DIDATTICA INFORMATICA 2018 Best Paper Award

Developers' week: Alternanza Scuola-Lavoro rovesciata

Marcello Missiroli[1,2], Daniel Russo[1], Paolo Ciancarini[1], and Paolo Torricelli[3]

[1] DISI, Università di Bologna
{marcello.missiroli,daniel.russo,paolo.ciancarini}@unibo.it
[2] IIS Fermo Corni e Liceo, Modena prof.missiroli@gmai.com
[3] GN Srl - General Networking, Modena ptorric@gnet.it

Sommario Presentiamo un nuovo modello di alternanza di scuola-lavoro, che riunisce tanto elementi tipici dello *stage* aziendale classico quanto quelli derivanti dalla simulazione di impresa. Si realizza lo sviluppo di un prodotto software sotto la direzione di un'impresa seppur svolto all'interno dell'istituto scolastico, rovesciando la prospettiva classica dello stage aziendale. Riportiamo i risultati di una sperimentazione giudicata estremamente positiva per tutti soggetti coinvolti.

Keywords: Alternanza Scuola Lavoro, Impresa Simulata, Didattica per progetti e problemi, Pensiero Computazionale

1 Introduzione

L'introduzione dell'obbligo di alternanza scuola lavoro (ASL) introdotto dalla legge 107/15 (più nota come "Legge sulla buona scuola") ha indotto ogni istituto scolastico ad operarsi per implementarlo. Solitamente tale obbligo è messo in pratica scegliendo tra due modalità: *stage aziendale* oppure *impresa formativa simulata*.

Lo stage aziendale (o *internship* usando il termine inglese) continua la tradizione consolidata negli istituti tecnici già prima della riforma. I ragazzi sono inviati ad una azienda per diverse settimane e direttamente integrati nel ciclo produttivo di questa. È l'opzione probabilmente più in linea con lo spirito della legge, anche se introduce complessità organizzative per gli istituti. Occorre infatti contattare e coordinare un notevole numero di aziende, poiché ciascuna può ospitare un limitato numero di studenti; la varietà di aziende spesso comporta esperienze estremamente difformi per attività, formazione aziendale, coinvolgimento dei ragazzi e risultati formativi; inoltre, la valutazione dell'esperienza è difficoltosa da integrare nel quadro della valutazione scolastica complessiva. Gli stage, peraltro, devono avvenire nello stesso periodo per l'intero istituto, per evitare difficoltà all'attività didattica regolare. Conseguentemente, molte scuole decidono di effettuare tutta o parte della ASL durante il periodo estivo, seppur questo non è sempre accolto con favore da parte di studenti e famiglie.

La simulazione d'impresa prevede invece la costituzione di una vera e propria azienda costituita da studenti (sotto la supervisione di una azienda-tutor) per

la realizzazione di prodotti e servizi. La forma consigliata [1] prevede tuttavia una lunga serie di impegni formali (quali, ad esempio, la richiesta di una vera partita IVA, iscrizione ad un registro nazionale,...) che scoraggia i docenti. Nella maggioranza dei casi, quindi, le classi simulano l'impresa tramite progetti di gruppo, solo occasionalmente coinvolgendo committenti esterni, sacrificando l'aspetto formativo. La parte di formazione è spesso fornita in modo generale, come ad esempio conferenze sulla cultura d'impresa o sul diritto del lavoro, che risultano di limitato impatto sugli allievi.

La nostra proposta intraprende una strada diversa. L'intera classe partecipa ad un *unico* periodo di stage che si svolge fisicamente all'interno dell'istituto scolastico ma prevede la realizzazione di un progetto per conto di un'azienda reale.

Dal punto di vista dell'azienda, l'attività è assimilabile a una forma di *outsourcing*. La formazione specifica è stata ripartita tra gli insegnanti curricolari — modificando il curriculum in base alle necessità del progetto — e i rappresentanti dell'azienda con attività seminariale. La parte operativa, assimilabile in gran parte al Problem-Based Learning [6], si svolge nell'arco di un'intera settimana, denominata **Developers' Week** (DW). In tale periodo circoscritto i ragazzi sviluppano il software per conto dell'azienda mentre i docenti curricolari si limitano alla sorveglianza. In questo modo i docenti possono seguire i progressi dei ragazzi giorno dopo giorno, mentre l'azienda mantiene la direzione generale del lavoro e valuta il prodotto finale. Si tratta quindi di una forma di **ASL rovesciata** (ASLR) (che possiamo chiamare **stage renversé** o **externship**), in quanto è l'azienda ad entrare fisicamente nella scuola piuttosto che il contrario.

Il principale obiettivo dell'attività, oltre all'adempimento degli obblighi di legge, è quello porre i ragazzi di fronte alla complessità, ai ritmi e alle esigenze dello sviluppo software contemporaneo. I ragazzi sono costretti a confrontarsi con problemi complessi, affrontando tematiche che solo parzialmente sono state sviluppate in classe, scontrandosi con i problemi giornalieri e risolvendoli tramite un efficiente *teamwork*.

Al termine del periodo, l'azienda ottiene un prodotto potenzialmente utilizzabile, la scuola una valutazione reale e concreta e, soprattutto, gli allievi effettuano una esperienza coinvolgente e comparabile con una reale attività di sviluppo.

2 Idea e Preparazione

2.1 Obiettivi

Gli obiettivi che si intendevano raggiungere da questa attività erano i seguenti:

1. Realizzare un prodotto software secondo le modalità e gli standard indicati dall'azienda, quindi potenzialmente utilizzabili.
2. Fornire una esperienza quasi-lavorativa in situazioni di stress.
3. Responsabilizzare gli studenti, stimolando la loro auto-organizzazione e la loro capacità di auto-apprendimento.

4. Valutare non solo le prestazioni dei singoli e dei gruppi, ma anche degli aspetti sociali e psicologici di tale esperienza.
5. Ottenere una valutazione effettiva ed utilizzabile in ambito scolastico.

L'esperienza di sviluppo è tanto più realistica quanto più si avvicina alle correnti prassi di mercato; nel nostro caso, si è deciso di applicare la metodologia di sviluppo agile, specificamente Scrum. La scelta è stata influenzata da diversi fattori: in primo luogo, si è ritenuto che una metodologia più tradizionale, con una lunga fase di pianificazione, elevata documentazione e fase di test al termine avesse una minore probabilità di successo; inoltre, l'utilizzo dei metodi Agile è in aumento anche in Italia, specie nelle PMI e, in particolare, presso la GN Networking; infine, Scrum è ben noto al docente che lo ha utilizzato in diverse altre occasioni [7–9, 11].

La classe scelta per la sperimentazione è una 4° di indirizzo Informatico stimata di media abilità e prestazioni. La scelta ha tuttavia implicato in problema non piccolo: quasi tutti gli aspetti relativi alle applicazioni di rete sono affrontati nell'anno successivo ed erano assolutamente necessari per la realizzazione del prodotto richiesto dall'azienda. Ciò ha reso il progetto estremamente difficile per i partecipanti e messo a rischio il risultato finale, specie dopo aver stabilito che la durata complessiva del progetto doveva essere limitata a una settimana — di fatto, questo vincolo ha reso il problema da risolvere assai simile a un *wicked problem* [4]. Stante l'elevato livello di difficoltà e complessità, l'aspetto della sicurezza (autenticazione, programmazione sicura...) è stato interamente tralasciato. Si è concordato con l'azienda che, tra tutti gli obiettivi prefissati, il primo – cioè la realizzazione effettiva del prodotto – era quello meno sacrificabile.

Un altro punto cruciale è stato quello della valutazione. Si è deciso di effettuare **tre** tipi di valutazione: una valutazione in itinere, giorno per giorno, basata sull'evoluzione del processo (responsabilità del docente); una valutazione conclusiva del prodotto, basata sull'aderenza alle specifiche e alla qualità del codice realizzato (responsabilità dell'azienda); infine, una valutazione sulla performance del singolo studente all'interno del gruppo (responsabilità condivisa tra docenti e studenti).

2.2 Il progetto

Il progetto da sottoporre agli studenti è un classico esempio di *web application* distribuita. Abbiamo scelto come argomento la fatturazione e l'anagrafica interna di una ditta generica - il dominio non era infatti uno dei punti fondamentali dell'attività, e non doveva distogliere gli studenti da argomenti più tecnici.

Come mostrato dalla Fig. 1, il progetto è costituito da tre sottoprogetti interdipendenti: gestione anagrafica e persone (P1), gestione fatture (P2), gestione registrazioni entrate/uscite (P3). Tutti i progetti condividono l'architettura di fondo:

– Interfaccia utente realizzata tramite HTML/CSS/Javascript;

– Backend realizzato in Java, che gestisce la base dati sia per l'applicazione di competenza, la rende disponibile via rete e la integra con quelle delle altre applicazioni, tramite interfacce REST
– Ogni sottoprogetto può accedere in lettura a tutti i database;
– Ogni sottoprogetto può modificare soltanto il proprio database.

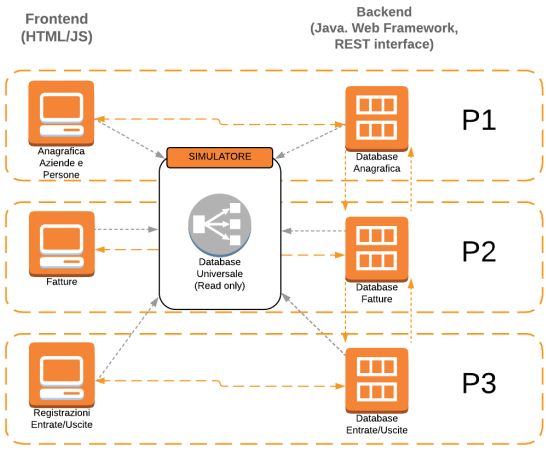

Figura 1. Schema ad alto livello del progetto

Come si evince dalla figura, l'accesso in lettura ad altri database era necessario per la realizzazione di ciascun sottoprogetto; questo avrebbe comportato il completamento degli stessi, cosa rischiosa e realizzabile solo a fine settimana. Per ovviare al problema, abbiamo realizzato un semplice generatore di dati, accessibile qui: `https://dwsim-194816.appspot.com/`. Grazie ad esso, ogni gruppo avrebbe potuto proseguire nello sviluppo senza dipendere dallo stato di avanzamento di altri progetti. In fase di integrazione, il simulatore sarebbe stato rimosso.

2.3 La formazione curricolare

Il principale ostacolo alla realizzazione del progetto è data dal curriculum; infatti, una classe quarta informatica (a metà anno) è assolutamente priva delle competenze necessarie per realizzarlo. Attorno a gennaio-febbraio una classe standard ha affrontato le prime problematiche della programmazione OOP (nel nostro caso in Java), una certa conoscenza di HTML/CSS/Javascript, e le basi delle rete (IP, routing, ecc..). Non hanno esperienza di programmazione di rete, di server web, server side programming, framework, AJAX... tutti argomenti trattati l'anno successivo. Questa carenza di conoscenze ed esperienza ha trasformato un'attività di media difficoltà per un professionista del ramo in un compito di elevatissima difficoltà. In termini didattici, si pone largamente al di fuori della zona prossimale

di sviluppo di Vygotskij; dal loro punto di vista, assume piuttosto i contorni di un problema *wicked*: "un problema difficile o impossibile da risolvere per la presenza di richieste incomplete, contraddittori, mutevoli e difficili da riconoscere" [4].

Non potendo (né volendo) rivoluzionare la programmazione di due anni abbiamo concordato di fornire agli studenti una semplice infarinatura degli argomenti di quinta assolutamente necessari. Il "prezzo" da pagare è un ridotto approfondimento di taluni argomenti (come ad esempio sull'ereditarietà, la serializzazione, la programmazione parallela).

Più in dettaglio, si è deciso di applicare le seguenti modifiche:

- **Informatica**: Anticipo e approfondimento della documentazione automatica (Javadoc), Eccezioni, Unit testing (Junit); introduzione del Pair Programming. Alleggerimento ereditarietà, posticipo Swing e Android;
- **T.EP.S.I.T**: Uso di RCS tramite Git e Github. Presentazione del Test Driven Development e accenno ai metodi Agile. Posticipo della programmazione parallela;
- **Sistemi e Reti**: Anticipo del protocollo HTTP, metodi GET e PUT.

Complessivamente questi cambi hanno influenzato circa il 20% del monte ore del periodo settembre-gennaio e non hanno comportato particolari problemi didattici.

2.4 La formazione aziendale

Due settimane prima della settimana scelta per l'esperienza, il titolare dell'impresa ha svolto un intervento didattico-motivazionale all'interno della classe. Ha presentato la modalità di sviluppo utilizzata in azienda (Scrum, appunto), e ha delineato alcune differenze tra le modalità di collaborazione a scuola e in azienda. Per evitare che le informazioni mostrate rimanessero vuote nozioni e come tali rapidamente dimenticate, si è pensato di proporre agli allievi due attività laboratoriali di tipo ludico. La pratica dei *serious games* è una tradizione consolidata nel mondo Agile (si vedano ad esempio siti quali *Tasty Cupcakes*); permette infatti permette di concentrarsi sul *processo* di sviluppo anziché sul *prodotto* e di rimuovere impedimenti dovuti alle differenze di abilità tra i vari partecipanti. Le attività presentate sono state:

1. Build a party (1 ora): gli studenti dovevano pianificare e stimare i passi necessari per l'organizzazione di una festa.
2. Lego Scrum City Simulation (3 ore): attività molto coinvolgente che consiste nel realizzare una città utilizzando mattoncini Lego® [2].

Entrambe le attività sono state molto apprezzate dai ragazzi.

2.5 La formazione specifica

Come anticipato precedentemente, poche settimane prima dell'attività sono presentate brevi lezioni sui seguenti argomenti:

– Framework di rete (3 ore). Si è scelto di utilizzare un framework semplice realizzato in Java che fornisse le funzionalità di web server e REST. La scelta è caduta sul framework Java Pippo, ritenuto tra i più semplici.
– JSON (2 ore). Struttura del file e modalità di conversione nei vari linguaggi.

Si noti che le problematiche legate ad AJAX e ai database non sono stati affrontati in alcun modo: in questo modo si è voluto stimolare la capacità degli studenti nel risolvere richieste complesse e "quasi" impossibili.

3 La settimana degli sviluppatori

3.1 Logistica

Il problema principale è stato quello di disporre di laboratori di informatica per l'intero periodo, cosa che ha comportato il ripensamento dell'orario scolastico. Ciò ha causato qualche disagio e conseguente lamentela da parte di studenti e genitori di altre classi (nonché la segreteria).

3.2 Prima giornata: progettazione

La prima giornata è stata articolata su due fasi: presentazione del progetto e analisi. Il titolare della ditta ha ricoperto il ruolo di cliente e spiegato per linee generali il progetto. È seguita una discussione tramite la quale gli studenti hanno tentato di capire la struttura del progetto e le richieste che avrebbero dovuto soddisfare. Sono stati quindi divisi in sei gruppi potenzialmente equipollenti; ad ogni coppia di gruppi è stato assegnato un sottoprogetto. In questo modo si è anche ridotto il carico di lavoro per i docenti e introdotto un blando elemento di competizione.

Ad ogni gruppo è stata fornita una presentazione generale del progetto e un elenco delle specifiche del protocollo REST usato dal simulatore cui occorreva conformarsi. Subito dopo ogni gruppo ha cominciato a discutere, a creare schemi, scrivere le specifiche sotto forma di user story stimate e preparare l'ambiente di lavoro (Github, Trello, Dropbox, chat su Slack e altro ancora).

Osservazioni: Dopo un breve brainstorming, la maggior parte dei gruppi ha correttamente scomposto le cose da fare in user story, seppur molto generiche. Tutti hanno tuttavia fortemente sottostimato l'aspetto di comunicazione tra le due componenti software (server Java e client JS). Il docente ha comunque valutato positivamente questa fase del processo,

3.3 Giornate 2-3-4-5: Gli sprint

Le giornate successive hanno avuto la struttura classica di uno sprint Scrum: Standup meeting, Sprint preparation, Coding, Review e Retrospect. Per spronarli a seguire la metodologia utilizzata dalla ditta, è stata fornita una *rubric* che valuta i singoli aspetti dello sprint, nonché la redazione di un documento di narrazione [3] aggiornato giornalmente che descrivesse le attività svolte. Il sistema si è rivelato

efficace, seppure mal digerito da qualche studente che lo ha trovato eccessivamente intrusivo.

Osservazioni: Durante il primo sprint il codice realizzato non è stato molto, poiché lo sforzo era concentrato nell'organizzazione e nello sviluppo dellHTML/CSS, considerato più facile. Il secondo giorno i team si sono scontrati con le difficoltà d'uso del framework, mentre il terzo tutti si sono resi conto della difficoltà dell'interazione tra front e backend, generandomolta tensione; per recuperare, qualche gruppo ha lavorato anche da casa, persino a notte inoltrata. La situazione è proseguita il giorno successivo, dato che la deadline si avvicinava sempre di più.

3.4 Presentazione e conclusione

La giornata di sabato è stata dedicata alla presentazione del prodotto. Ad ogni gruppo sono stati assegnati 10 minuti, durante i quali sia il docente sia il titolare della ditta hanno effettuato domande relative sia all'ambito metodologico sia quello quello dello sviluppo. Terminata questa fase, si è concordata una "classifica" conclusiva dei vari gruppi. A seguire, una breve cerimonia di premiazione con buffet annesso. Agli studenti è stato quindi chiesto di realizzare una articolata riflessione sul lavoro compiuto; tale elaborato è stato consegnato anche al docente di Italiano che ha provveduto a correggerlo e a fornire un commento dettagliato.

3.5 Valutazione

L'attività ha messo a disposizione dei docenti numerosi dati, che vorremmo ora descrivere più in dettaglio.

Come accennato, la **valutazione del processo** di sviluppo è stata demandata ai docenti, che rilevavano giornalmente lo stato di avanzamento. Tutto questo ha prodotto una misurazione dinamica dell'andamento di ogni gruppo, utilizzato dall'insegnante anche per fornire consigli organizzativi. I risultati sono stati ottenuti tramite rubric nota ai ragazzi(Sez. 5) e riportati in Tab. 1, colonna *Processo*. Aldilà del valore numerico, i docenti (incluso i colleghi che prestavano assistenza) sono rimasti colpiti dall'impegno profuso dai ragazzi durante l'attività.

Tabella 1. Valutazione lavoro degli studenti

		Processo (in itinere)						Prodotto	Opinioni	
Progetto	Gruppo	Analisi	S1	S2	S3	S4	Media	Classifica	Teamwork	Gradimento
P1	Beta	5	6	5,5	7,5	6,5	6,1	6°	6	7,5
P1	Epsilon	9	9,2	7	8	7,5	8,14	2°	9	9,5
P2	Delta	9	9,2	7,2	7,6	8	8,2	2° (-)	7,5	9
P2	Gamma	7,5	8,5	7,2	7,4	8	7,72	2° (+)	8,5	9,5
P3	Alpha	9	7,5	6	6,5	6	7	5°	7,5	7,3
P3	Zeta	9,5	8,8	6,5	9,5	8	8,46	1°	9	9,5

La **valutazione del prodotto** è stata affidata all'azienda, che ha tenuto conto della presentazione finale, delle funzionalità effettivamente realizzate e dell'analisi del codice (strutturazione, qualità, commenti) e la presentazione. I risultati sono sintetizzati nella colonna *Prodotto*. Il gruppo Zeta, risultato vincitore, si è distinto per una ottima presentazione e il numero di feature realizzate. I tre gruppi successivi (Gamme, Epsilon, Delta) hanno ottenuto risultati tutto sommato buoni, differenziandosi leggermente per il numero di funzionalità realizzate e la qualità della presentazione, per cui possono essere raggruppati in un'unica categoria. Gli ultimi due gruppi hanno realizzato entrambi poche feature, ma un gruppo ha pagato una presentazione di bassa qualità, rendendo il prodotto di livello "inaccettabile".

La **valutazione personale** deriva da una autovalutazione del lavoro di gruppo (colonna *Teamwork*), dall'esame delle riflessioni personali, del documento di narrazione e un colloquio personalizzato. Si noti la grande correlazione tra i vari tipi di valutazione.

Le tre valutazioni (processo, prodotto, personale) sono state quindi combinate e hanno permesso di esprimere un voto complessivo individuale per Informatica e Italiano. Per completezza, mostriamo anche il risultato del sondaggio somministrato "a caldo" (Tab.2). Come si vede, le impressioni sono generalmente molto positive, con qualche perplessità sull'uso della demo come strumento di valutazione.

Tabella 2. Sondaggio "a caldo", (Likert scale, 1=per nulla, 5= moltissimo)

Domanda	**Risposta**
La valutazione giorno per giorno è stata utile?	4,09
La valutazione finale (prodotto funzionante) tramite demo è stata equa?	3,5
Questa esperienza ti ha stimolato la voglia di approfondire aspetti tecnici?	3,95
Ritieni utile ricevere una preparazione sul mondo del lavoro già a scuola?	4,69
Pensi che questa iniziativa ti prepari per la vita lavorativa?	4,18

Un esame più dettagliato degli elaborati consegnati al docente di Italiano ha permesso di estrapolare le seguenti **opinioni** comuni:

- l'esigenza di una maggiore spiegazione su aspetti tecnici necessari per la DW, unita alla certezza di poter riuscire a risolvere molti problemi con poco tempo in più;
- un maggiore attenzione alla formazione dei gruppi, perché alcuni avevano capacità limitate e altri non hanno saputo rapportarsi correttamente col gruppo;
- un giudizio positivo sull'esperienza e la consapevolezza dei loro limiti, della necessità di superarli con un impegno maggiore (per la maggior parte di loro) e dell'abisso esistente tra scuola ed esigenze del mondo del lavoro (tra cui la necessità di lavorare con persone con cui non si va d'accordo);
- l'acquisizione di una maggiore consapevolezza delle loro possibilità e delle loro potenzialità;

– il desiderio di rifare un'esperienza simile.

4 Riflessioni

L'attività offre diversi spunti di discussione. Si noti, intanto, che le valutazioni di processo e quelle di prodotto sono strettamente correlate — una sorta di conferma empirica della legge di Conway [5]. La nostra interpretazione è che la fase di analisi e di auto-organizzazione di gruppo ha avuto un peso fondamentale nell'ottenere risultati apprezzabili, confermando la dimensione sociale nella programmazione riveste un ruolo sempre più importante. Purtroppo, la didattica nelle superiori non è in genere strutturata per promuovere lo sviluppo delle *soft skills*. Pensiamo che attività di questo tipo possano risultare particolarmente utili per stimolare le abilità sociali tra i programmatori; in particolare, le metodologie Agile sembrano un'opzione efficace per proseguire lungo questa strada.

Il progetto è risultato impegnativo per tutte le persone coinvolte. Gli studenti hanno avuto un coinvolgimento emotivo elevato, ma anche fisico. L'azienda ha investito una certa quantità di tempo nella valutazione, nella consulenza e nella presenza fisica. La scuola ha dovuto riorganizzare l'orario. I docenti hanno seguito il progetto con continuità, producendo un continuo feedback per gli studenti. Intraprendere questa forma di ASL richiede quindi impegno e pianificazione, che tuttavia paiono ben ripagati dai risultati.

5 Conclusioni

In sintesi, i risultati di questa nuova modalità di ASL sono stati estremamente positivi e, per certi versi, sorprendenti. La motivazione dei ragazzi ne ha certamente tratto grande giovamento, come risulta dal feedback fornito. Tale risultato era auspicabile e prevedibile, ma l'intensità è stata superiore alle attese.

Nonostante evidenti difficoltà, quasi tutti i gruppi sono riusciti ad orientarsi e produrre risultati accettabili e in ogni caso migliori di quanto previsto dall'azienda. È una conferma che le potenzialità di un gruppo sono superiori alla somma dei valori individuali, come precedentemente sospettato [10,12].

La DW è molto flessibile, per durata, complessità, metodologia,valutazione e tipo di prodotto; può quindi essere tarata sulle richieste dell'azienda, sulle caratteristiche dell'istituto e di ogni singola classe; si può quindi applicare ad una grande varietà di situazioni (incluso i licei LSSA e gli ITES). A nostro parere tuttavia, tre sono i punti fondanti e irrinunciabili di questa attività:

1. La sospensione dell'attività didattica "regolare" in modo continuativo per la durata prevista - senza eccezioni.
2. L'individuazione di un partner esterno alla scuola capace e affidabile. È essenziale che gli studenti si rapportino *direttamente* ad una entità non scolastica, in modo da percepirne le differenze di richieste, impostazioni, relazioni.

3. Il lavoro deve essere realizzato come gruppo, pur differenziando le prestazioni individuali.

Altri aspetti possono invece essere riconsiderati alla luce dell'esperienza. Ci chiediamo quindi se l'aspetto competitivo, seppur limitato, sia un elemento positivo o negativo. Allo stesso modo se la scelta di lavorare su un compito "impossibile" sia di fonte stimolo o di depressione. Verificheremo questo l'anno venturo, quando ripeteremo l'esperienza di *ASL rovesciata*.

I colleghi interessati possono consultare la pianificazione dell'attività e le rubric utilizzate per la valutazione a questo indirizzo: `https://goo.gl/qH6aF7`. Lo schema propone un peso pressoché equivalente tra valutazione di processo e prodotto. Inoltre, tutti i progetti (con relativi codici) realizzati dai ragazzi sono accessibili su Github all'indirizzo `https://github.com/4binfo18/`.

Riferimenti bibliografici

1. Impresa formativa simulata, dettagli. `http://www.cnos-fap.it/sites/default/files/rapporti/impresa_formativa_simulata_09_ottobre_2016.pdf`
2. Lego city scrum simulation. `https://infoagilecoop.blogspot.com/2018/01/lego-scrum-city-tutorial.html`
3. Lepida scuola, documento di narrazione. `http://www.lepidascuola.org/il-metodo/documento-di-narrazione/`
4. Buchanan, R.: Wicked problems in design thinking. Design issues 8(2), 5–21 (1992)
5. Conway, M.E.: How do committees invent. Datamation 14(4), 28–31 (1968)
6. Hung, W., Jonassen, D.H., Liu, R., et al.: Problem-based learning. Handbook of research on educational communications and technology 3, 485–506 (2008)
7. Missiroli, M., Russo, D., Ciancarini, P.: Learning agile software development in high school: an investigation. In: Proc. 38th Int. Conf. on Software Engineering (ICSE). pp. 293–302 (2016)
8. Missiroli, M., Russo, D., Ciancarini, P.: Una didattica agile per la programmazione. Mondo Digitale 15(64) (2016)
9. Missiroli, M., Russo, D., Ciancarini, P.: Agile for Millennials: a comparative study. In: Proc. 1st Int. Workshop on Software Engineering Curricula for Millennials. pp. 47–53. IEEE Press (2017)
10. Missiroli, M., Russo, D., Ciancarini, P.: Cooperative Thinking, or: Computational Thinking Meets Agile. In: Software Engineering Education and Training (CSEE&T). ACM (2017)
11. Missiroli, M., Russo, D., Ciancarini, P.: Teaching test-first programming: assessment and solutions. In: Proceedings of the Computer Software and Applications Conference. IEEE (2017)
12. Russo, D., Missiroli, M., Ciancarini, P.: Poster: a Conceptual Model for Cooperative Thinking. In: Proc. 40th Int. Conf. on Software Engineering (ICSE). ACM (2018)