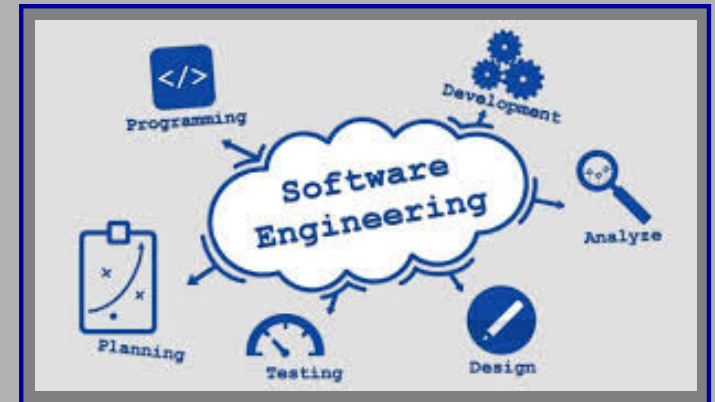




Introduzione

Anno accademico 2023/2024
Ingegneria del Software

Tullio Vardanega, tullio.vardanega@unipd.it





Cosa facciamo qui (*what*) – 1/2

- ❑ **Apprendiamo metodi e pratiche di lavoro alla base della professione informatica**
 - **Gestire il tempo**
 - Disponibilità, scadenze, conflitti, priorità
 - **Collaborare**
 - Fissare obiettivi, dividersi compiti, verificare progressi, riportare difficoltà
 - **Assumersi responsabilità**
 - Fare quanto pattuito, agire al meglio delle proprie capacità, auto-valutarsi prima di valutare
 - **Auto-apprendere**
 - “Imparare a imparare”, essenziale competenza trasversale
- ❑ **Integriamo progressivamente la teoria con la pratica**



Perché lo facciamo (*why*)

- ❑ Per avvicinarci a modo di lavorare (*way of working*) **professionale**
- ❑ Cioè: «operante allo stato dell'arte»
 - Per conoscenze tecnologiche e metodologiche
- ❑ Lo stato dell'arte nel dominio informatico avanza continuamente
 - Per questo dobbiamo imparare a «colmare i buchi» con l'auto-apprendimento



Come si apprende SWE

- ❑ **Costruendo **incrementalmente** il proprio glossario**
 - **Basandolo inizialmente sulla teoria**
 - **Consolidandolo con la pratica**
 - **Confrontandolo con i colleghi**
 - Unendo conoscenze parziali, correggendosi reciprocamente
- ❑ **Il glossario serve a cogliere, fissare, ritrovare i concetti chiave della materia**
 - **Per evitare di «scivolarci sopra» con superficialità**
- ❑ **La conoscenza passa dalla comprensione profonda, sperimentata, dei significati racchiusi in quei concetti**
 - **Non ricordare, ma riconoscere (non è la stessa cosa!)**



Come lo facciamo (*how*) – 1/2

- ❑ **Tramite un *progetto* didattico collaborativo**
 - **Promosso da un proponente esterno**
 - **Con esigenze e obiettivi funzionali innovativi**
 - **Complesso, impegnativo, visionario**
 - **Tecnologicamente avanzato**
- ❑ **Confermando le conoscenze acquisite tramite una prova scritta**



Fonte: Harold Kerzner (1940-), uno dei maggiori esperti mondiali di *project management*

❑ Progetto

○ Insieme di attività che

- Devono raggiungere determinati obiettivi a partire da determinate specifiche
- Hanno una data d'inizio e una data di fine fissate
- Dispongono di risorse limitate (persone, tempo, denaro, strumenti)
- Consumano tali risorse nel loro svolgersi

○ L'uscita di un progetto è un prodotto composito

- SW sorgente/esequibile, librerie, documenti, manuali
- Raccolti ed esposti in modo organizzato



I costituenti di un progetto – 1/2

□ Pianificazione

- Gestire risorse (persone, tempo, denaro, strumenti) in modo responsabile, continuativamente, in funzione degli obiettivi

□ Analisi dei requisiti

- Definire cosa bisogna fare

□ Progettazione (→ *design*)

- Definire come farlo

Studieremo e praticheremo tutte queste attività

□ Realizzazione (→ *implementation*)

- Farlo, perseguendo qualità
- Accertando l'assenza di errori od omissioni
- Accertando che i risultati soddisfino le attese



Cosa non è un progetto – 1/2

- ❑ *One is blinded to the fundamental uselessness of their products, by the sense of achievement one feels in getting them to work at all*
- ❑ *In other words, their fundamental design flaws are completely hidden by their superficial design flaws*

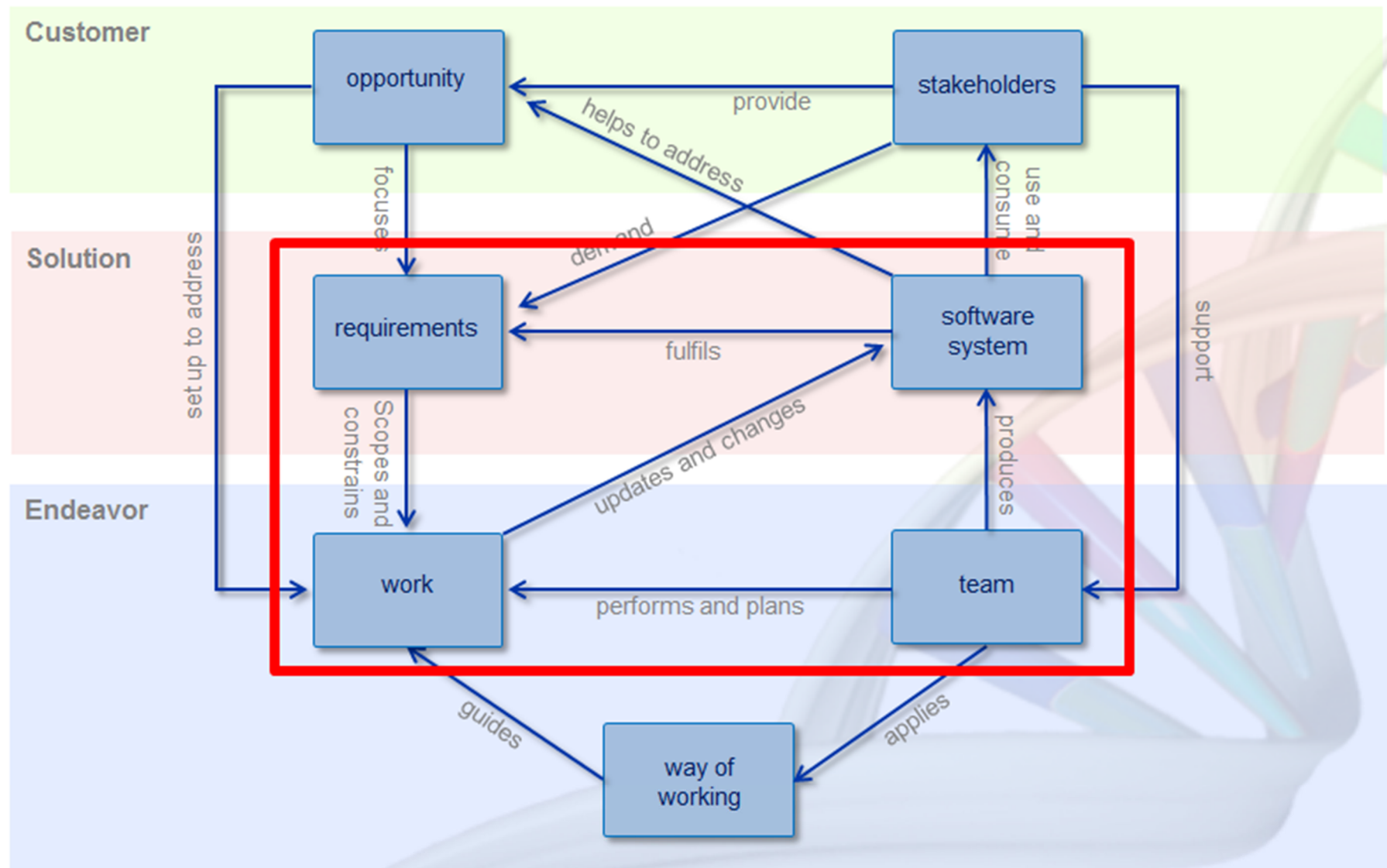
Fonte: Douglas Adams, “The Hitchhikers Guide to the Galaxy”, 1979



Cosa non è un progetto – 2/2

- ❑ Nella filosofia greca, arte (*μίμησις*) significa copia / riproduzione, bella e significativa, della natura
 - Tangibile o spirituale
- ❑ In Latino, *ars* significa «abilità professionale»
 - Significato rimasto in vigore fino all'Illuminismo
- ❑ Con il Romanticismo, arte è divenuta «espressione di contenuto emozionante»
- ❑ Quando qui diciamo «stato dell'arte» intendiamo il significato latino
- ❑ Un progetto non è arte romantica, ma «professionale»

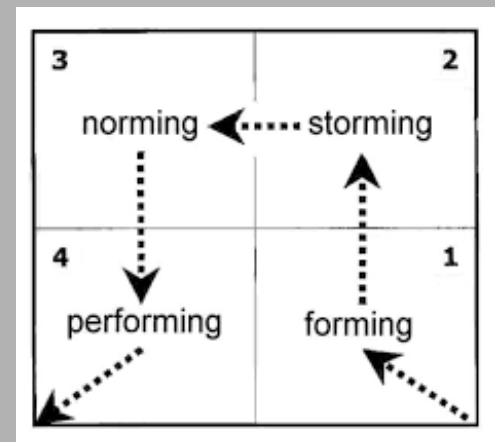
I costituenti di un progetto – 2/2





□ **Teamwork**

- **Lavoro collaborativo per raggiungere un obiettivo comune in modo efficace ed efficiente**
- **Dividere il lavoro rende inter-dipendenti i membri del *team***
- **La gestione di questa inter-dipendenza richiede il rispetto di regole e di buone pratiche**
 - Comunicazioni aperte e trasparenti: risoluzione dei conflitti
 - Costruzione e preservazione delle fiducia reciproca: condivisione e collaborazione
 - Assunzione di responsabilità: coordinamento
 - Condivisione dei rischi
- **La sua base è un solido *way of working***





❑ ***Stakeholder*** (portatore di interesse)

○ **Tutti coloro che a vario titolo hanno influenza sul prodotto e sul progetto**

- La comunità degli utenti (che usa il prodotto)
- Il committente (che compra il prodotto)
- Il fornitore (che sostiene i costi di realizzazione)
- Eventuali regolatori (che verificano la qualità del lavoro)

❑ ***Way of working***

○ **Come organizzare al meglio le attività di progetto**

- In modo professionale



Ingredienti

- ❑ Per svolgere un progetto potendo confidare nel suo successo serve **ingegneria**

Engineering: application of scientific and mathematical principles to practical ends

Fonte: American Heritage Dictionary

- Applicazione (non creazione!) di principi noti e autorevoli: ***best practice***
- Practical ends spesso civili e sociali, associati a responsabilità etiche e professionali



❑ *Software engineering* [SWE]

- Disciplina per la realizzazione di **prodotti SW** così impegnativi da richiedere il dispiego di attività collaborative
- Capacità di produrre “in grande” e “in piccolo”
- Garantendo qualità: **efficacia**
- Contenendo il consumo di risorse: **efficienza**
- Lungo l'intero periodo di sviluppo e di uso del prodotto: **ciclo di vita**



Glossario



❑ Efficacia

- Misura della capacità di raggiungere l'obiettivo prefissato

❑ Efficienza

- Misura dell'abilità di raggiungere l'obiettivo impiegando le risorse minime indispensabili



SWE rispetto a se stessa

- ❑ **Un sistema SW è tanto più utile quanto più è usato**
 - **Metrica:** integrale della sua intensità d'uso nel tempo
- ❑ **Più lunga la vita d'uso di un prodotto, maggiore la sua intensità di **manutenzione****
 - **Attività necessarie a garantire l'uso continuativo del prodotto**
 - Reattivamente: per correzione dopo malfunzionamento
 - Proattivamente: anticipando necessità future
- ❑ **Il costo di manutenzione ha varie componenti**
 - **Mancato guadagno, perdita di reputazione, recupero o reclutamento esperti, sottrazione di risorse ad altre attività**
- ❑ **I principi SWE puntano ad abbassare tali costi**
 - **Sviluppando SW più facilmente manutenibile**



Cos'è l'ingegneria del *software* – 1/2

❑ Nasce nel 1968

- Conferenza NATO (☹) 7-11/10/1968 @ Garmisch, D

❑ Raccogliere, organizzare, consolidare la conoscenza (*body of knowledge*) necessaria a realizzare progetti SW con efficacia ed efficienza

- Collezione e manutenzione migliorativa di *best practice*

❑ Applicare principi ingegneristici calati nella produzione del SW



Cos'è l'ingegneria del software – 2/2

*L'approccio sistematico, disciplinato e quantificabile
allo sviluppo, l'uso, la manutenzione e il ritiro del SW*

Fonte: Glossario IEEE

❑ **Sistematico**

- Modo di lavorare metodico e rigoroso
- Che conosce, usa ed evolve le *best practice* di dominio

❑ **Disciplinato**

- Che segue le regole che si è dato

❑ **Quantificabile**

- Che permette di misurare l'efficienza e l'efficacia del suo agire

Planning



Completed

Design



Completed

Implementation



In Progress

Testing



Waiting

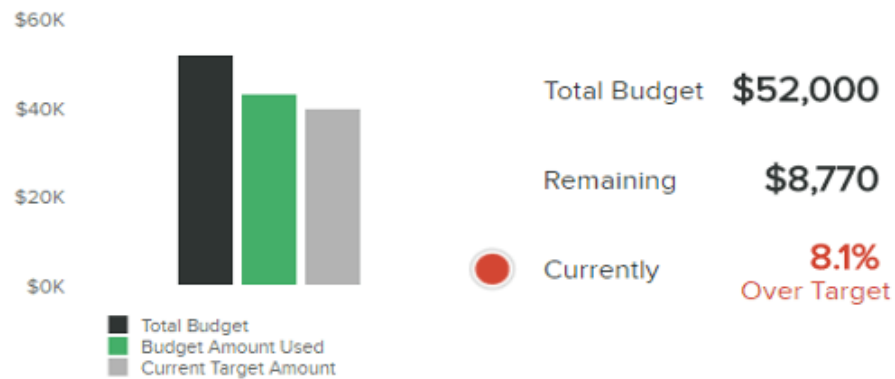
Projected
Launch Date



107 Days

Friday, December 15

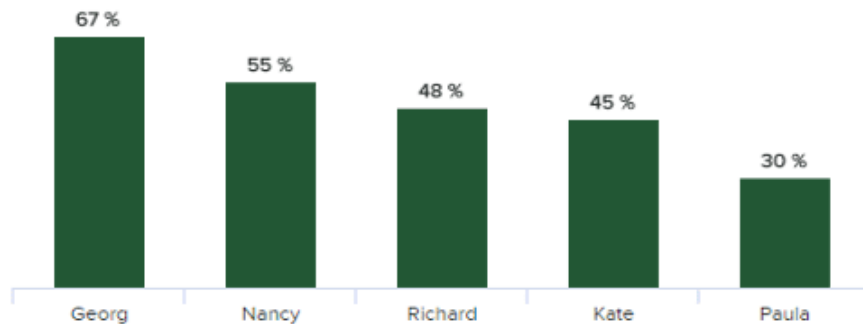
Project Budget



Overdue Tasks

Overdue	Task	Deadline	Employee
1 Day	Status Update For Board	2017-08-15	Paula
4 Days	Finish UX Optimizations	2017-08-06	Kate
10 Days	Configure Mobile View	2017-08-01	Nancy
24 Days	Relational Database Connections	2017-07-18	Georg

Workload



Upcoming Deadlines

Employee	Task	Deadline	Workload
Kate	Interactive Dashboard Features	2017-08-15	34%
Georg	Facebook API Connector	2017-08-06	56%
Nancy	Set-Up Test Environment	2017-08-01	15%
Paula	Finalize Testing Plan	2017-07-18	11%

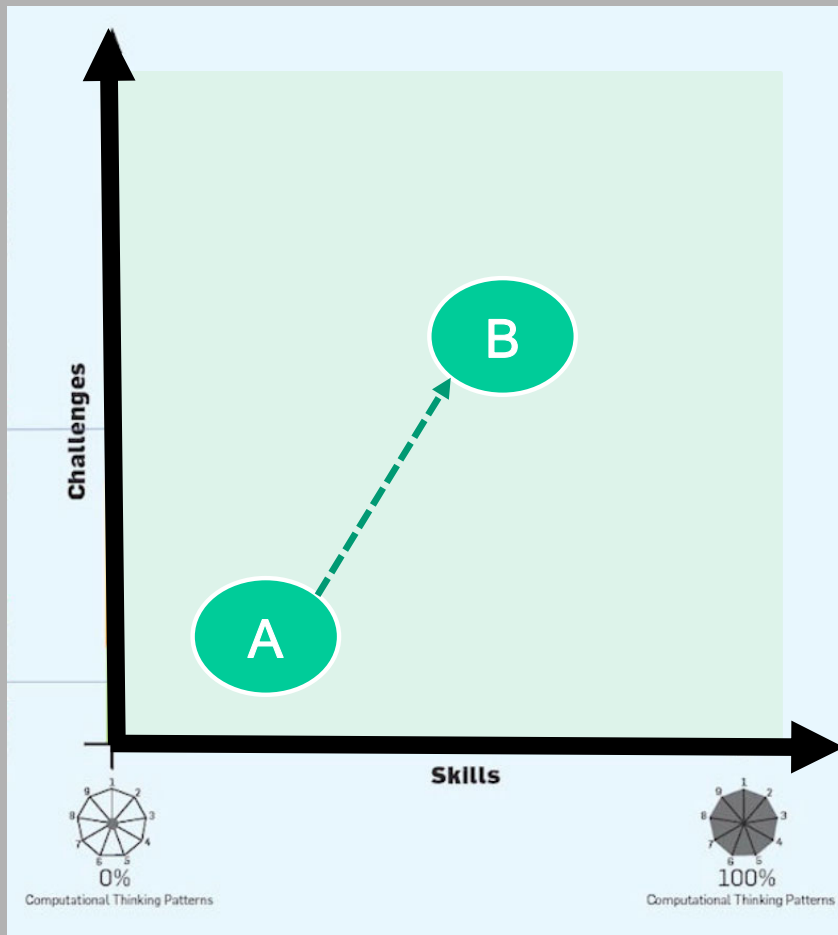


***Software engineering* come professione**

- ❑ **Realizzare parti di un sistema complesso con la consapevolezza che potranno essere usate, completate e modificate da altri**
- ❑ **Comprendere il contesto d'uso in cui si colloca il sistema**
 - Per chi, in quale situazione, tramite quale dispositivo
- ❑ **Attuare compromessi intelligenti e lungimiranti tra obiettivi e vincoli contrastanti**
 - Costi – qualità
 - Risorse – disponibilità
 - Esperienza utente – facilità di realizzazione



Perché facciamo così (*why*) – 1/2

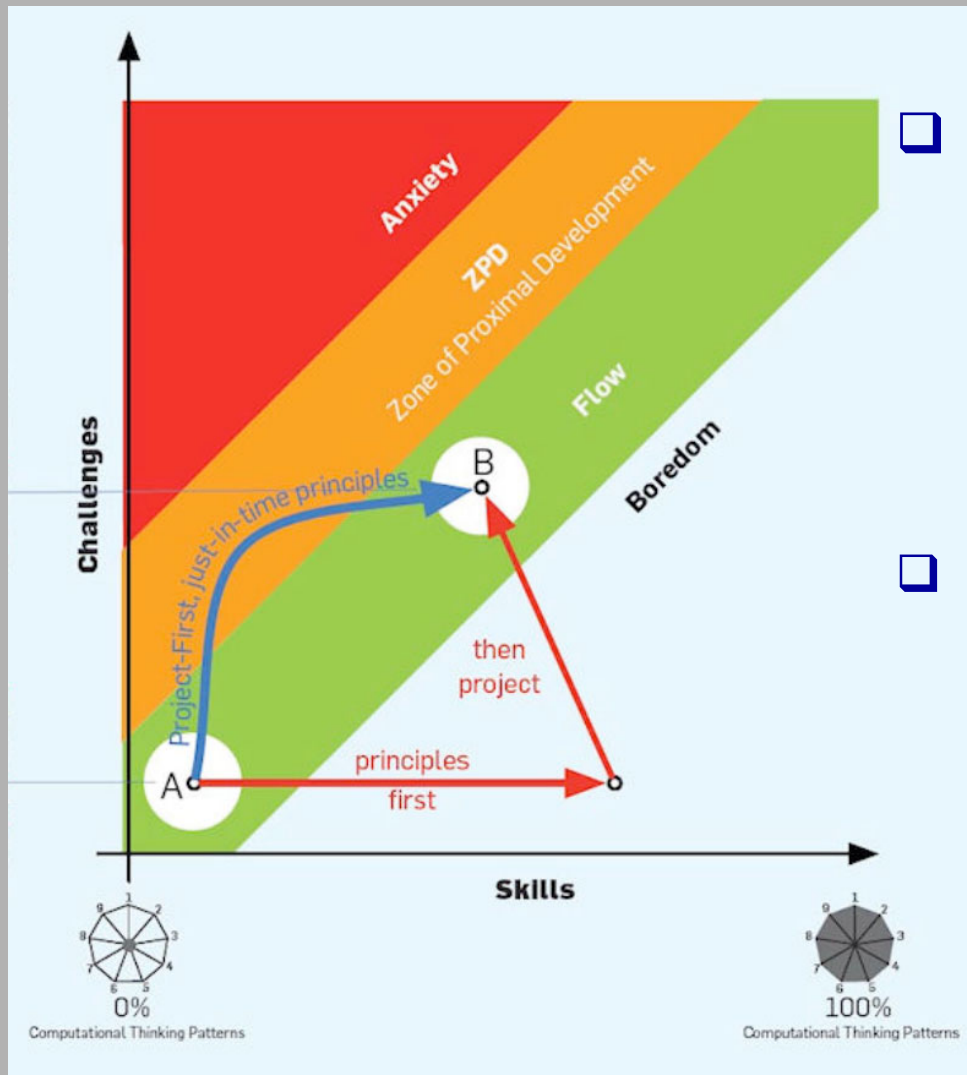


- ❑ *Student acquisition of [methodical] skills advances in response to challenges*
- ❑ *Pedagogical approaches can be described as instructional trajectories connecting a skill/challenge starting point (A) with a destination point (B)*

Fonte: *D.C. Webb, A. Repenning, K.H. Koh, "Toward an emergent theory of broadening participation in computer science education", Proc. 43rd ACM Computer Science Education symposium, 173-178 (SIGCSE '12)*



Perché facciamo così (*why*) – 2/2



- ❑ *The **project-first just-in-time-principles** approach lies in the Zone of Proximal Flow (ZPF)*
 - *The ideal condition for learning*
- ❑ *The ZPF orchestrates students' take in best practices with assistance and tool use*



Con quale quantità di impegno

- ❑ **12 crediti** → **300** ore di lavoro complessivo
- ❑ **96** ore in lezioni di teoria, pratica, monitoraggio attività, esercitazioni
- ❑ **150** ore nel progetto didattico
 - ~95 in attività rendicontate
 - ~ 55 per auto-formazione su strumenti e metodi di lavoro utili al progetto
 - **Ore produttive** (non tempo trascorso): devono portare i risultati attesi
- ❑ ~50 ore per studio personale in preparazione alla prova scritta e revisioni di avanzamento





Regole e vincoli – 1/2

- ❑ **Partecipa solo chi soddisfa le propedeuticità**
 - **Basi di Dati**
 - **Programmazione a oggetti**
- ❑ **Chi ha altri “arretrati”, li sani prima di cimentarsi con il progetto**
- ❑ **I gruppi sono formati in sessione pubblica**
 - **Gli aventi diritto si registrano in tabellone condiviso pubblicato sulla pagina IS @ Moodle STEM**





Regole e vincoli – 2/2

- ❑ **L'impegno dispiegato per raggiungere gli obiettivi di progetto ha limite superiore stretto (95 ore produttive)**
 - Per lasciare congruo spazio agli altri obblighi personali
- ❑ **Gli obiettivi di progetto didattico sono elastici**
 - Per essere sostenibili entro i limiti di impegno fissati
 - Tra un min e un max concordati dinamicamente con il docente e il proprio proponente
- ❑ **Svolgibile solo in blocchi di tempo «solidi»**
 - Quanto necessario per raggiungere specifici obiettivi
- ❑ **Non lascia spazio sufficiente per «arretrati»**



Gli argomenti che tratteremo

- ❑ **Processi, ciclo di vita e modelli di sviluppo del SW**
 - ❑ **Gestione di progetto**
 - ❑ **Amministrazione IT**
 - ❑ **Analisi dei requisiti**
 - ❑ **Progettazione**
 - ❑ **Documentazione**
 - ❑ **Qualità**
 - ❑ **Verifica e validazione**
- ❑ **UML: diagrammi dei casi d'uso**
 - ❑ **UML: diagrammi delle classi e dei *package***
 - ❑ **UML: diagrammi di sequenza e di attività**
 - ❑ ***Design pattern*: creazionali, comportamentali, architetturali**
 - ❑ **Stili architetturali**
 - ❑ **Principi SOLID**
-



Come lo facciamo (*how*) – 2/2

□ Tramite tre diversi tipi di attività d'aula

- T: Teoria (Vardanega)
- P: Pratica (Cardin)
- PD: Monitoraggio del progetto didattico (entrambi)
- E: Esercitazioni (entrambi)

□ Stile di lavoro

- Alle lezioni T si viene avendo studiato l'argomento
- Alle lezioni P si dà seguito facendo esercizi
- Nelle lezioni PD si dialoga, approfondendo temi, questioni e criticità



Fonti e risorse – 1/3

□ Faremo riferimento a

- ***Software Engineering*, 10th ed., 2014, di Ian Sommerville, edito da Addison Wesley (Pearson Education)**
- ***Guide to the Software Engineering Body of Knowledge (SWEBOK v3)*
IEEE Computer Society
Software Engineering Coordinating Committee**

□ Che ci aiutano a familiarizzarci con le aree di conoscenza della disciplina SWE

- **Insieme a materiali di approfondimento associati agli argomenti di lezione («Per approfondire»)**



Fonti e risorse – 2/3

Table I.1. The 15 SWEBOK KAs

Software Requirements
Software Design
Software Construction
Software Testing
Software Maintenance
Software Configuration Management
Software Engineering Management
Software Engineering Process
Software Engineering Models and Methods
Software Quality
Software Engineering Professional Practice
Software Engineering Economics
Computing Foundations
Mathematical Foundations
Engineering Foundations

**Noi qui ci
occupiamo
di queste 10**



Fonti e risorse – 3/3

❑ Come altri testi di consultazione useremo

- E. Gamma, R. Helm, R. Johnson, J. Vlissides
Design Patterns, 2002
Addison-Wesley (Pearson Education Italia)
- C. Larman
Applicare UML e i *pattern*
Pearson Italia (5° edizione, 2020)

❑ Insieme alle moltissime risorse digitali disponibili in rete su quei temi