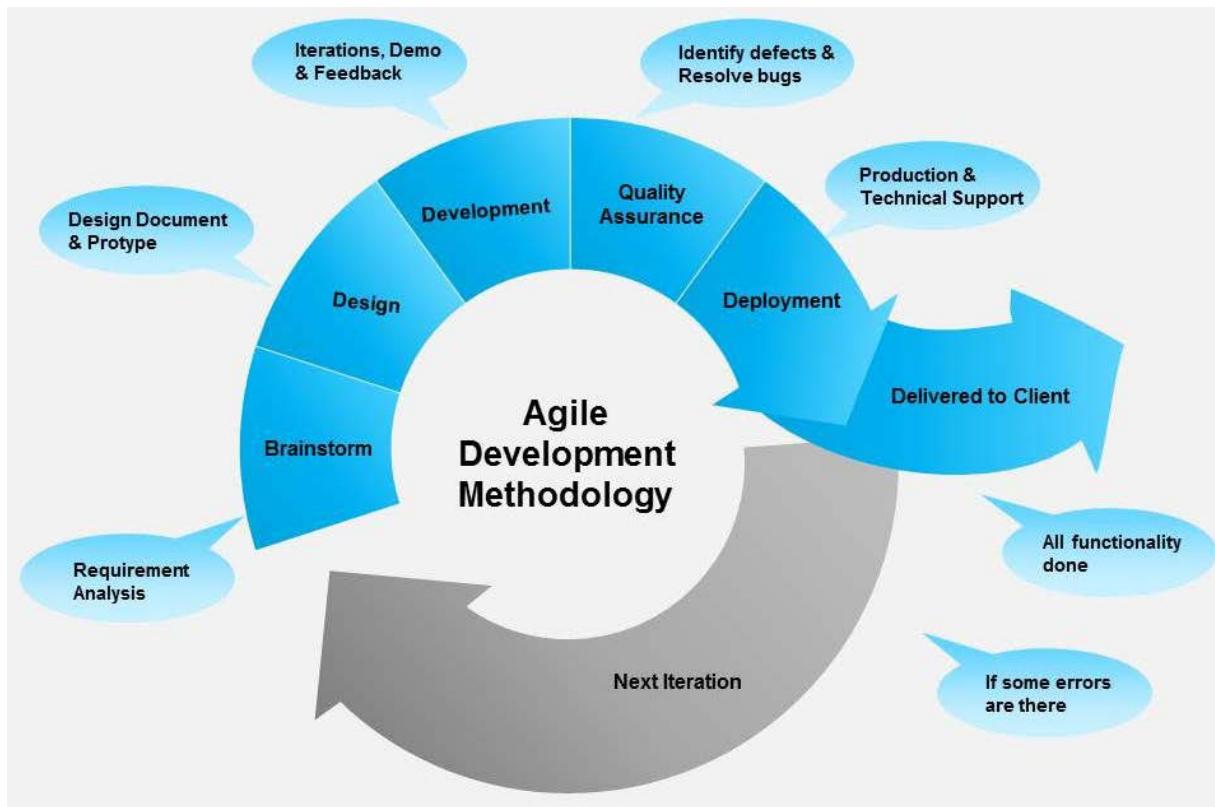
1. Persone e interazioni sono più importanti di processi e strumenti (importanza di relazioni e comunicazione tra gli attori di un progetto software)
2. Maggiore importanza del software funzionante rispetto alla documentazione (rilascio frequente di nuove versioni, riducendo la documentazione al minimo indispensabile)
3. Importanza della collaborazione con i clienti/target user
4. Flessibilità: il team di sviluppo deve essere flessibile e rispondere ai cambiamenti, modificando priorità di lavoro (mantenendo l'obiettivo finale)

AGILE: pratiche

- Le pratiche applicabili all'interno di una metodologia agile sono varie. Dipendono essenzialmente dalle necessità dell'azienda e del committente e dall'approccio scelto
- Alcune pratiche:
 - Involgimento attivo del cliente
 - Comunicazione stretta
 - Consegni frequenti
 - Iterative development
 - Modellizzazione
 - Versioning

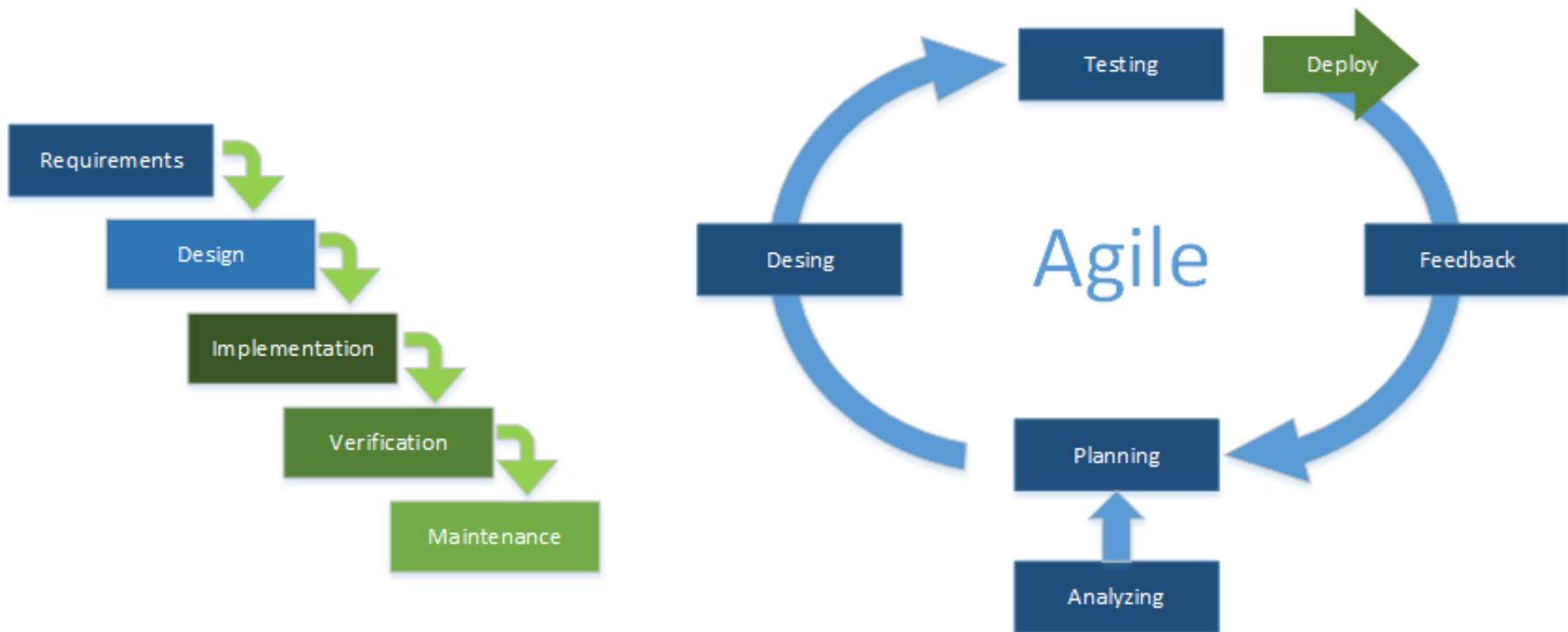
AGILE: fasi

Le fasi sono riconducibili alle sottoattività del modello Waterfall, ma sono ripetute iterativamente, fino al completamento del software



Waterfall vs AGILE

Waterfall vs. Agile



MVP

- “*The Minimum Viable Product is that version of a new product which allows a team to collect the maximum amount of validated learning about customers with the least effort*”



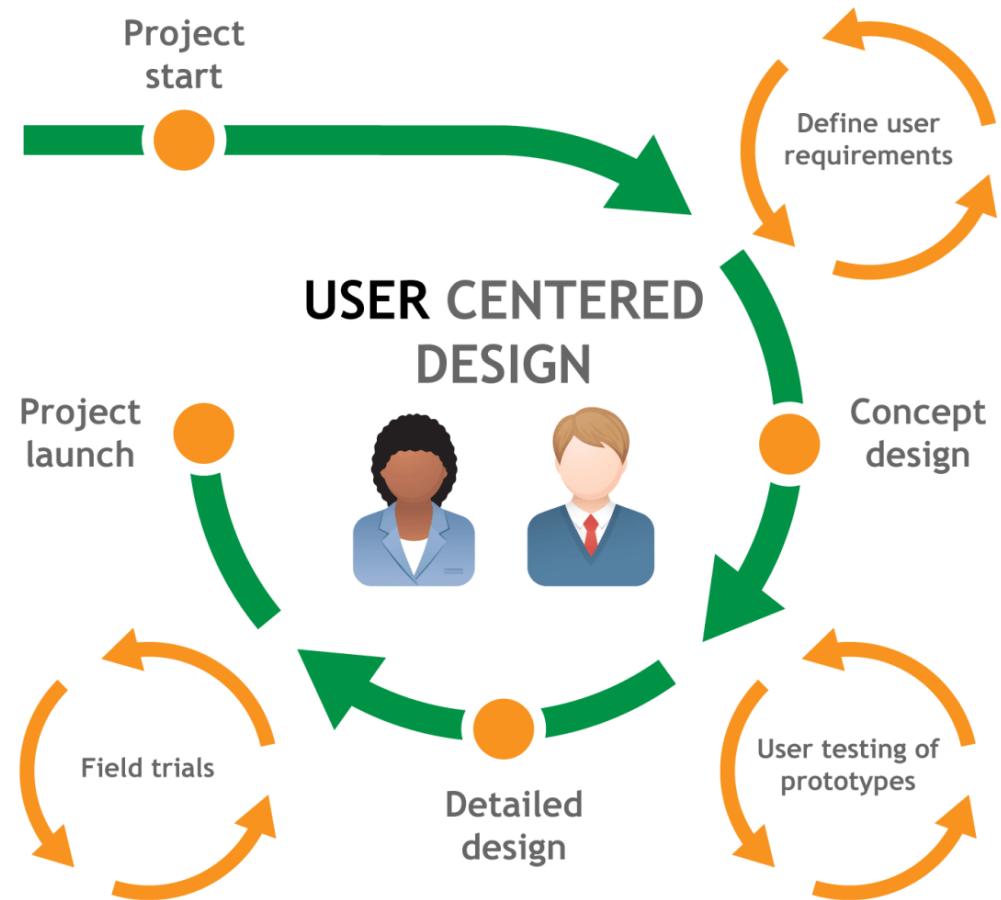
Eric Ries

Q&A

- Ci sono domande?

User Centered Design (UCD)

- Approccio nato a partire dalla seconda metà degli anni '80
- Consiste nel design e sviluppo di applicazioni (o prodotti) da parte di un team di sviluppo che si focalizza sui bisogni degli utenti, in modo iterativo



User Centered Design

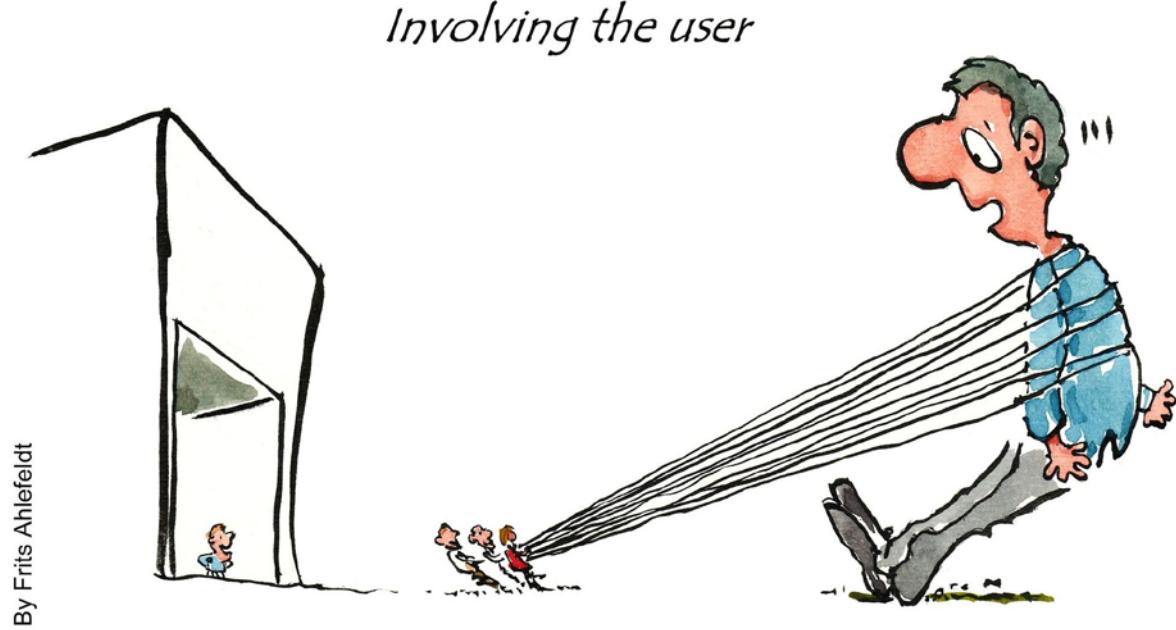
- Questo approccio potrebbe essere adottato senza l'effettivo coinvolgimento di utenti reali (potrebbero essere coinvolti solo nella fase di testing)
- Gli utenti potrebbero essere “virtualizzati” durante l'intero ciclo di progettazione/sviluppo/valutazione del software, simulandoli, creando dei modelli di utenti (es. ***personas***)
- Il team di sviluppo si focalizza sulla applicazione da implementare, con l'obiettivo di rispondere ai bisogni e alle richieste degli utenti, sulla base delle loro caratteristiche

User Centered Design

- Cosa può caratterizzare l'utente:
 - Competenza
 - Abilità fisica
 - Livello di attenzione
 - Motivazioni
- Necessario:
 - Comprensione dei requisiti e caratteristiche degli utenti
 - Fare focus sui task della applicazione e su come questi incontrano l'utente

Approcci partecipativi

- Alcuni approcci sono più «inclusivi»: non solo design e progettazione si concentrano su caratteristiche e bisogni dell’utente, ma vanno oltre
- Utenti target sono coinvolti (fisicamente) nelle varie fasi del processo



Approcci partecipativi

- Sviluppatori e target user sono in diretta e continua comunicazione
- In alcuni casi, alcuni target user fanno direttamente parte del team di sviluppo, con un coinvolgimento attivo e diretto
- Permette di considerare il punto di vista dell'utente sin dalle prime attività di definizione dei requisiti

Design partecipativo (Participatory Design)

- Approccio usato non solo nello sviluppo software
- Spesso riferito anche come ***Co-design***
- Consiste nel coinvolgimento di target user, stakeholder, nel team di sviluppo, facendoli partecipare a tutte le fasi del processo, non solo al testing, ma in modo tale che possano contribuire attivamente anche alle fasi di design e di sviluppo (per lo meno di contenuti e funzionalità)
- Gli utenti coinvolti sono definiti ***co-designer***: possono proporre e generare design issue, funzionalità, servizi dell'applicazione

Design partecipativo

- Progettisti, sviluppatori e target user collaborano durante la fase di design ideando e definendo possibili contesti di utilizzo usando diverse tecniche (ne vedremo alcune nel seguito di questa lezione)
- Nella fase di design, il ruolo di progettisti e sviluppatori sfoca, rispetto a quello dell'utente, che diventa una componente predominante nel processo



UCD e PD: intersezioni

- Le differenze e le similarità tra User Centered Design e Participatory Design sono ancora molto dibattute
- In alcuni contesti, PD è considerato come un “approfondimento” di UCD
- Una interessante interpretazione si basa sul concetto che PD è user-centered, quando si focalizza sugli interessi e sulla soddisfazione degli utenti finali
- Intersezioni e confine di questi due approcci, così come la loro effettiva applicazione sono oggetto di interesse e di utilizzo in moltissimi contesti (non solo sviluppo di software)

Co-creation e Open Innovation

- ***Co-creation***: coinvolge i target user nella fasi di definizione dei requisiti di una applicazione (o di un prodotto non software). Potenziali utenti guidano la definizione non solo dei requisisti, ma anche dei goal
- ***Open innovation***: attiva collaborazione tra diverse organizzazioni ed enti, con condivisione della proprietà intellettuale
- Entrambi basati su un forte coinvolgimento degli utenti, non solo nelle fasi di design, ma anche nelle fasi precedenti (definizione dei goal)

Q&A

- Ci sono domande?

Interazione e interfacce

Interazione e Interfaccia

- Che cos'è una ***INTERFACCIA***?
- Cosa vuol dire ***INTERAZIONE***?

Di porte e maniglie



Di porte e maniglie



Di porte e maniglie



Di porte e maniglie



Di porte e maniglie



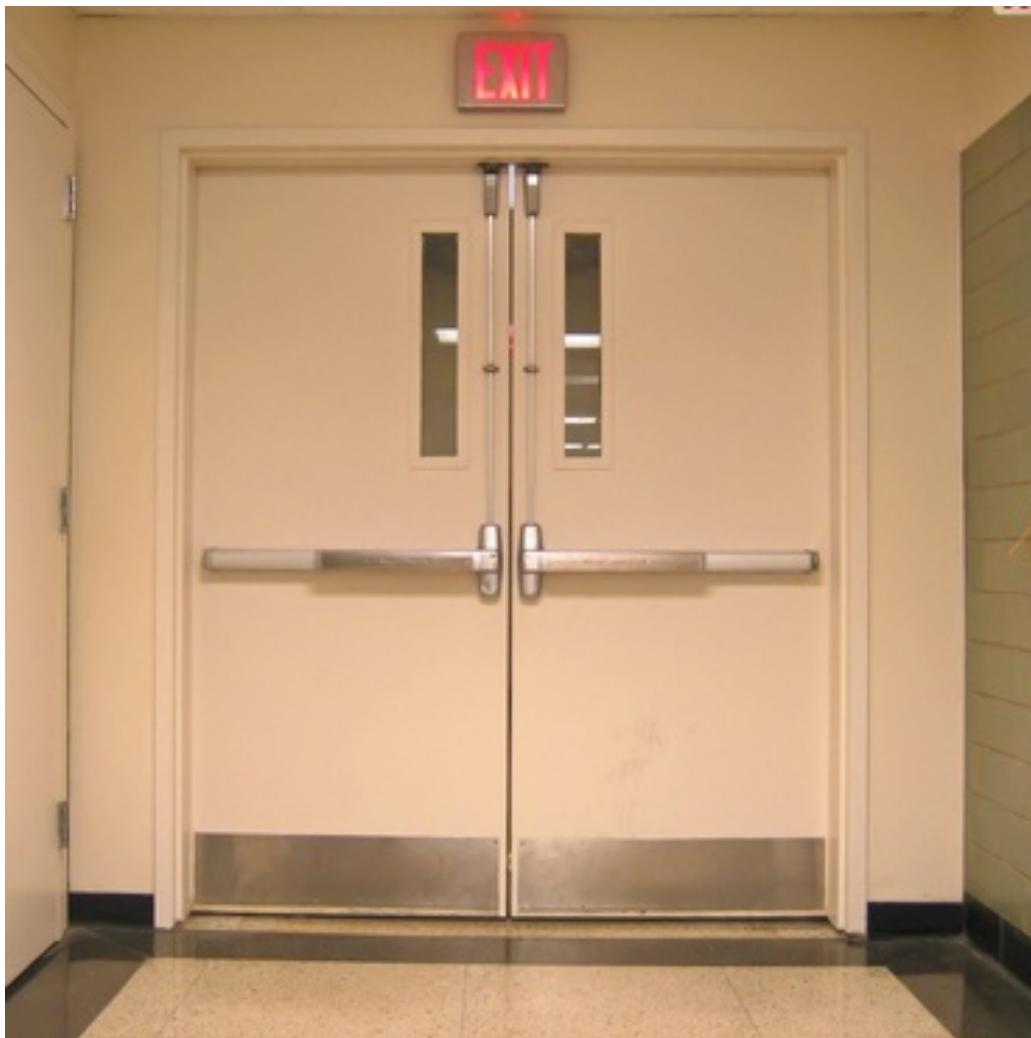
Di porte e maniglie



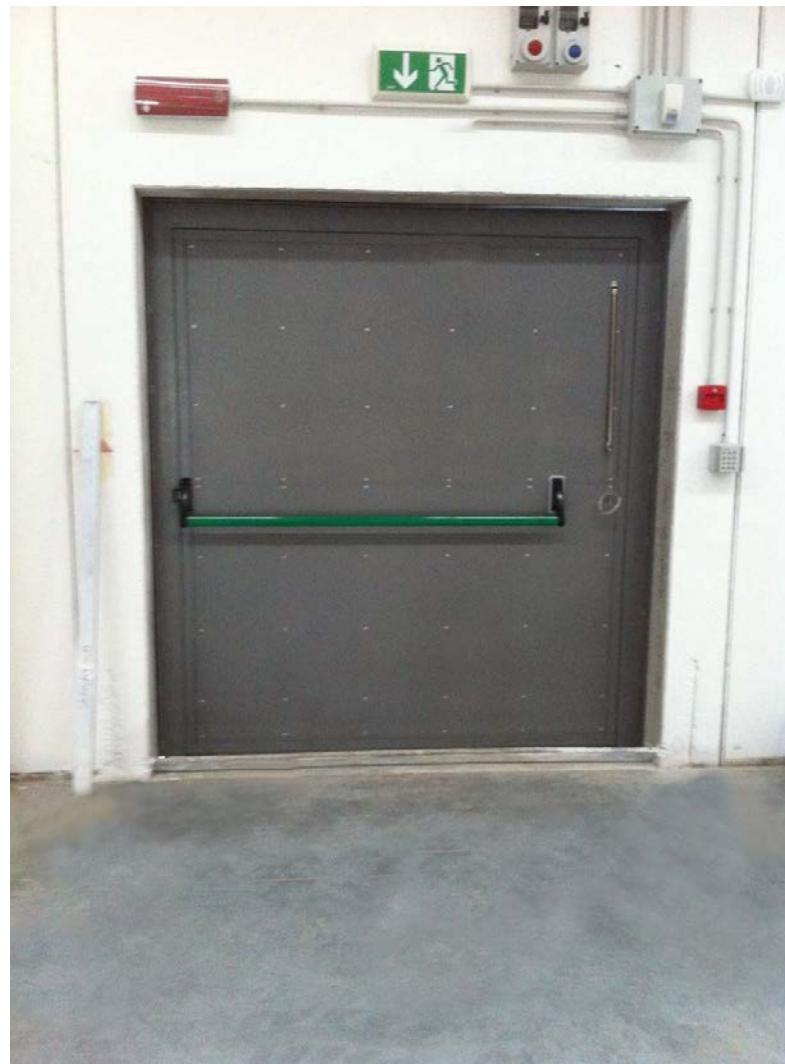
Di porte e maniglie



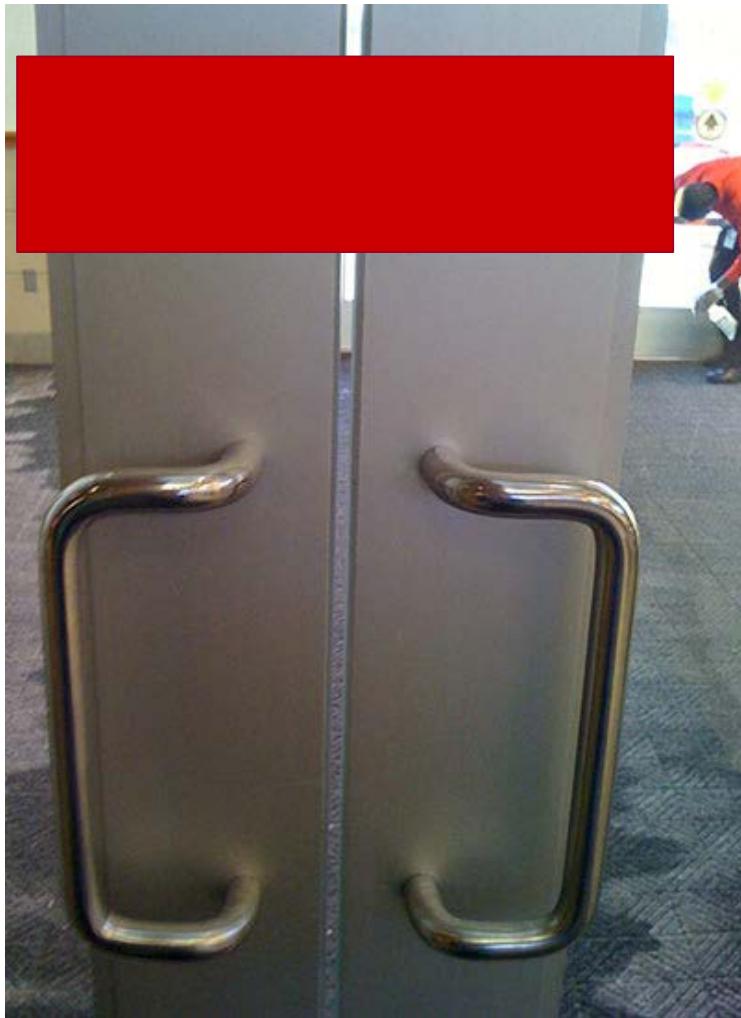
Di porte e maniglie



Di porte e maniglie



Di porte e maniglie



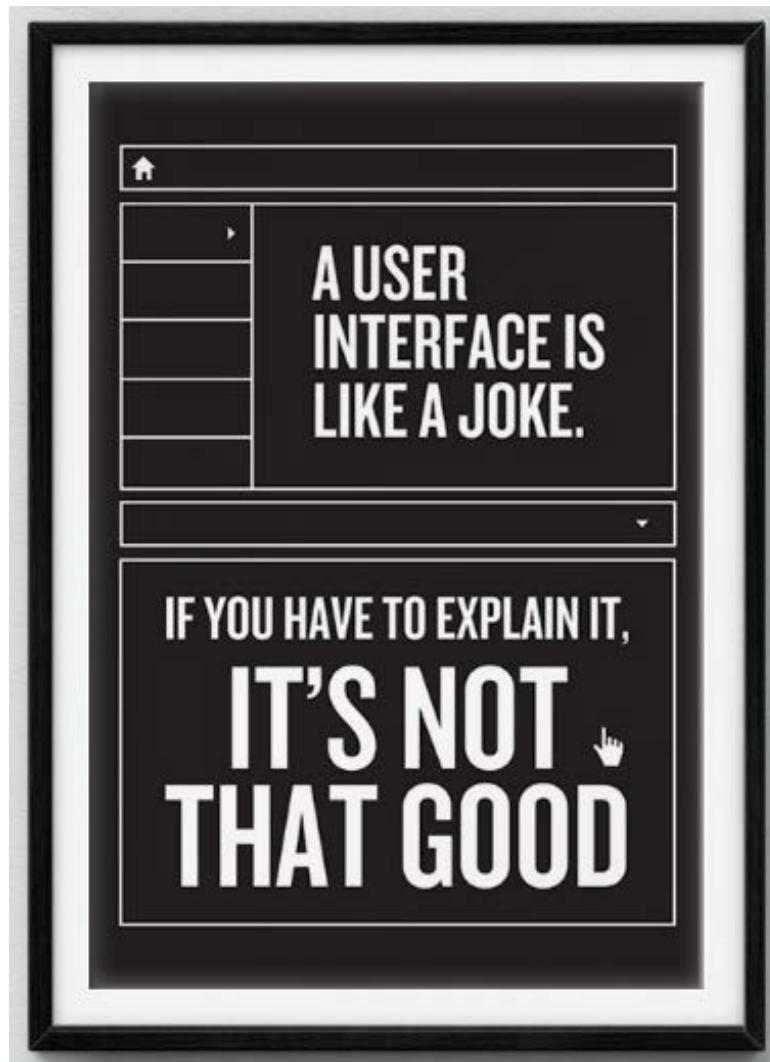
Di porte e maniglie



Di porte e maniglie



Ma ...



Norman's Door

It's not you. Bad doors are everywhere:

<https://www.youtube.com/watch?v=yY96hTb8Wgl>

Donald Norman

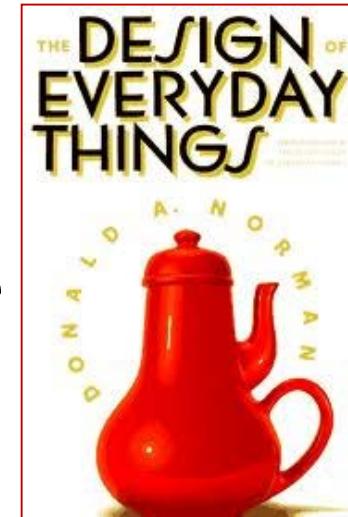
- Laureato in ingegneria elettronica al MIT. Master e dottorato di ricerca all'Università della Pennsylvania
- Professore di psicologia e scienze cognitive all'Università della California
- Professore emerito al MIT
- Vicepresidente del gruppo di ricerca sulle tecnologie avanzate di Apple Computer
- Dirigente alla Hewlett Packard e alla U Next, per la formazione a distanza
- Insegna psicologia, scienze cognitive e informatica alla Northwestern University
- Ha fondato il Nielsen Norman Group nel 1998 con Jakob Nielsen



Di jordanfischer -
<http://flickr.com/photos/jordanfischer/61429449/>,
CC BY 2.0,
<https://commons.wikimedia.org/w/index.php?curid=2618273>

La Caffettiera del Masochista

- A chi non à mai capitato di spingere una porta invece di tirarla o di rinunciare a lavarsi le mani perché non riesce ad azionare il rubinetto?
- In questi casi la sensazione di incapacità personale è molto forte: la colpa non è dell'utente, ma di chi ha progettato questi oggetti d'uso comune senza considerare le normali attività mentali la cui conoscenza è essenziale per la progettazione di un ambiente ben organizzato e rispondente alle esigenze della mente
- Tutto ciò vale per qualunque oggetto e qualunque interazione e interfaccia, a maggior ragione è vero per interazione e interfaccia tra utente e device digitali e tecnologici



La Caffettiera del
Masochista

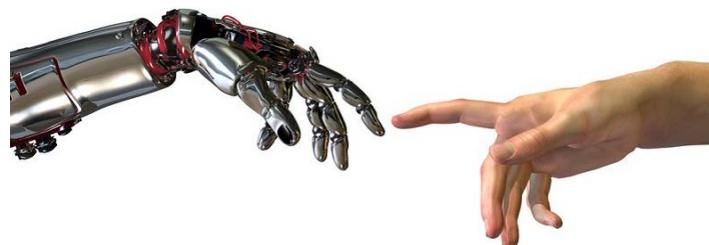
By Source, Fair use,
<https://en.wikipedia.org/w/index.php?curid=34029875>

Q&A

- Ci sono domande?

HCI

- HCI → Human Computer Interaction (interazione uomo- computer o interazione uomo-macchina)
- È la disciplina che studia l'interazione tra gli utenti e i computer per definire la progettazione e lo sviluppo di sistemi interattivi che siano usabili, affidabili e che supportino e facilitino le attività umane
- È una disciplina intrinsecamente interdisciplinare per poter cogliere i vari aspetti che possono essere rilevanti nell'interazione tra i due sistemi
 - Uomo
 - Macchina



Perché HCI?

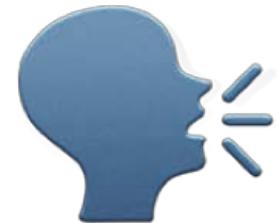
- La sempre maggiore diffusione e il sempre maggior uso di applicazioni informatiche in ogni aspetto della vita ed attività umana richiede una progettazione che sappia tenere conto:

- dei vari possibili contesti d'uso
- degli obiettivi degli utenti
- delle nuove tecnologie di interazione
- delle diverse competenze degli utenti
- dello scopo dell'applicazione stessa



Evoluzione di interfacce e interazione

```
ULTIM89 COMPUTER BY MATT SARNOFF  
DEBUG ROM DD 3 - 28 JUN 2018  
SIZEK AVAILABLE RAM  
  
CRD: NO NAME  
1280x960 BYTES  
OFFSET: F33 R27 D311  
  
MONITOR READY  
> kg hello.ek9  
  
Hello, San Francisco!  
It sure feels good to be out of that  
tiny memory card.  
(press any key to quit)  
Bye!  
  
BREAK AT 0117.  
MONITOR READY.  
>
```



Keyboard

Mouse

Touch

Voice

Interazione e Interfaccia

- **Interfaccia utente:** componente del software che fa sì che l'utente riesca a comunicare ed interagire con le altre parti dell'applicazione (e quindi con il device). Es: la maniglia della porta (che serve all'utente per aprire la porta)
- **Interazione:** consiste nella relazione tra l'utente e il sistema/device. Si compone delle azioni che l'utente può effettuare sul sistema e come il sistema reagisce a tali azioni (fornendo feedback). Es: come l'utente può/deve usare la maniglia della porta (per aprire la porta)

HCI: il lato umano

- L'utente è (e deve essere) il protagonista principale
- Come si può progettare e sviluppare una interfaccia utente e un sistema di interazione che siano efficaci?
- Come si può migliorare la qualità del sistema di interazione?

COINVOLGENDO GLI UTENTI

HCI: il ruolo degli utenti

- Il ruolo degli utenti nei processi di design e di creazione di applicazioni e di artefatti software è alla base di molte metodologie HCI
- Nell'evoluzione di queste metodologie, il ruolo degli utenti è via via diventato sempre più attivo:
 - L'utente è diventato il protagonista in molti degli step degli approcci usati più comunemente per lo sviluppo di applicazioni
- Vediamo alcune metodologie HCI e vediamo il coinvolgimento degli utenti che prevedono

Q&A

- Ci sono domande?

Metodologie e strumenti per il design delle interfacce

Metodologie e strumenti

- Target users analysis: Personas, scenari d'uso
- Storyboard, sketch, mockup
- Focus group
- Experience prototyping

Analisi dei target user

- Di tutti gli utenti?
 - No, va individuato il target di riferimento
 - Fascia di età, motivazioni, interessi, scopo, ...
- Individuato il target è utile creare delle ***Personas***
 - Una ***persona*** (termine in inglese) rappresenta una tipologia di possibile utente del sistema
 - Occorre raccontare una storia e si valuta la sua credibilità
 - ***Personas***
 - Contesto (context)
 - Trama (use cases o scenario)

Personas

- Si inventano finti utenti che incarnano le caratteristiche che vogliamo sostenere ed aiutare con il progetto.
- I personaggi sono dunque archetipi astratti di
 - Intenzioni
 - Scopi
 - Abitudini
- Informazioni da fornire per ogni persona
 - Scopi, motivazioni
 - Atteggiamenti, comportamento
 - Storia (nome, età, lavoro, educazione, contesto familiare, aspetto)

Cattive Personas

- Una persona deve rappresentare una classe di utenti
- Grandezza classe di utenti rappresentati
 - Non troppo grande: L'«utente elastico» non va bene!
 - Non troppo piccola:
 - L'utente anarchico non va bene!
 - Il caso sociale non va bene!
 - L'utente auto-referenziale (la bella copia del progettista) non va bene!

Buone Personas

- **No persona media**
 - Nessuno scrive storie su persone medie, a cui non succede nulla di interessante, senza un passato rilevante, senza una personalità concretamente percepibile e descrivibile
- **Sì persona interessante**
 - Una storia ha come protagonista una persona interessante, tipica ma non media, anzi un po' di confine, le cui peculiarità la facciano risaltare sugli altri e la rendano una persona vivace, memorabile, con una personalità propria

Specificità

- Occorre definire in dettaglio l'utente e la sua storia prima di descrivere in dettaglio i task che deve compiere nel sistema
- Esempio
 - NO: L'utente è uno studente che usa UBUNTU
 - SI: L'utente è Marco, uno studente curioso e socievole che frequenta il Liceo Righi. E' sempre stato appassionato di computer e tecnologia e si sta appassionando di sistemi Linux, grazie alla sua prof di informatica che è contributore attivo in Debian e gli ha trasmesso la passione per i sistemi open source e nell'idea che sta alla base di questa filosofia

Il vero Marco non va bene?



- Come **Personas**, no!
 - L'utente reale ha peculiarità e idiosincrasie che un utente virtuale non ha
- Nel design partecipativo includiamo (fisicamente) l'utente in ogni fase del processo, ma non nella metodologia **Personas**
 - Non un solo utente, ma tanti utenti diversi che hanno una determinata caratteristica in comune
 - Anziani; bambini; dipendenti di un'azienda; giocatori online; ecc

Scenario d'uso

- Storia in cui si racconta nel dettaglio come un certo utente porta a termine un goal personale realizzando uno o più dei task evidenziati sul sistema.
- Descrive l'interazione della persona con il sistema
- Descrive i task, non i comandi dell'interfaccia
- E' molto specifico
- Descrive un'azione completa
- Descrive degli utenti (intesi come persona)

Storyboard

- Tecnica per illustrare in immagini la struttura dell'esecuzione di un task mostrando lo stato dello schermo durante le varie fasi dell'azione
- Mostra il succedersi delle pagine e l'attivazione dei widget interattivi (pulsanti, popup, ecc.) in maniera grafica, semplice e comprensibile

Storyboard



Disegni e Sketch

- Gli Storyboard possono essere realizzati attraverso disegni e sketch fatti a mano, oppure anche usando applicazioni specifiche per realizzare dei mockup
- Esempio:
 - alcuni sketch utilizzati in fase di design di una applicazione per device mobile per fare wayfinding indoor.
 - Lo scenario è basato su studenti e turisti come target user, che vogliono visitare e accedere alla vecchia sede in Via Sacchi del nostro Corso di laurea

Storyboard con disegni e Sketch: un esempio

PER LOCALIZZARSI SUA MAPPA?

IMMAGINIAMO DI USARE L'APP (ES. VIA SACCHI)

A anche una vista come queste puo' essere molto utile.
Appena entrati sappiamo che alle nostre destra c'e la segreteria,
di fronte a noi un cortile e
dall'altro lato la caffetteria.

Al colpo d'occhio sappiamo dove sono i bagni.

Se entriamo da via Chiaravonti ci rendiamo subito conto della nostra posizione rispetto all'entrata principale.

SI PUO' FARE PINCH TO ZOOM
PER ESPLORARE PIU' DA VICINO

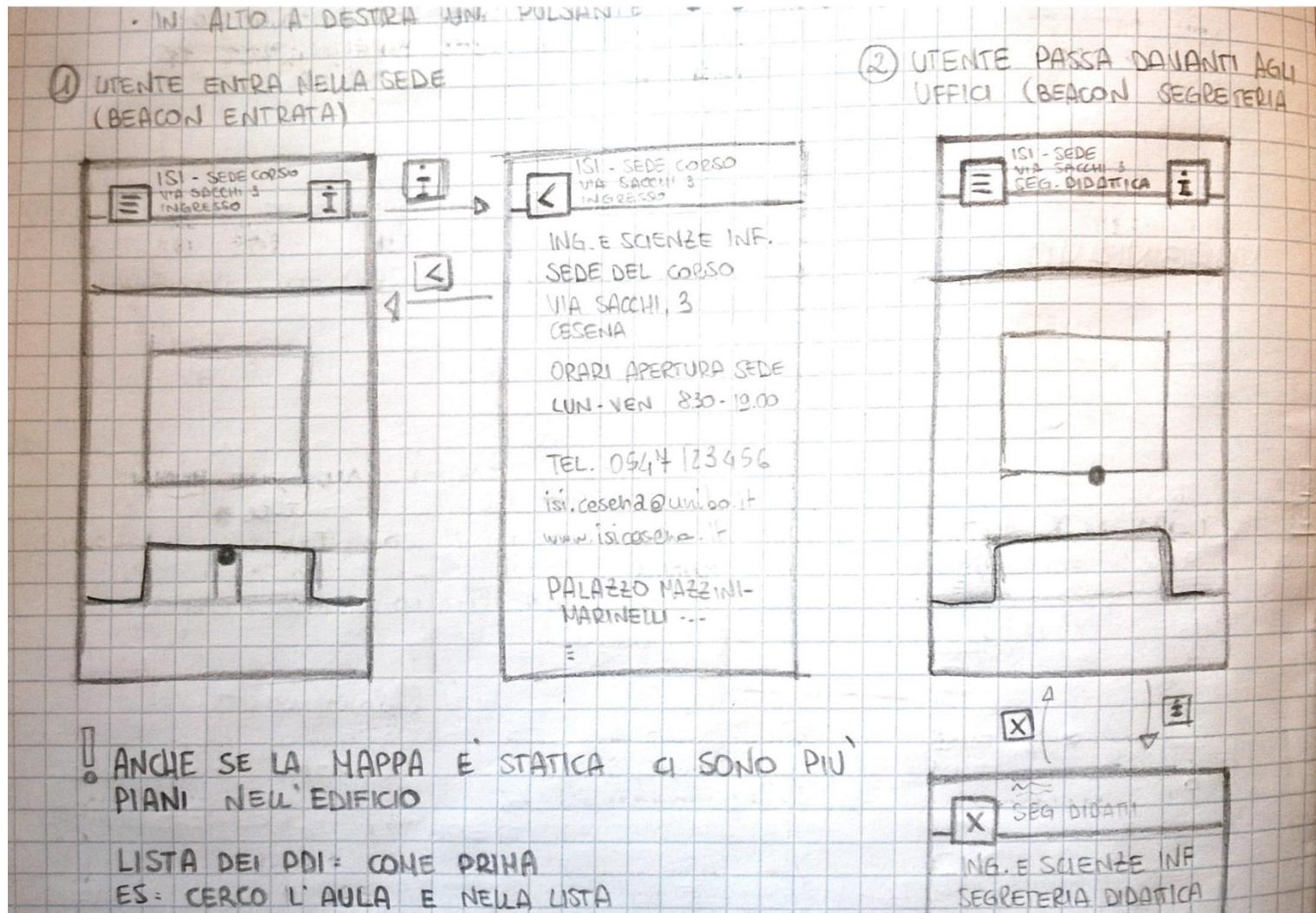
PER ESPLORARE IL PRIMO PIANO

IN QUESTO CASO, NON CI SONO PIU' LIVELLI MA UNO UNICO

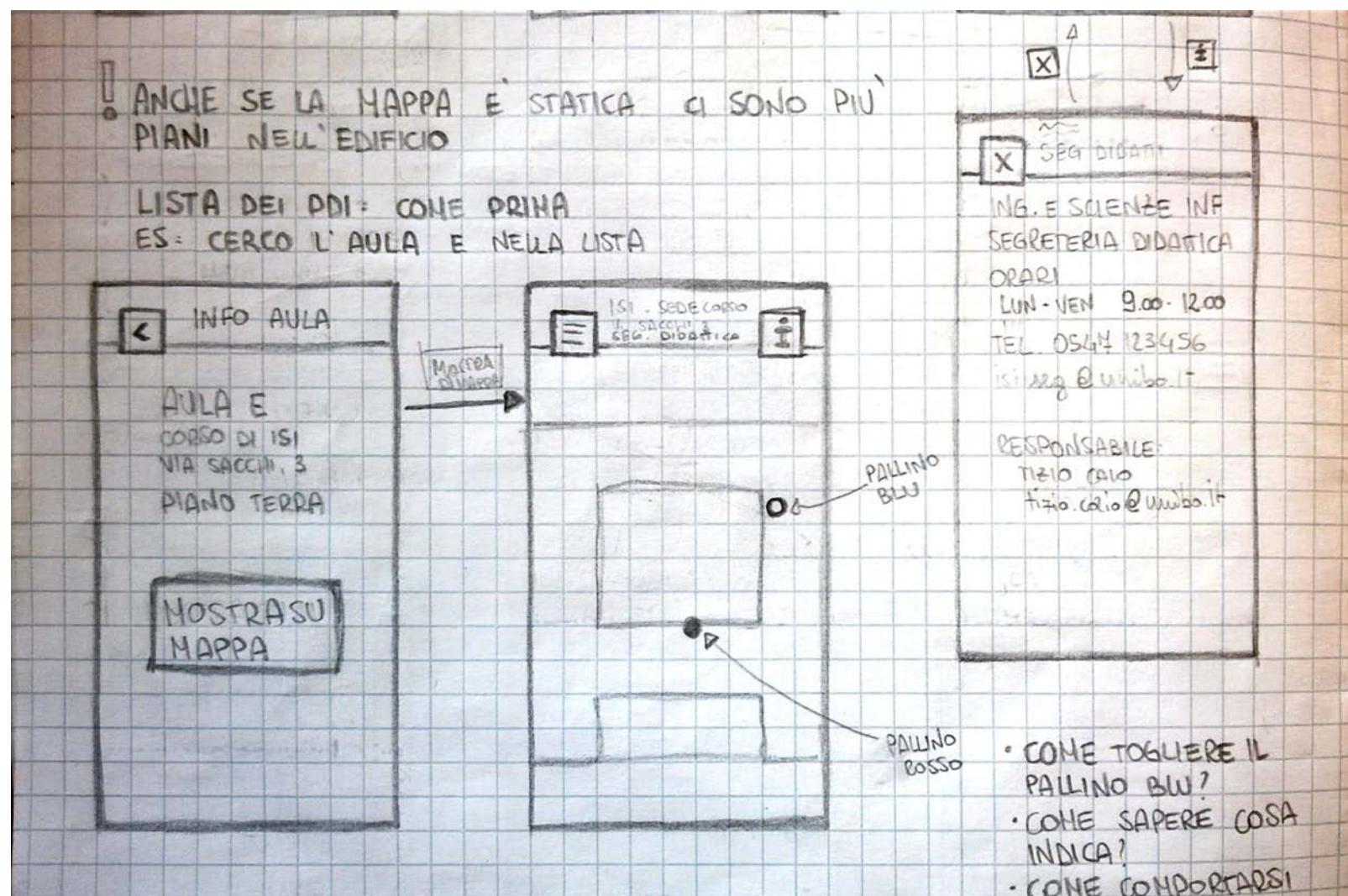
5 ZONE ZOOMABILI

2 EDIFICI
(PER IDENTIFICARE LE ZONE)

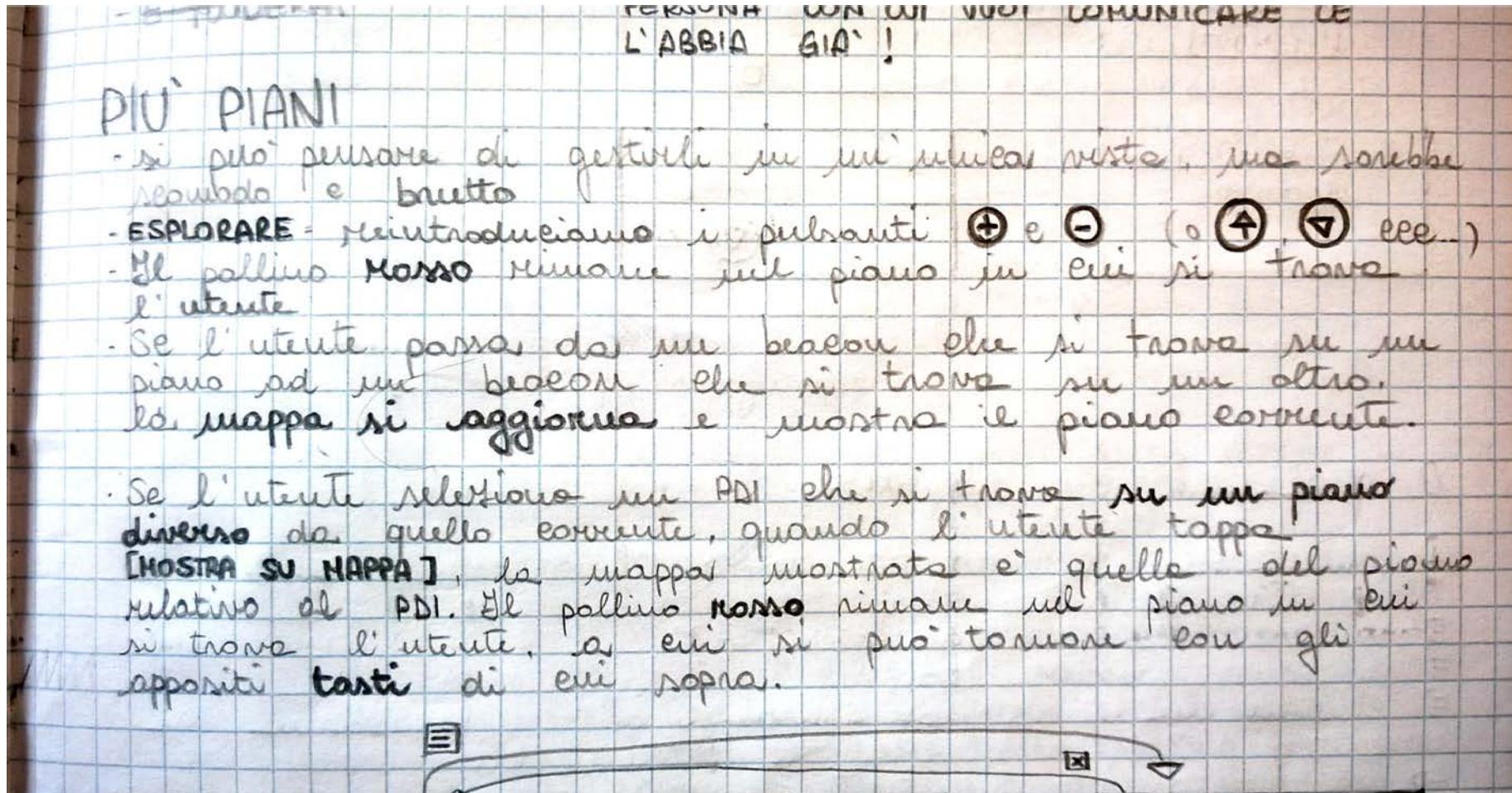
Storyboard con disegni e Sketch: un esempio



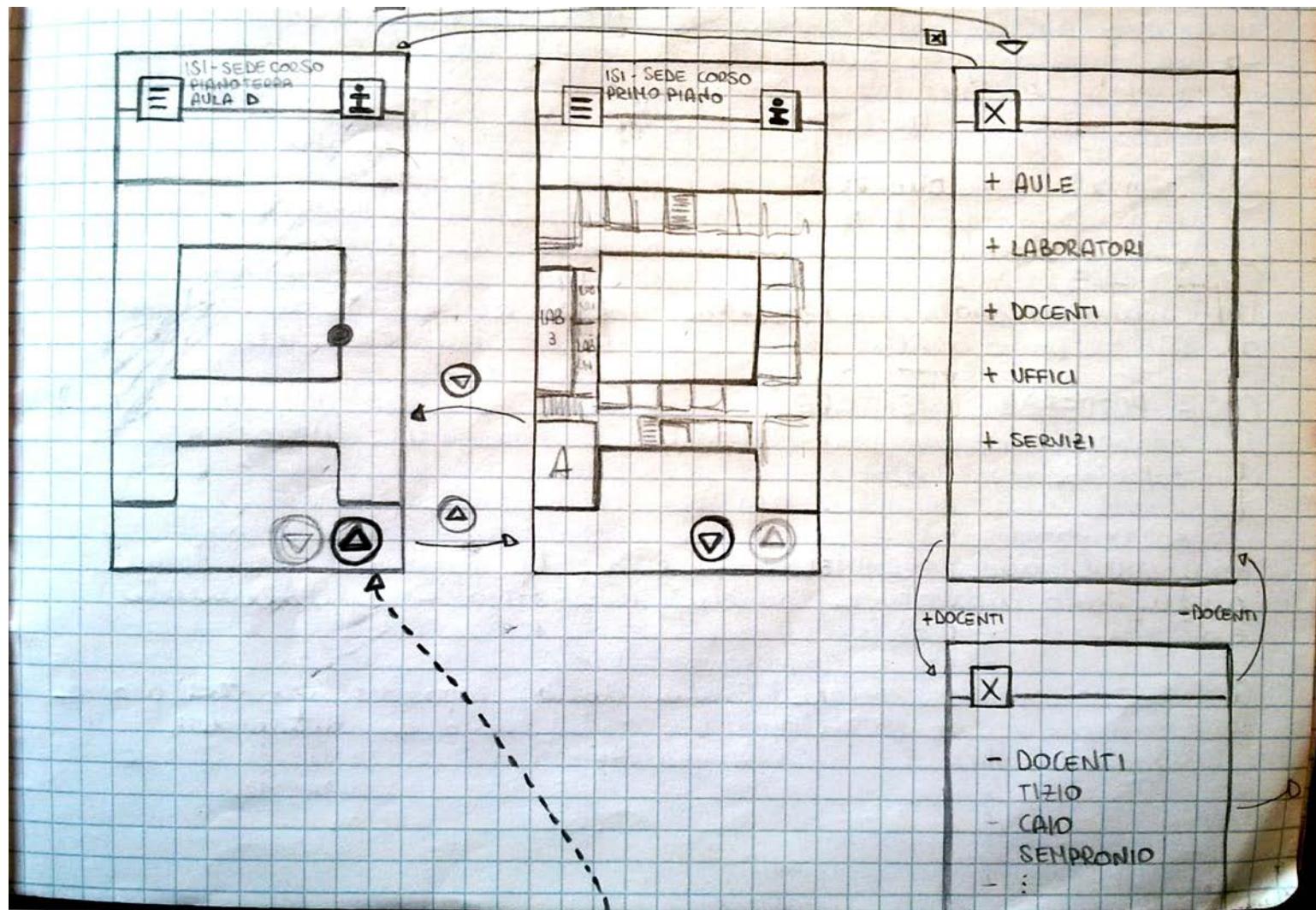
Storyboard con disegni e Sketch: un esempio



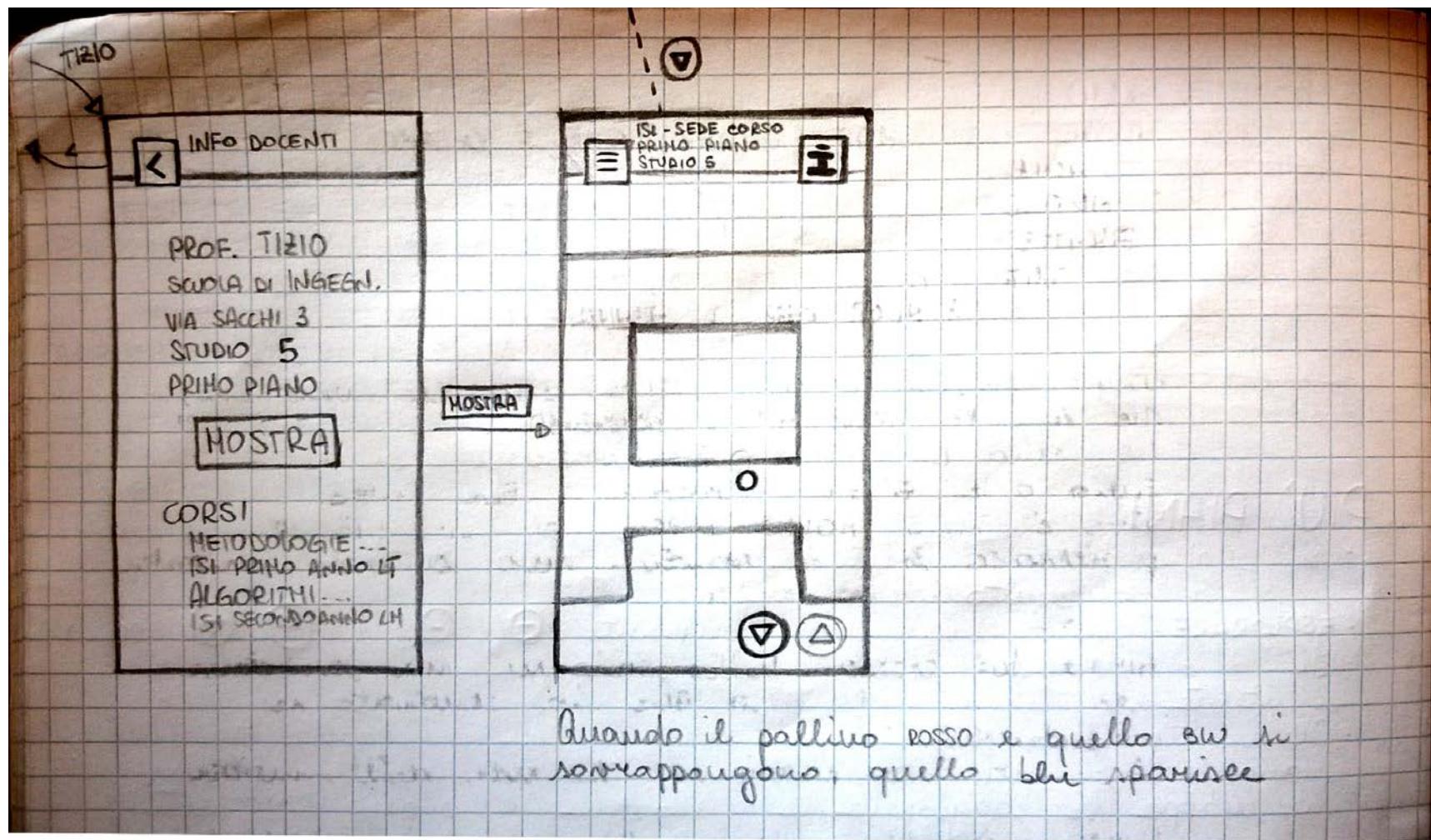
Storyboard con disegni e Sketch: un esempio



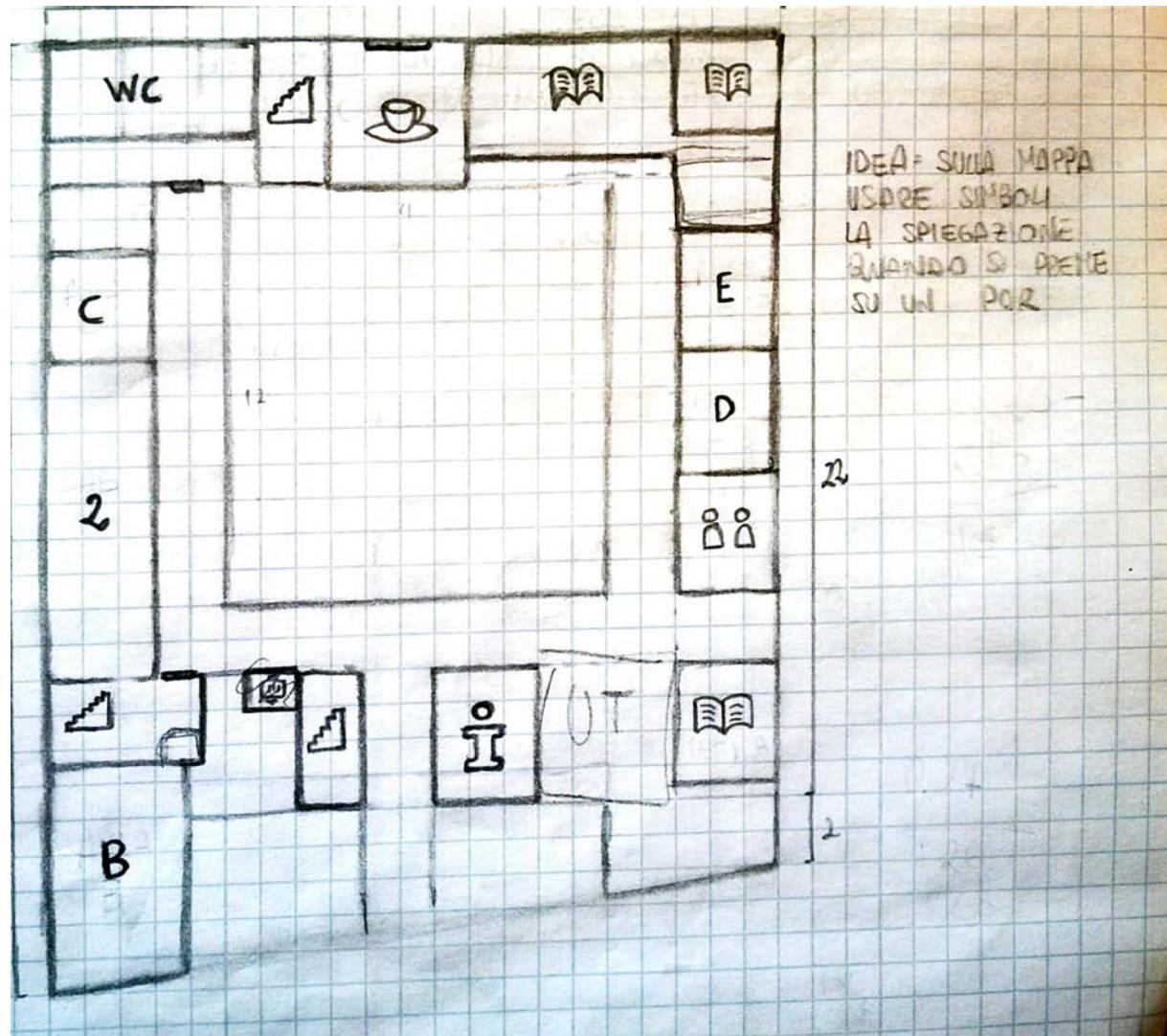
Storyboard con disegni e Sketch: un esempio



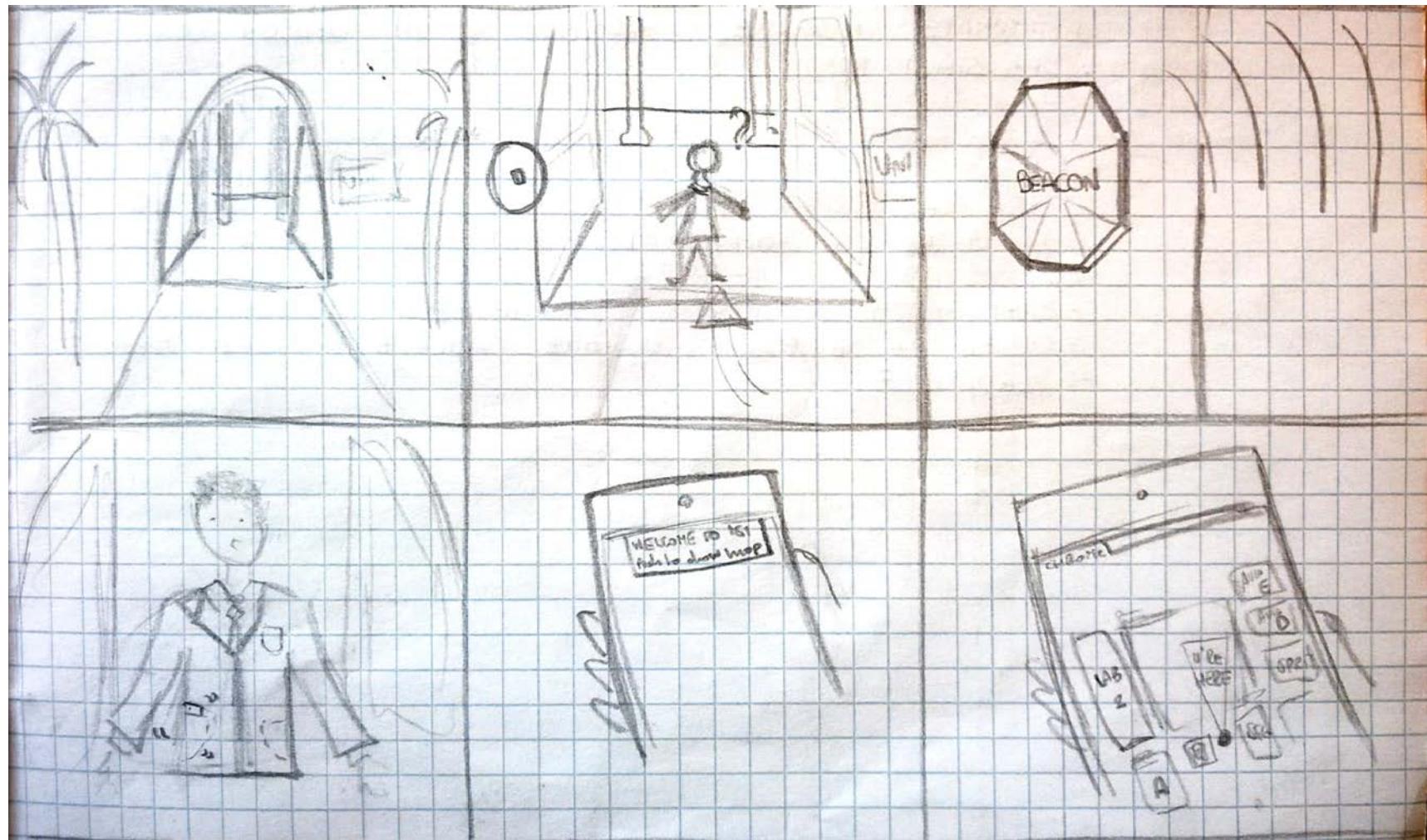
Storyboard con disegni e Sketch: un esempio



Storyboard con disegni e Sketch: un esempio



Storyboard con disegni e Sketch: un esempio

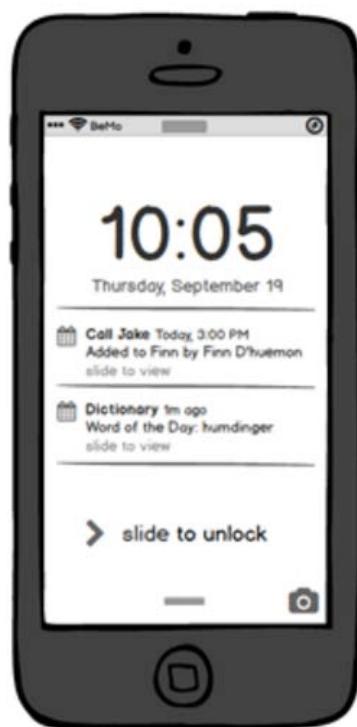


Mockup

- È una rappresentazione grafica della schermata (interfaccia) di interazione con il sistema
- Esistono molti software per crearli velocemente e in modo professionale (anche interattivi)
 - Balsamiq (<https://balsamiq.com/>)
 - Mockup (<http://mockup.io/about/>)
 - Pencil (<http://pencil.evolus.vn/>)
 - Adobe XD
(<https://www.adobe.com/it/products/xd.html>)

Mockup: Balsamiq

- Da un'idea di Giacomo “Peldi” Guilizzoni,
laureato in Informatica, all’Università di Bologna



The image shows a banking application interface titled "Bank of The World". The main header includes "My Account | Help | Log Out". Below the header, there are tabs for "Accounts", "Bill Pay", and "Transfer", with "All Accounts" selected and "Checking" chosen under it. The date "September 19th, 2013" is displayed, along with an "Available balance: \$4,217.48". A search bar with "search transactions" and a "Search" button are present. The transaction history table lists various transactions from September 1st to September 19th, 2013, including deposits, withdrawals, and checks. Each transaction row has a "View Details" link.

Date	Description	Type	Amount	Balance	Actions
processing	Balgreens	Debit	-\$27.33	\$4217.48	View Details
Sept 19	Bill's books	Debit	-\$18.37	\$4244.81	View Details
Sept 17	Bank of The World	Withdrawal	-\$80.00	\$4263.18	View Details
Sept 17	Check 465	Check	-\$27.00	\$4343.18	View Details View Check
Sept 16	Local 123	Debit	-\$12.17	\$4370.18	View Details
Sept 15	Bank of The World	Deposit	\$1711.68	\$4382.35	View Details
Sept 15	Monterey Market	Debit	-\$43.60	\$2670.67	View Details
Sept 15	The Cheese Board	Debit	-\$11.05	\$2714.27	View Details
Sept 14	BART	Debit	-\$20.00	\$2725.32	View Details
Sept 13	Bank of The World	Withdrawal	-\$80.00	\$2745.32	View Details

Mockup: Just in Mind

Working examples: <https://www.justinmind.com/examples>

SNASPLÖA WOMAN | MAN | INSPIRATION | OUR BRAND | BRANCHES | SOCIAL MEDIA | LOGIN | SHOPPING CART

Top Albums

Saengerkrieg
In Extremo

In EXREMO
Sternenise

Sterneisen
In Extremo

APOCALYPTICA
Plays Metallica...

Apocalyptica

Sagas
Equilibrium

In EXREMO
Sagastur

Apocalyptica

MMXI

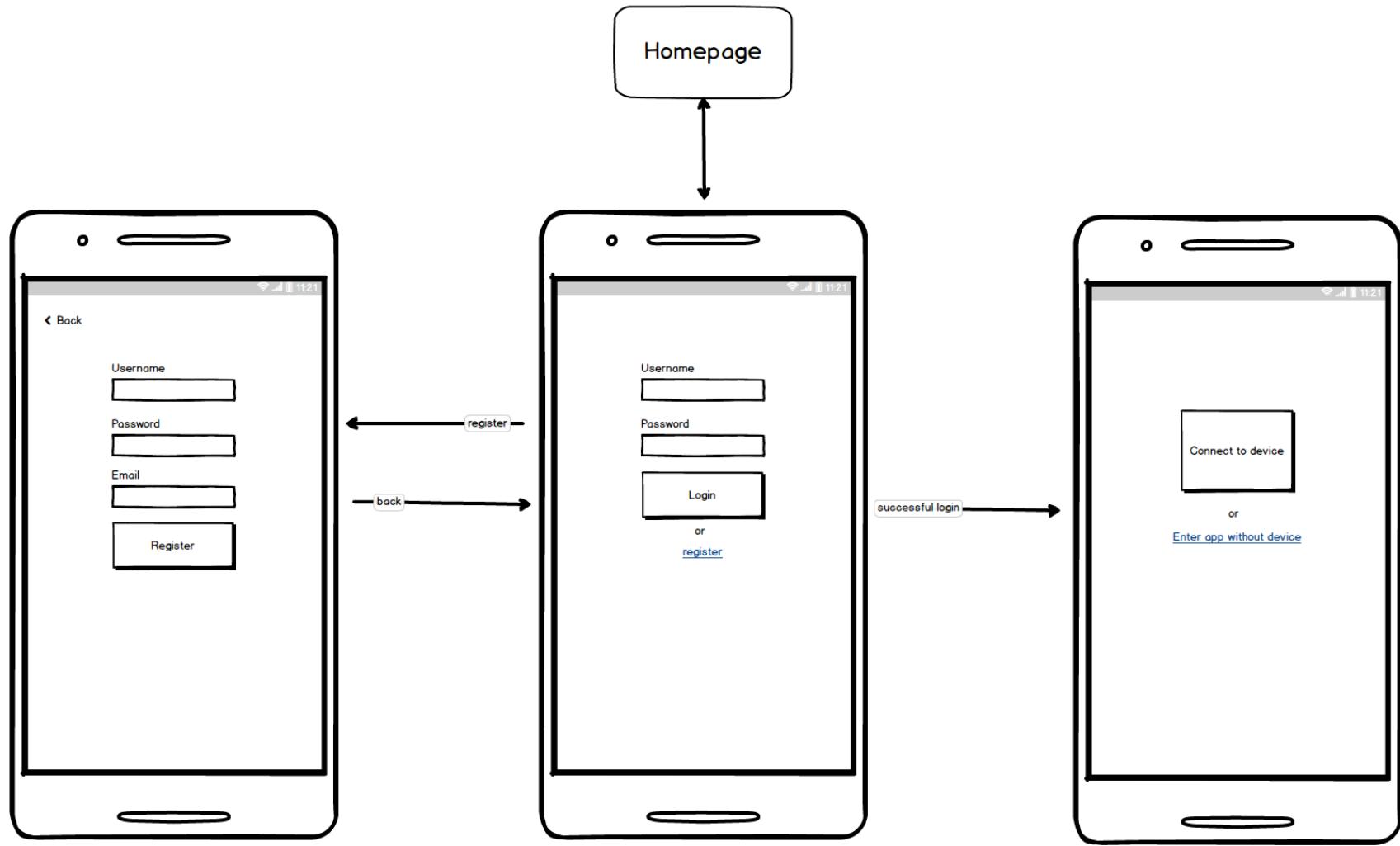
Winter 2011

Inspiration

Branches

SHOPPING | US | SERVICES | CONTACT US |

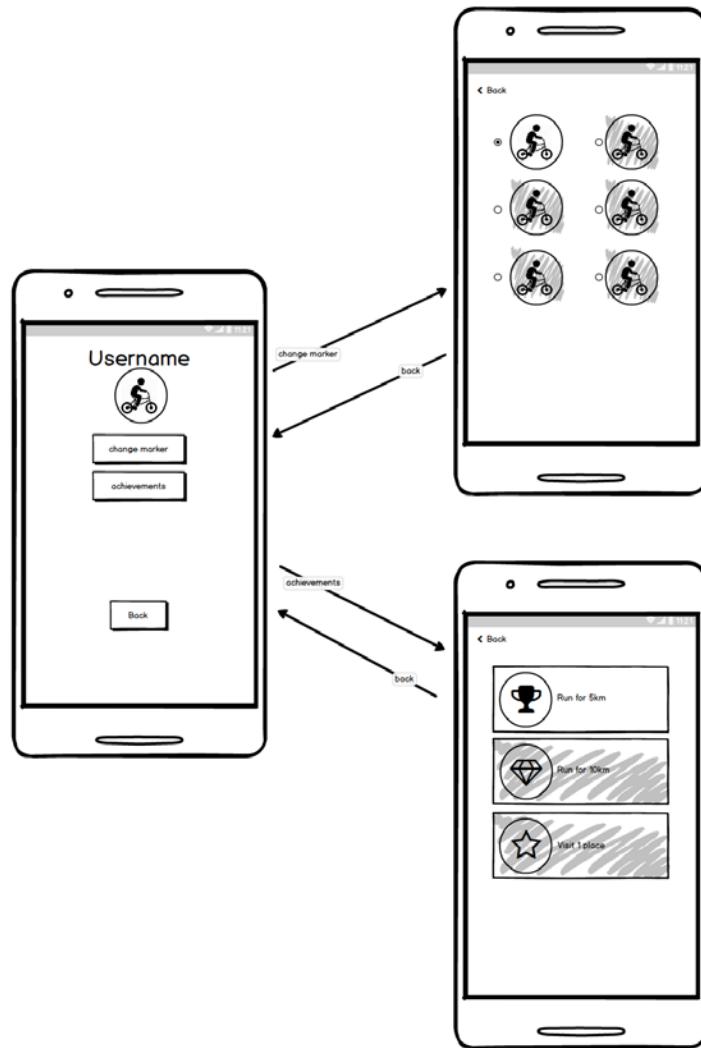
Storyboard con i mockup: un esempio



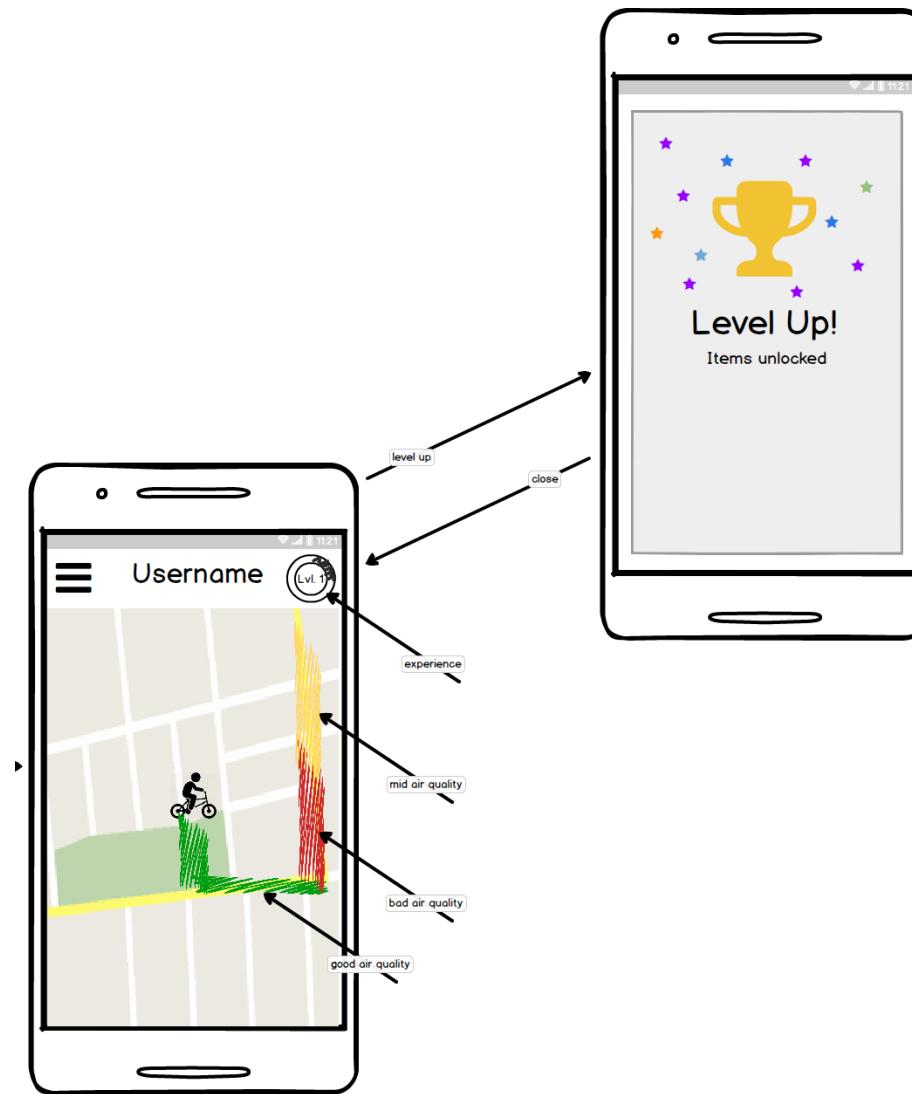
Storyboard con i mockup: un esempio



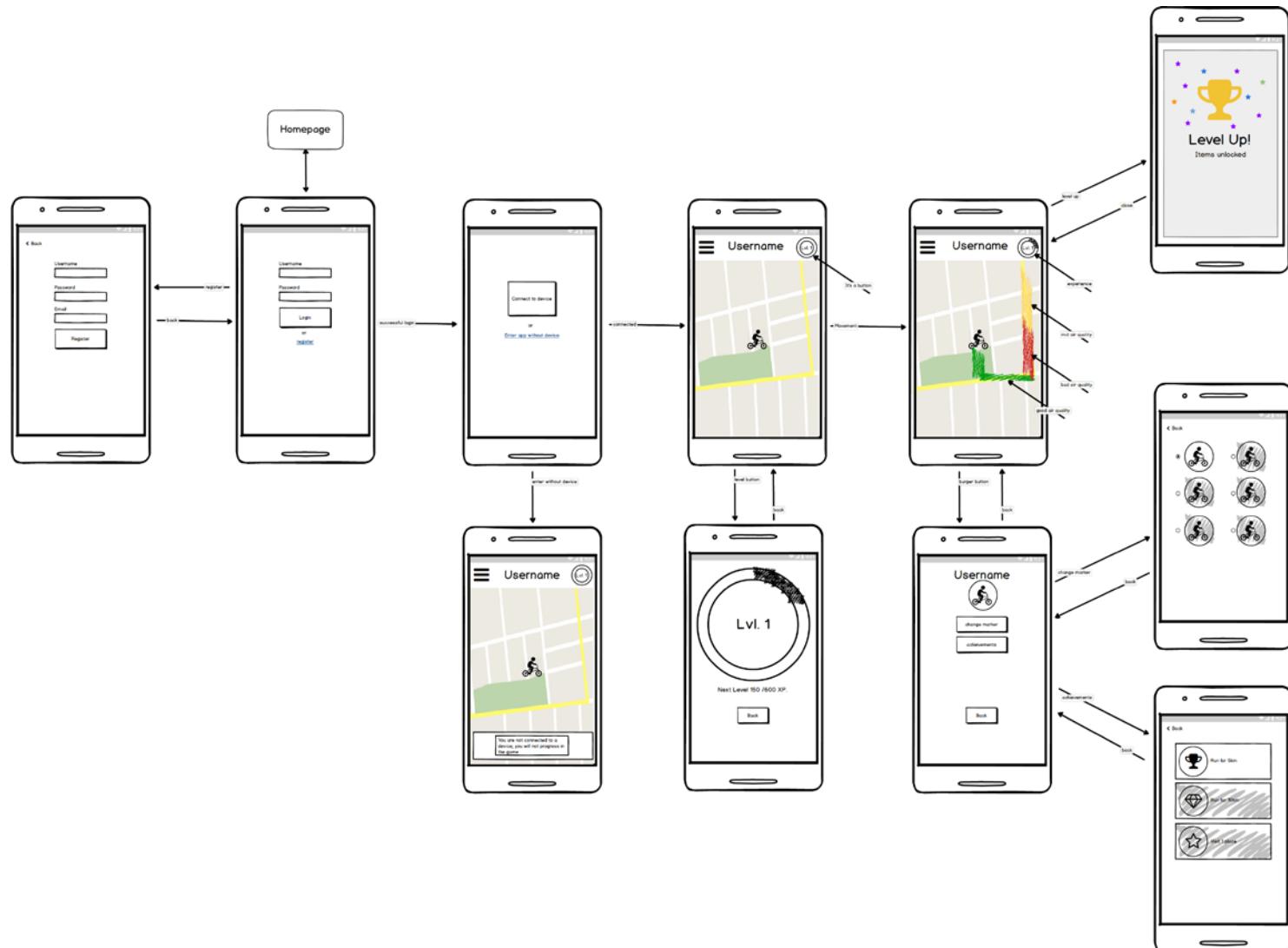
Storyboard con i mockup: un esempio



Storyboard con i mockup: un esempio



Storyboard con i mockup: un esempio



Q&A

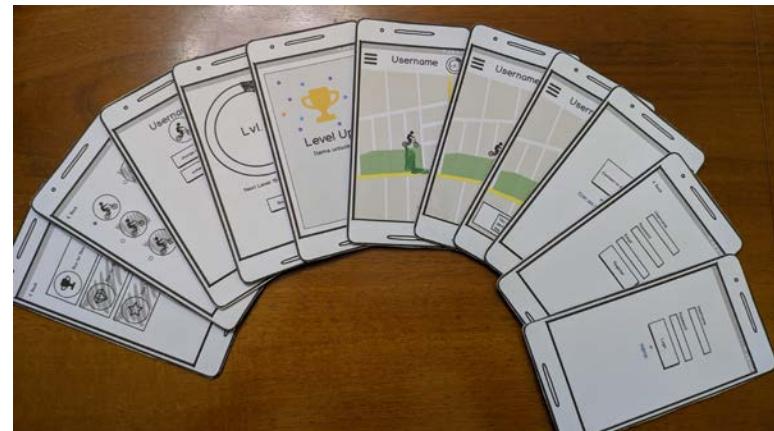
- Ci sono domande?

Focus Group

- Una volta che sono stati identificati i target group di utenti, una volta che sono stati definiti contesti e scenari d'uso, è possibile condurre dei **focus group**
- Metodologia che può essere usata in fase di analisi dei requisiti e in fase di design (per confermare/modificare alcune scelte già prese e/o per meglio direzionare lo sviluppo)
- Permette di esplorare e scoprire vantaggi e/o svantaggi di un limitato e predefinito insieme di funzionalità e feature

Focus Group

- Numero ideale di partecipanti: da 8 a 12
- Le caratteristiche dei partecipanti coinvolti dipendono dalla natura di quello di cui si vuole discutere/analizzare
- A partecipanti vengono presentate le funzionalità/feature che si vogliono esplorare e si invitano gli utenti a discuterne, dando feedback sugli argomenti proposti
- A supporto di questa attività, si possono utilizzare mockup, disegni e sketch, storyboard



Focus Group: definizione

- Un focus group è una intervista, alla quale partecipano più persone contemporaneamente, nello stesso gruppo, si incentive la discussione e l'interazione tra loro, in modo guidato
- In questo modo si raccolgono feedback e commenti da potenziali utenti
- La discussione tra gli utenti permette di collezionare informazioni che possono essere utili nelle fasi di design e sviluppo dell'applicazione (possono emergere elementi e risvolti inaspettati)

Focus Group: definizione

- Durante una sessione di focus group è possibile:
 - Raccogliere opinioni e feedback e atteggiamenti in reazione alle questioni di interessi dell'applicazione in oggetto
 - Testare le assunzioni e le idee iniziali relative alle specifiche di progetti
 - Incoraggiare la discussione su un topic specifico
 - Far emergere aspetti inattesi e/o formare dettagli aggiuntivi relativi ad una funzionalità specifica
- La durata di un focus group dovrebbe essere tra 1,5 e 2 ore
- Vanno considerate fasi iniziali (benvenuto, spiegazioni della procedura e delle attività) e conclusive (ringraziamenti e saluti), a corredo della discussione dei temi principali

Focus Group: step

1. Preparazione del focus group:

- Preparazione della sessione
 - Definizione e individuazione dei partecipanti e dei moderatori (un moderatore e un co-moderatore)
- Definizione di una lista dei topic di discussione
- Planning (definizione dell'agenda, inviti, individuazione degli spazi, organizzazione del layout dello spazio)

2. Conduzione

- Gestione dei tempi
- Registrazione (audio e/o video) + note
- Fairness: raccogliere feedback da tutti i partecipanti

3. Analisi dei risultati e preparazione dei report conclusivi

Experience Prototyping

- L'idea alla base di questa metodologia è quella di creare una esperienza completa per poter immergere l'utente nel contest d'uso dell'applicazione
- Il prototipo in questione potrebbe essere uno storyboard realizzato con disegni o sketch o con mockup, anche interattivi (senza bisogno di una vera applicazione funzionante), oppure una applicazione in versione beta
- Non si tratta di un prototipo completo: potrebbe essere focalizzato sulle funzionalità principali oppure più peculiari e originali
- L'intera esperienza potrebbe essere quindi simulata, in termini di interazione, sistema, logiche, ecc

Experience Prototyping

- La metodologia prevede il coinvolgimento di 1 utente per volta, al quale vengono proposti alcuni task (numero limitato, in genere non più di 3-4 task per sessione)
- All'utente viene proposto un task per volta e viene chiesto di indicare come si comporterebbe rispetto al prototipo: come interagisce con l'interfaccia per completare il task? Che tipo di feedback si aspetta dall'interfaccia in risposta alla sua interazione?
- Il tutto viene eseguito in un contesto reale, consiste in una simulazione molto realistica
- Il conduttore della sessione raccoglie feedback, commenti, suggerimenti da parte dell'utente
- E' necessario registrare la sessione (audio, video, foto)

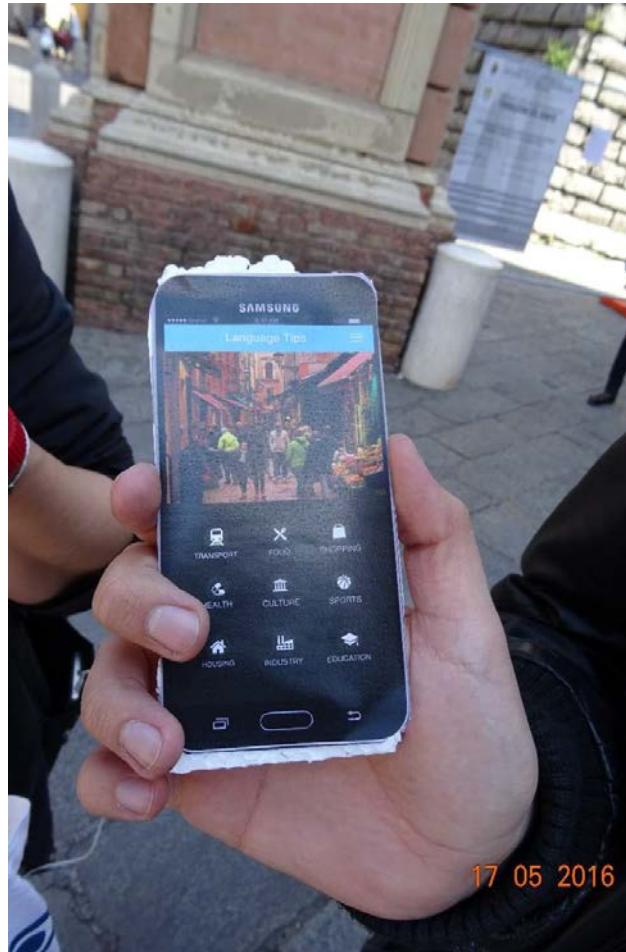
Experience Prototyping

- Potrebbero emergere nuove soluzioni, nuovi elementi dell’interfaccia, nuove funzionalità e/o nuovi servizi dell’applicazione
- Questa metodologia è considerata il modo migliore per osservare il comportamento dell’utente di fronte al prototipo realizzato (prima di averlo sviluppato)
- Metodologia spesso usata in contesti mobile
- Può essere usata sia in fase di design che in fase di testing (soprattutto in contesti di processi AGILE)

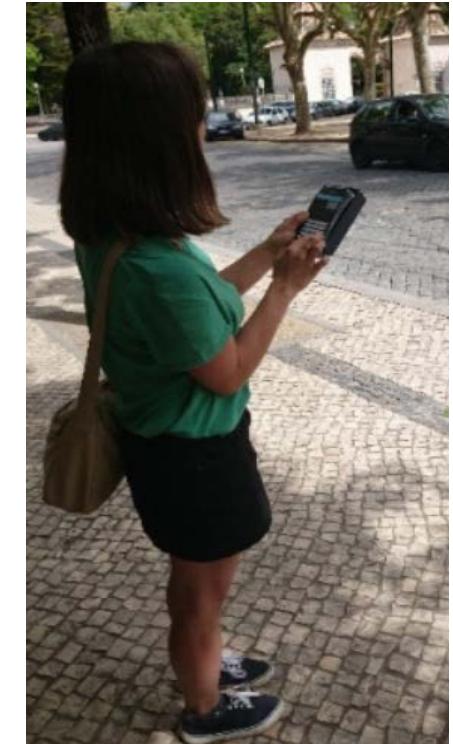
Experience Prototyping: un esempio



Experience Prototyping: un esempio



Experience Prototyping: un esempio



Experience Prototyping: step

1. Preparazione dell'experience prototyping:

- Preparazione della sessione (obiettivi, modelli del prototipo, individuazione dei partecipanti, definizione dei task che si vogliono valutare)
- Planning (scheduling, facilitatori e moderatori, mockup o prototipi, materiali di supporto)

2. Conduzione

- I partecipanti simulano come interagirebbero con i prototipi sulla base dei task proposti
- Think aloud protocol, interviste pre e post test, questionari

3. Analisi dei risultati e preparazione dei report conclusivi



Experience Prototyping: DOs e DONT's

- DOs
 - I prototipi devono essere create velocemente, testate con gli utenti e devono essere la base per miglioramenti successive: non importa che siano belli e non devono richiedere molto tempo per essere realizzati; la cosa importante è l'*idea*
 - Non importa se i prototipi non ottengono successo: è importante ottenere feedback, così da porterli migliorare
- DON'Ts
 - Non avere paura di osare e di usare l'immaginazione. I prototipi non devono essere funzionanti! Devono essere di supporto, possono essere fornite spiegazioni a voce a corredo, per migliorare/completare l'esperienza dei partecipanti

Q&A

- Ci sono domande?

Riferimenti

- Allman, Eric. "Managing technical debt." Communications of the ACM 55, no. 5 (2012): 50-55
- <http://c2.com/doc/oopsla92.html>
- Lehman, Manny M. "Laws of software evolution revisited." In European Workshop on Software Process Technology, pp. 108-124. Springer, Berlin, Heidelberg, 1996
- <https://thedigitalprojectmanager.com/project-management-methodologies-made-simple/>
- <https://thedigitalprojectmanager.com/agile-vs-waterfall/>
- The Design of Everyday Things (Donald Norman)
- Emotional Design (Donald Norman)

Riferimenti

- Experience Prototype, Service Design Tools:
<http://www.servicedesigntools.org/tools/21>
- Buchenau, M. and Suri, J.F., 2000, August. Experience prototyping. In Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques (pp. 424-433). ACM
<http://hci.stanford.edu/dschool/resources/prototyping/SuriExperiencePrototyping.pdf>
- Strömberg, H., Pirttilä, V. and Ikonen, V., 2004. Interactive scenarios—building ubiquitous computing concepts in the spirit of participatory design. Personal and Ubiquitous Computing, 8(3-4), pp.200-207