Programma TFA Informatica di Base A.A. 2012-13

*Federico Gobbo (*Disim*)*

31 Maggio 2013

1. Che cos'è il software?

La natura del software è un aspetto dell'informatica che viene raramente analizzato in tutti i suoi aspetti. Il software infatti ha almeno due livelli, il codice sorgente (leggibile dall'uomo) e il codice binario (leggibile dalla macchina). Ma questo primo livello di analisi lascia aperte una serie di domande importanti: quanto è importante l'implementazione? Perché usiamo diversi linguaggi di programmazione se in teoria sono tutti Turing-equivalenti? Che differenza c'è tra istruzioni, esecuzione, e dati? Vedremo il caso particolare della meta-programmazione, e il ruolo del programmatore come (meta-)autore del software.

Materiale fornito: verrà consegnata bibliografia aggiornata e ragionata a tutti i partecipanti, sulla base degli ultimi risultati di ricerca in *Computing & Philosophy*.

Esercizi in classe: sulla base dell'esperienza dei software usati e/o programmati dai docenti, vedremo di applicare il modello teorico presentato ai casi concreti. Seguirà discussione.

2. Modelli di produzione del software

Attorno al software c'è tutto un ecosistema, formato da diverse figure, professionali e non solo: il designer, lo sviluppatore, il committente, l'utente finale, ecc. A seconda della licenza scelta (proprietaria, a sorgente aperto, software libero) si configurano diversi modelli di produzione, con risvolti diversi anche da un punto di vista economico. Vedremo il modello di produzione industriale del software proprietario di tipo tayloristico, il modello Toyota (dall'*eXtreme Programming*), il modello a bazar di Raymond, la strategia della doppia licenza, la legge della coda lunga di Anderson e altri modelli noti in letteratura.

Materiale fornito: verrà consegnata bibliografia aggiornata e ragionata a tutti i partecipanti, sulla base degli ultimi risultati di ricerca in *Agile methods* ed *eXtreme Programming*.

Esercizi in classe: sulla base dei software già analizzati nella sessione precedente. Seguirà discussione conclusiva.

3. Analisi di casi etici in informatica

La pervasività dell'informatica nella società comporta una serie di dilemmi etici di difficile soluzione, che possono essere analizzati tramite il metodo dell'analisi dei casi etici. Dopo aver spiegato il metodo in tutti i suoi passaggi, verranno proposti alcuni casi etici noti in letteratura, quali: chi è responsabile del drone che in guerra uccide erroneamente un civile? È giusto mettere videocamere di sorveglianza ovunque o potrebbe essere usato per fini poco leciti e quindi andrebbe limitato?

Materiale fornito: verrà consegnata bibliografia aggiornata e ragionata, in inglese e in italiano a tutti i partecipanti, a fine lezione, sulla base degli ultimi risultati di ricerca in *Agile methods* ed *eXtreme Programming*.

Esercizi in classe: analisi collettiva guidata e discussione dei casi presentati in classe.

Date delle lezioni previste

- **10 Maggio 2013**: Che cos'è il software?
- **17 Maggio 2013**: Modelli di produzione del software
- **31 Maggio 2013**: Analisi di casi etici in informatica

1. Che cos'è il software?

La storia dell'informazione può essere divisa in due parti: quella pre-informatica, che va dalla matematica greca (geometria)[1] e babilonese (algebra)[2], poi unite dalla rivoluzione scientifica seicentesca, che per realizzare il sogno Leibniziano del *calculus ratiocinator* e della *characteristica universalis* arriverà alla fondazione teorica dell'informatica, con risultati di Church, Turing e Post nel 1936.[3]

Ma è solo con l'architettura di Von Neumann, descritta nel 1949, che possiamo cominciare a parlare di informatica moderna, e conseguentemente della dicotomia 'hardware *vs.* software'.[4] Molte sono le possibili definizioni del concetto di software; partiamo da una possibile definizione *esterna*: il software è l'interfaccia uomo-macchina prima e ultima per compiere la computazione, intesa come manipolazione di numeri Turing-calcolabili aventi significato simbolico (nel senso di *aliquid pro aliquo*).[5]

Abbiamo un agente informativo AgInf che **osserva** il mondo M, **formalizza** l'informazione in un linguaggio L astratto e leggibile dalla macchina ottenendo il dominio D, che inserisce in **input** nella scatola (nera o trasparente), la scatola Box effettua la computazione, estrae in **output** il codominio C, che viene poi **interpretato** retroattivamente (*feedback*) dall'AgInf.[6] L'interpretazione è un confronto pragmatico tra la semantica attesa di C e quella attuale – che può (a volte deve) sorprendere.

All'interno della Filosofia dell'Informazione, abbiamo descritto un organismo informazionale (*inforg*), vale a dire un complesso AgInf + Box, che interconnesso con i sui simili dà vita a un ambiente dinamico e globale detto *infosfera*.[7] L'operazione di *formalizzazione* è il principale livello di astrazione (**LdA**), che consiste nell'inserimento dell'informazione nel corrispondente livello di organizzazione (**LdO**). Questo inserimento può consistere in dati nel caso dell'uso di un applicativo da parte di un utente finale (e avremo la scatola nera), o nel design e sviluppo di algoritmi e strutture dati, nel caso della programmazione vera e propria (e dunque avremo la scatola trasparente). L'operazione di *interpretazione* rappresenta invece il livello di spiegazione (**LdS**), l'orizzonte epistemologico che comprende il *telos* dell'inforg, e che sta al di fuori del sistema.[8]

La storia dell'informatica moderna (dopo il 1949) è un susseguirsi di *esternalizzazione dei LdA in LdO* che vengono via via *nascosti* in un gioco di scatole cinesi: per esempio, in principio gli *operators* che facevano avviare i calcolatori erano esseri umani, poi vengono rimpiazzati dagli *operating systems* (il nome non è casuale!). A questo punto consideriamo come data simbolica da cui partire il 1968, dopo

[1] Leonard Mlodinow. *Euclid's windows: the story of geometry from parallel lines to hyperspace*. Free Press, 2002

[2] John Derbyshire. *Unknown Quantity: A Real and Imaginary History of Algebra*. Plume, 2007

[3] Martin Davis. *Il calcolatore universale*. Adelphi, 2012

[4] John von Neumann. First Draft of a Report on the EDVAC. *IEEE Ann. Hist. Comput.*, 15(4):27–75, October 1993

[5] Federico Gobbo and Marco Benini. From Ancient to Modern Computing: A History of Information Hiding. *IEEE Annals of the History of Computing*, 99(PrePrints), 2013

[6] Normalmente lo consideriamo un essere umano; nel caso sia una macchina, il modello va applicato ricorsivamente fino a trovare l'umano che effettua l'operazione di interpretazione. Se manca l'interpretazione, non abbiamo simboli ma mera analisi numerica.

[7] Luciano Floridi. *La rivoluzione dell'informazione*. Codice edizioni, 2012. Prefazione di Juan Carlos De Martin

[8] Federico Gobbo and Marco Benini. The Minimal Levels of Abstraction in the History of Modern Computing. *Philosophy & Technology*, pages 1–17, 2013

che Backus ha formalizzato la semantica dei linguaggi di programmazione (FORTRAN, poi ALGOL) e Böhm-Jacopini hanno ottenuto la base teorica dei compilatori, che a loro volta hanno permesso l'introduzione di nuovi LdA nella programmazione, non a caso chiamati 'di alto livello'.[9] I linguaggi di programmazione sono la *L* che permette la formalizzazione di *M* in *D* nel nostro schema.

Il caso piú interessante di inforg è quello della programmazione, perché ci permette di vedere il software come il risultato di un vero e proprio processo di creazione, una nuova forma d'arte, possibile solo dagli esseri umani come AGINF. Infatti, c'è bisogno di coscienza dell'ambiente (*a-coscienza*), coscienza di sé (*s-coscienza*) e coscienza fenomenica (*f-coscienza*), e solo gli esseri umani li possiedono tutti e tre.[10]

Anche il caso speciale della metaprogrammazione, che rompe il dogma della programmazione strutturata, secondo cui le esecuzioni sono istanze delle istruzioni sui dati, permettendo che le istruzioni vengano modificate dinamicamente, non modifica il quadro sostanziale.[11] Difatti, è sufficiente considerare *tre livelli* nel software invece che due (sorgente *vs.* binario) per uscire dallo stallo: il primo livello del codice sorgente (parte di *D*, scritto in *L*); il secondo livello del codice binario in esecuzione (la computazione in BOX, nel nostro schema), e infine il livello del codice sorgente automodificato, in cui il codominio *C* modifica *D senza* passare attraverso il processo di interpretazione da parte di AGINF. Perché questo non è un caso di intelligenza artificiale (IA)? Perché tali modifiche non sono s-coscienti né f-coscienti.[12]

[9] Questi risultati hanno permesso l'introduzione di linguaggi ad alto livello e dunque dei principali paradigmi di programmazioni posteriori alla programmazione strutturata, vale a dire: procedurale, funzionale, logica, a oggetti. Questo argomento verrà approfondito in altri corsi TFA di questo ciclo, per cui verrà solo accennato.

[10] Federico Gobbo and Marco Benini. Why zombies can't write significant source code: The Knowledge Game and the Art of Computer Programming. *Journal of Experimental & Theoretical Artificial Intelligence*, 2013. forthcoming

[11] La metaprogrammazione è possibile tra l'altro in Cobol, LISP, Prolog e Ruby.

[12] Gli agenti autonomi robotici, per esempio basati sugli animali, sono a-coscienti, possono adattarsi all'ambiente modificando le proprie istruzioni e quindi apprendere automaticamente. Questa capacità viene considerata la versione debole dell'IA, introdotta da Searle nell'esperimento mentale della Stanza Cinese nell'ormai lontano 1980.

Esercizio in classe Presentare in aula il modello proposto di inforg, specificando LdA, LdO e LdS, a un caso studio tratto dalla propria esperienza didattica, che verrà poi discusso collettivamente. Il caso studio presentato rappresenta la base (circa 13.000 battute sul totale di 20.000 previste) dell'approfondimento che vale come prova scritta d'esame.

2. I modelli di produzione

Abbiamo visto che il software è una serie di LdA e di LdO paralleli, di cui rivestono importanza fondamentale il codice sorgente e il codice binario. Per capire i modelli di produzione del software bisogna capire come viene trattato a livello legislativo. Le forme di protezione del software sono tre: copyright (diritto angloamericano) e diritto d'autore (diritto europeo continentale); brevetto; design – generalmente omesso, vediamo sotto perché.[13]

La prima linea di protezione consiste nel **diritto d'autore**, secondo cui il software è un'opera letteraria a carattere scientifico, e deve dunque avere carattere di creatività e originalità. Si tratta della forma prevalente di protezione che però non è esente da problemi.[14] Poiché le funzionalità di un software descritte dal suo algoritmo in quanto tali non possono essere messe sotto copyright, l'operazione di *reverse engineering*, che permette di ottenere da un codice binario eseguibile un codice sorgente diverso dall'originale, non viola il diritto d'autore. x Una seconda linea di protezione, generalmente piú contestata della prima, è la tutela mediante **brevetto**. Si noti che non è possibile brevettare un software in sé, ma solo in quanto realizzazione tecnica: il software viene considerato un *ausilio* all'attività inventiva, che deve prevedere un'applicazione industriale.[15]

La terza linea di protezione, quella del **design**, è meno evidente e maggiormente problematica. Può essere protetto il *look & feel* di un software, per esempio la sua interfaccia utente o la funzione di taglia-e-incolla? I colossi dell'industria informatica ingaggiano battaglie legali enormi su questi aspetti: noto è l'attacco di Apple a Samsung per supposta violazione del brevetto del design dell'iPad da parte del tablet Galaxy, in cui Samsung si è difesa citando una scena del film *2001: Odissea nello spazio* di Kubrick dove compaiono due dispositivi molto simili a un tablet. Al di là della specifica vicenda, è evidente la problematicità di questa linea di protezione.[16]

In un contesto didattico di creazione di software – quale una scuola secondaria superiore – normalmente non ci sono possibilità di applicazione né della linea del brevetto (manca la parte industriale) né di quella del design, pertanto rimane la scelta diventa obbligata, nella forma della *licenza* da applicare al software come oggetto coperto da diritto d'autore. Se non viene specificato nulla, il software è proprietà del suo autore, cioè di chi scrive il codice, e viene indicato con il simbolo ©, insieme al nome dell'autore e all'anno di rilascio, vale a dire di *pubblicazione* del software – la messa in rete su un sito web pubblico è una pubblicazione a tutti gli effetti. Naturalmente, se il codice è scritto a piú mani (per esempio da un gruppo di lavoro

[13] Giustino Fumagalli. *La tutela del software nell'Unione Europea. Brevetto e diritto d'autore.* Nyberg, 2 edition, 2005

[14] Tralasciamo le differenze giurisprudenziali tra le due filosofie del diritto perché non rilevanti per i nostri scopi di didattica dell'informatica.

[15] Reinier B. Bakels. Software patentability: what are the right questions? *European Intellectual Property Review*, 10:514–522, 2009

[16] In questo caso le differenze tra il diritto statunitense e quello europeo complicano ulteriormente il quadro.

nell'ambito di una classe) la scelta del copyright 'classico' potrebbe non essere la piú opportuna. Nel momento in cui viene scaricato un software, se ne effettua una copia, e quindi a rigore si sta violando il copyright commettendo un atto illegale – il principio retrostante è quello del libro fotocopiato per intero. Per questo motivo, a partire dalla prima generazione di hacking della fine degli anni 1960 ci si è posti il problema della distribuzione del software e in particolare del codice sorgente, e si sono cercate delle soluzioni.[17] I fondatori del C, di Unix e gli altri membri del MIT AI Lab negli anni 1970 erano usi scambiarsi copie dei sorgenti liberamente, ma con l'emergere dell'industria e della professione i *computer boys* dovettero crescere ed entrare nel sistema capitalistico – ciò che fecero quasi tutti, compresi Steve Jobs e Bill Gates, i co-fondatori rispettivamente di Apple e Microsoft, hacker di prima generazione che 'tradirono' la causa.[18]

L'alternativa era fare una scelta radicale, antisistema, che fu perseguita da Richard Stallman, colpito dal fatto che Unix e il C venisse 'rubato' dalle aziende che lo modificarono, chiusero l'accesso al sorgente e lo fecero proprio: il software proprietario e il copyright andava abbattuto, propugnando il suo opposto, vale a dire il software *libero* e il *copyleft*.[19] Nel cruciale biennio 1983-4[20] Stallman lancia il progetto GNU e fonda la Free Software Foundation (FSF), con l'intento di rifare un sistema operativo alla Unix con una licenza, la GPL, che garantisse le libertà del programmatore e che impedisse alle aziende di appropriarsi del software. Il programmatore è un po' artista, un po' operaio: il frutto del suo lavoro lo deve godere lui. Naturalmente, poiché il software libero è a sorgente aperto ed è incompatibile con il sofware proprietario ("chiuso") difficilmente il software può essere venduto dal programmatore:[21] l'idea di Stallman è quella di *regalare* il codice, e di *vendere l'assistenza*; non esisteva ancora l'utente finale, tutti gli utilizzatori erano programmatori o sistemisti, quindi disposti a spendere del tempo per imparare (senza assistenza) o a investire del denaro per risparmiare tempo. L'informatica non era un fenomeno di massa, ma di un'élite piuttosto ristretta.[22] Dopo dieci anni al progetto GNU mancava un pezzo fondamentale per rilasciare il sistema operativo: il kernel. E un giovane sconosciuto finlandese scrisse un kernel monolitico che funzionava sui processori *x*86 dei PC, sfidando la convenzione accademica rappresentata da Tanenbaum – che però funzionava. Quel giovane si chiamava Linus Torvalds e, pur rilasciando in licenza GPL il suo sistema operativo, Linux, con un'appendice permetteva a chiunque di abbinare del software indipendentemente dal fatto di essere libero o proprietario, scombinando i piani di Stallman e aprendo l'informatica a una nuova era.

Il movimento *open source* nasce il 3 febbraio 1998 a Palo Alto, in California, da parte di Eric Raymond, hacker della generazione di

[17] Steven Levy. *Hackers. Gli eroi della rivoluzione informatica*. Shake edizioni, 2002

[18] Nathan Ensmenger. *The computer boys take over: computers, programmers, and the politics of technical expertise*. The MIT Press, 2010

[19] Richard Stallman. *Software libero, pensiero libero*. Nuovi equilibri, 2003

[20] Fu un momento cruciale nella storia dell'informatica: finisce la stagione dello *home computing*, esce il primo Macintosh, e si impone l'architettura del PC-IBM.

[21] In realtà Stallman vendette per posta i sorgenti di Emacs, quando la distribuzione del software non era supportata dalla rete.

[22] Un paragone è con il mercato delle automobili: finché bisogna essere meccanici per poter accedere a un'automobile, evidentemente solo in pochi avranno la possibilità di guidare.

Richard Stallman, insoddisfatto dal progetto GNU e folgorato da Linux. L'idea di base è sganciare la portata rivoluzionaria del software libero e di far accettare il software aperto alle aziende.[23] Raymond descrive il software il cui controllo di versione è centralizzato una *cattedrale*, dove i programmatori sono in una sorta di linea di montaggio taylorista, e il codice-cattedrale viene scritto in silenzio e riverenza: idealmente, il rilascio è uno, quello finale.[24] Viceversa, il mondo del software a sorgente aperto iniziato da Linus Torvald è un *bazar*: i programmatori sono artisti di strada, venditori di se stessi, i rilasci sono molti e frequenti. Le leggi del mercato fanno emergere il rilascio migliore, dove si aggrega la comunità degli sviluppatori.[25] Facendo il punto, agli albori del nuovo millennio i modelli di produzione del software sono i seguenti:

[23] Eric Steven Raymond. *La cattedrale e il bazaar*. Apogeo Online, 1998

[24] Si noti che per Raymond sono cattedrali sia il software libero alla Stallman che il software industriale, alla Microsoft o Apple.

[25] Possiamo considerare Linus Torvalds e la sua generazione, quella definita dal software a sorgente aperto, al seconda generazione di hacker.

nome del modello	*personaggi simbolo*	*programmatori*	*produzione del software*	*sorgente*
proprietario	S. Jobs & B. Gates	impiegati	come prodotto	chiuso
libero	R. Stallman	rivoluzionari	come servizio	aperto
aperto	L. Torvalds	artisti	doppia licenza	aperto

Nel *modello proprietario* il software si vende come un pacchetto, chiuso, da prendere o lasciare: non si può modificarlo alle proprie esigenze, cosí come non si modifica una lavatrice o una caffettiera. Non può essere copiato. Nel *modello libero* il software viene venduto come un servizio: io programmatore ti dò il codice sorgente, se vuoi imparare a usarlo da solo sei libero di farlo; se vuoi che ti spiego come installarlo, personalizzarlo, o altro, mi paghi come un consulente. Il *modello aperto* è una via di mezzo, che viene letteralmente inventata da RedHat, una delle piú importanti distribuzioni di Linux in quegli anni: con la licenza *base* ti dò il prodotto e il servizio minimi per usarlo, se vuoi avere tutte le funzionalità complete allora c'è una licenza *premium* con l'abbonamento all'assistenza e tutto quello che ne comporta.

Ma con il nuovo millennio, dopo il crollo del Nasdaq – 10 marzo 2010, la cosiddetta bolla delle *dot com* – tutti e tre i modelli mostrano chiaramente i loro limiti. Il modello proprietario è indebolito dalla *pirateria*, il software viene scaricato illegalmente (cosí tra l'altro chi lo produce non sa nemmeno quanti utenti lo usano), la licenza di sblocco viene *cracked* della licenza, per risparmiarne il costo. Gli altri due modelli si basano sull'*effetto noosfera*: cosí come nell'economia dell'abbondanza, chi riceve un dono è stimolato a ricambiarlo: se provo liberamente il tuo software e decido di usarlo, magari per farne profitto, o faccio una donazione (moneta sonante) oppure lo miglioro in qualche modo.[26] Se questo effetto fallisce, abbiamo il fenomeno dei 'battitori liberi' (*free riders*): utenti che usano il software senza

[26] Un programmatore potrebbe correggere un baco o aggiungere una funzionalità; un non programmatore potrebbe tradurre l'interfaccia in italiano o scrivere la documentazione (spesso lacunosa): non bisogna per forza programmare per contribuire!

fare nulla in cambio, perché tanto 'software libero = gratuito', il che è palesemente falso.

Con il fenomeno della digitalizzazione dell'opera d'ingegno (la musica prima su CD poi su mp3, i film prima su DVD poi in altri formati piú compatti come il DIVX), questo tipo di problemi ha travalicato i confini del software per diventare centrale nell'industria dell'intrattenimento. Chris Anderson, direttore di *Wired*, nell'analizzare il successo di Amazon enuncia la legge della coda lunga.[27] Con il web lo scaffale diventa potenzialmente infinito: non c'è piú bisogno di mettere i best-seller in vetrina, e sacrificare i mercati di nicchia, ma al contrario l'insieme dei mercati di nicchia diventa piú importante dei titoli piú venduti.[28]

Un avvocato americano di nome Lawrence Lessig ha analizzato a fondo la normativa sul copyright e i fenomeni del software libero e aperto, ed è giunto a formulare un copyright modulare, dove i diritti possono essere riservati solo in parte, non tutti (copyright classico) o nessuno (copyleft, dove in sostanza si regala la proprietà alla FSF).[29] Nasce cosí il movimento *Creative Commons*, importante per una serie di ragioni: è internazionale, vale a dire le licenze vengono adattate alle legislazioni nazionali; le licenze sono modulari, cioè si può scegliere cosa riservare e cosa no; le licenze sono scritte in tre modi: in linguaggio corrente, per permettere la scelta degli autori, in legalese, per l'aderenza al diritto e infine in codice, cosí l'opera viene indicizzata subito dai motori di ricerca una volta registrata e dunque pubblicata. Nelle licenze con la doppia c (cc) , l'*attribuzione* all'autore (simbolo: (BY:)) è automatica, mentre gli altri diritti sono modulari. In particolare, questi sono: la (non) commerciabilità[30] del proprio software (simbolo: (=)); la (non) possibilità di fare opere derivate[31] e infine il 'condividi allo stesso modo' (simbolo: (↻)) che recupera la viralità inventata da Stallman con la licenza GPL. Attualmente, in Italia le licenze adattate sono la versione 3.0:

[27] Chris Anderson. *La coda lunga: Da un mercato di massa a una massa di mercati*. Codice Edizioni, 2010

[28] Ciò implica che chiunque, anche alla periferia del mondo, può scrivere e pubblicare un software utile se non c'è nulla che fa *esattamente* quello di cui lui ha bisogno, presupponendo che anche gli altri abbiano lo stesso bisogno, ma non lo sappiano ancora.

[29] Lawrence Lessig. *Cultura libera. Un equilibrio fra anarchia e controllo, contro l'estremismo della proprietà intellettuale*. Apogeo, 2005

[30] Non significa che non si può vendere il software, ma solo che bisogna chiedere preventivamente il permesso al detentore del copyright

[31] Nel caso del software, si tratta dei *fork*; negli altri casi, possono essere i remix di un brano musicale, per esempio.

Nome	*Quali diritti riservati*
Attribuzione	(BY:)
Attribuzione - Non opere derivate	(BY:) (=)
Attribuzione - Non commerciale - Non opere derivate	(BY:) ($) (=)
Attribuzione - Non commerciale	(BY:) ($)
Attribuzione - Non commerciale - Condividi allo stesso modo	(BY:) ($) (↻)
Attribuzione - Condividi allo stesso modo	(BY:) (↻)

Esercizio in classe Applicare i modelli di produzione realmente usati o possibilmente usabili al caso studio di inforg già analizzato oppure a un altro caso in aula, specificando le motivazioni didattiche.

3. L'analisi di casi etici

L'etica dell'informatica, in inglese *Computer Ethics*, viene identificata come un'etica applicata (come l'etica della medicina, per esempio) negli Stati Uniti nella seconda metà degli anni 1980, in particolare con il volume di Deborah Johnson, che ha avuto molta fortuna.[32] In realtà, il vero fondatore della disciplina è Norbert Wiener, il cui lavoro in questo campo fu però ignorato per decenni: la disciplina nasce dunque in ritardo, privilegiando agli aspetti teorici quelli pratici: si cercano soluzioni empiriche a problemi concreti. In pratica, l'etica dell'informatica si occupa dei dilemmi posti dall'uso dell'informatica nella società, e naturalmente è cresciuta moltissimo proprio perché il computer è diventato sempre piú presente in molti àmbiti. In generale, l'etica si relazione da un lato con l'epistemologia (cosa possiamo sapere) dall'altro con la giurisprudenza (che può essere vista come la normazione dell'etica stessa in seno a una società). Per tradizione culturale, l'etica dell'informatica tende a privilegiare il conseguenzialismo rispetto al deontologismo.[33] Questo lo dimostra il fatto che la tradizione giurisprudenziale angloamericana è di tipo induttivo: essa si basa sull'analisi dei casi, e cerca di ricondurre nuove situazioni poste in tribunale ai casi precedenti per *analogia*.

Verso la metà degli anni 1990 il dibattito tra gli studiosi su come formulare un metodo di analisi dei casi etici in informatica ha trovato un punto di equilibrio stabile, in particolare grazie al lavoro di Moor[34], che ha riunito i due principali approcci etici – conseguenzialismo e deontologismo – in una procedura di analisi unificata, che è quella che verrà seguita in questa sede, nella formulazione del manuale di Barger – in particolare il capitolo 5.[35]

[32] Deborah G. Johnson. *Computer Ethics*. Prentice Hall, 1985. 2nd ed 1994; 3rd ed 2001; 4th ed 2009

[33] Il conseguenzialismo è l'approccio analitico derivato dal pragmatismo, orientato all'azione, al calcolo dei costi e benefici e alle *conseguenze* delle azioni. Il deontologismo deriva dalla tradizione continentale, in particolare Kantiana, privilegia l'intenzione e individua norme etiche di valore universale.

[34] James H. Moor. Just consequentialism and computing. *Ethics and Information Technology*, 1:65–9, 1999

[35] Robert N. Barger. *Computer Ethics*. Cambridge University Press, 2008

I passi di Moor

passo zero: prima facie Cosa suscita in noi emotivamente la lettura del caso etico a prima vista, vale a dire senza riflessione razionale? È importante tenerne conto, prenderne nota e metterle da parte, per non influenzare l'analisi.[36]

passo uno: fatti & agenti Abbiamo acquisito tutte le informazioni necessarie? Sono sufficienti? In particolare, bisogna chiedersi: cosa è successo (i fatti)? chi fa fatto cosa (agenti attivi)? chi è coinvolto (agenti passivi)?[37]

passo due: regole morali Quali sono quelle applicabili? In particolare, ci sono dei diritti violati? Questo passo segue l'approccio deontologico.[38]

[36] Questo passo tiene conto delle istanze esistenzialiste, che pongono il soggetto al centro, non solo dal punto di vista razionale, ma anche e soprattutto dal punto di vista emotivo.

[37] L'etica dell'informazione di Floridi in particolare sottolinea l'importanza di questo aspetto, puramente epistemologico, per evitare di avere una visione parziale del caso stesso.

[38] Le regole morali Kantiane sono tre. (1) Regola d'oro o dell'imparzialità: 'agisci come se quello che fai tu possa essere fatto da chiunque altro.' (2) Regola teleologica: 'agisci trattando l'uomo (te stesso e gli altri) come un fine, mai come un mezzo.' (3) Regola del legislatore universale: 'agisci come se ciò che fai sia valido per tutti gli individui.'

passo tre: conseguenzialismo A partire dalle conseguenze possibili, individuiamo tre scenari: lo scenario peggiore, lo scenario migliore, e infine viene ricavato lo scenario ottimale come il miglior "giusto mezzo". Questo passo segue l'approccio conseguenzialista.[39]

[39] Essenzialmente si tratta di calcolare la legge *minmax*: *min*imizzare i costi e *mass*imizzare i benefici per il massimo numero di agenti coinvolti. Il problema principale è quantificare in termini numerici i costi sociali – esempio: quanto vale una vita umana?

Test di verifica

Barger ha proposto alcuni test per verificare i risultati del metodo di analisi di Moor: sono stati rielaborati qui per semplicità d'uso.

Test deontologico si prenda lo scenario ottimale; chi può usufruire dello scenario ottimale? lo scenario tratta tutti gli esseri umani come fini, mai come mezzi?

Test pragmatico-realista La soluzione ottimale è ben bilanciata rispetto ai due 'corni' del dilemma? perché? quanto rispetta la legge *minmax*? in particolare, quali sono gli agenti (s)vantaggiati maggiormente? C'è un accordo sociale della maggioranza sull'efficienza dello scenario ottimale?[40]

[40] Questo test sussume diverse tradizioni filosofiche analitiche (realismo, pragmatismo) collegate al conseguenzialismo.

Test non-antropocentrico Quali sono le conseguenze in termini ambientali (sia in senso sociale che in senso naturale) della soluzione ottimale? Quali conseguenze per gli artefatti dell'infosfera?[41]

[41] Negli ultimi decenni sono emersi tra i filosofi morali diversi approcci innovativi, che non pongono l'uomo al centro: anziché essere centrati sull'intenzione o sull'azione (lato attivo), si concentrano sulle conseguenze per le vittime (lato passivo). Ce ne sono diversi: *land ethics* o *green ethics*, che guarda l'impatto ambientale delle tecnologie; l'animalismo o antispecismo, che considera l'uomo uno tra i tanti animali, senza per questo avere una posizione privilegiata; l'etica delle macchine o *machine ethics*, che dà rilevanza etica agli agenti artificiali.

Test esistenzialista Riprendendo il passo zero (*prima facie*), lo scenario ottimale risulta rispondente alla coscienza propria del soggetto giudicante, vale a dire l'analista del caso? In particolare, si è sentito libero di esprimere il proprio giudizio o costretto dalle convenzioni sociali e culturali del proprio ambiente?

Esercizio in classe Lettura collettiva e applicazione del metodo sovraesposto, completo di test, sui casi etici presentati qui di seguito. Si tratta di casi nuovi, di particolare interesse, emersi negli ultimi mesi.

Primo caso: privacy della vita digitale post mortem

Chi controlla la vita digitale post mortem? Il caso di Alison Atkins.[42] Alison è morta all'età di 16 anni il 27 Luglio 2012 a Toronto (Canada) per una malattia al colon. Sua sorella Jaclyn Atkins (studentessa universitaria di 20 anni) ha violato la password del MacBook Pro di Alison nell'ultimo periodo della malattia, quando era isolata in casa, perché la famiglia voleva accedere ai suoi account Facebook, Twitter, Tumblr, Yahoo e Hotmail. Alison infatti aveva fatto dei disegni e scritto delle poesie che Jaclyn voleva trasmettere sui social network per conservarli, come era solita fare la sorella.

Jaclyn ha in questo modo infranto la legge, perché i termini di licenza del contratto tra sua sorella e le società che gestiscono i servizi online non prevedono l'uso degli account da parte di terzi. Jaclyn si è connessa automaticamente ai servizi dal MacBook Pro di Alison senza sapere le password, e i suoi tentativi di cambiarle non sono andati a buon fine, e hanno infine bloccato gli account stessi per motivi di sicurezza e violazione della privacy. In particolare, il 21 Novembre 2012 Alison è scomparsa da Facebook, che la famiglia aveva usato per comunicare con gli oltre 500 amici di Alison condividendo la sua memoria.

La memoria digitale di Alison è chiusa a chiave nei server dei servizi online, e nessuno ha il diritto di accedervi, per via delle norme sulla riservatezza. Tra l'altro, Jaclyn ha scoperto suo malgrado che la sorella consultava giornali di dubbia natura (*dark journals*, nell'articolo originale) ed evidentemente non voleva che la famiglia ne venisse a conoscenza. Secondo la legge canadese, le comunicazioni elettroniche sono equiparate alle lettere cartacee, e non possono essere aperte senza il consenso del ricevente salvo ordinanza specifica per esempio per un'indagine di polizia.[43]

Vediamo ora come hanno reagito le diverse compagnie. I termini del servizio con **Yahoo** Inc. prevedono che alla ricezione dei certificato di morte tutti i dati dell'utente vengano cancellati in via permanente. L'utente deve dare un consenso esplicito al trasferimento dei propri dati agli eredi mediante una comunicazione all'azienda, ma pochissimo lo fanno perché non sono nemmeno a conoscenza di questa possibilità. La **Microsoft** Corp. (che possiede Hotmail) si è resa disponibile a passare le email su disco alla famiglia dietro la presentazione di "appropriata documentazione": in sostanza, si riserva di decidere caso per caso: il caso di Alison è finito sui giornali, ma quanti lo saranno? E l'azienda può permettersi di analizzare caso per caso con i numeri di account Hotmail esistenti? Le stime di **Facebook** sono di 580.000 utenti americani deceduti solo nel 2012, e non permette agli eredi o ai familiari l'accesso. In particolare, la richiesta di

[42] La fonte principale è l'articolo di Geoffrey A. Fowler sul *The Wall Street Journal* intitolato 'Life and Death Online: Who Controls Digital Legacy?' del 5 gennaio 2013.

[43] La riservatezza della comunicazione postale in Italia è sancita nella Costituzione stessa. L'equiparazione tra lettere elettroniche e cartacee è adottata in molti paesi del mondo.

accesso da parte della corte di San Jose (California) dell'account di Sahar Daftary, modella britannica deceduta con sospetto di suicidio, è stata respinta, perché la corte non è né un familiare né un erede.[44] Si noti che dal 2009 Facebook *non* disattiva l'account del deceduto, per far sí che i membri della sua rete possano lasciare commenti o foto del caro estinto sul suo muro digitale.[45] Anche **Google** Inc. ha preso posizione: il 12 Aprile 2013 è stato aperto il servizio Gestione Account Inattivo,[46] che permette ai morituri di decidere il destino post mortem di tutte le attività connesse ai propri account (Gmail, Picasa, YouTube, Blogger, Drive, G+. ecc.).[47] Si può scegliere la totale cancellazione o si possono specificare fino a 10 contatti fidati che decidono l'eliminazione parziale o totale dei dati in fasce periodiche (da tre mesi a un anno). In nessun modo l'account potrà essere usato post mortem, per esempio per mandare messaggi a nome del defunto. Si noti che l'account viene disattivato dopo un periodo di inattività (da tre mesi a un anno) e un sistema di verifica di reale inattività, mediante un sms telefonico: non c'è nessuna attesa di ricezione del certificato di morte o altro documento formale.

Il comportamento di Jaclyn e della famiglia è stato corretto? Se si fa accedere terzi a un account online, nessuno è piú sicuro di chi realmente stia scrivendo o messaggiando cosa: che succede all'identità personale digitale? È giusto che le compagnie distruggano la memoria digitale dei propri utenti una volta passati a miglior vita?[48]

[44] Ulteriore motivazione è di evitare la possibilità di far usare l'account del defunto per attività illecite come lo spam.

[45] Questa pratica viene chiamata *momorialized account*.

[46] Prontamente ribattezzato *Death Manager* in rete.

[47] Fanno eccezione Music Play e i film acquistati, perché la licenza d'uso di fruizione è personale e non cedibile.

[48] Si tenga conto inoltre dell'esistenza di servizi per mantenere attivi i propri account Twitter e Facebook post mortem, quali DeadSocial – dove si registrano dei messaggi che verranno mandati dopo il trapasso, a un certo ritmo – e LivesOn – che analizza il proprio account Twitter, genera in automatico tweet simili, e assegna un esecutore testamentario all'account identificato in un altro account Twitter.

Secondo caso: civili uccisi dai droni

Chi è responsabile in uno scenario di *cyberwar*, dove è quasi impossibile tracciare chi ha fatto cosa?[49] Dal 2009 gli attacchi militari degli Stati Uniti in Afghanistan e Pakistan, autorizzati dall'amministrazione Bush prima e dall'amministrazione Obama poi, hanno visto aumentare enormemente l'impiego dei **droni**, velivoli da guerra automatici senza pilota (*Unmanned Aerial Vehicles*, per esempio il RQ-1 Predator).[50] I droni sono impiegati per quattro tipi di azioni-obiettivo (*target goals*): uccisioni mirate; distruzioni di obiettivi militari (*signature strikes*); combattimenti aerei palesi; combattimenti aerei nascosti. Lo *jus in bello* si basa su due principi umanitari:[51] il principio di **discriminazione** richiede che i soldati non attacchino direttamente i civili e prendano le dovute precauzioni per non avere perdite tra i civili, mentre il principio di **proporzionalità** asserisce che l'uso della forza dev'essere proporzionale alle forze avversarie in campo. L'uso dei droni viene contestato in merito al primo dei due principi.[52]

Secondo il report della New America Foundation, dal 2006 al 2009 nel solo Pakistan i missili lanciati da droni hanno ucciso 750-1000 persone, di cui il 66-68% membri di Al Qaeda, talebani e gruppi associati (tra cui 20 leader) e 31-33% di di civili.[53] Questi numeri sono stati contestati dagli ufficiali militari americani, e nel 2010 gli attacchi su obiettivi militari hanno toccato il record di 122. Nel Maggio 2011 un attacco di questo tipo ha ucciso 24 soldati pakistani, e l'ambasciatore Usa in Pakistan, Cameron Munter, ha dovuto chiedere scusa e mettere a conoscenza il governo pakistano in anticipo degli attacchi previsti.[54] Risulta quindi **evidente che i droni non sono infallibili** o "chirurgici" nelle loro azioni in battaglia.[55]

C'è differenza tra un civile ucciso per errore da un drone e uno ucciso da un soldato, dal punto di vista delle vittime? L'impatto psicologico di essere passibili di attacco da parte di un velivolo senza uomini sembra essere ancora piú terrorizzante per i civili di un attacco 'tradizionale': la guerra diventa letteralmente inumana. Cambia qualcosa se il soldato sta comandando in remoto un velivolo, dal punto di vista delle vittime?[56] Cosa cambia dal punto di vista dei soldati? Si tenga conto che le interfacce di controllo sono volutamente simili ai videogiochi. Il principe Harry – l'erede alla corona inglese – ha dichiarato in un'intervista di essersi scoperto un ottimo cecchino nel servizio militare in Afghanistan, dove era in servizio sugli elicotteri Apache, probabilmente perché è un giocatore abituale di videogiochi di guerra su PlayStation e XBox.[57] In definitiva, la guerra dei droni completamente o parzialmente automatizzati è un'alternativa reale e meno disumana della guerra tradizionali?[58]

[49] Mariarosaria Taddeo. Information warfare: A philosophical perspective. *Philosophy & Technology*, 25(1):105–120, 2012

[50] Il 14 maggio 2013, vale a dire due settimane fa, per la prima volta nella storia dell'umanità un drone (X-47B) è decollato da una corazzata in mare aperto, una delle manovre aereonautiche tecnicamente piú difficili in assoluto. Ad oggi, i droni sono usati solamente dagli Usa, dai britannici e da Israele.

[51] Questi principi vengono definiti dopo la seconda guerra mondiale.

[52] Naturalmente, esiste il problema che formalmente gli Usa non sono in guerra con i paesi dove i droni vengono impiegati, e quindi ci sono anche altri tipi di violazione del diritto internazionale. Non li vediamo qui perché non sono legati all'uso specifico dei droni.

[53] Prima del 2006 i droni erano impiegati per operazione di sorveglianza (foto del campo di battaglia), ma non erano armati, come spiegato nel rapporto della *charity* britannica Medact del 2012. Per la cronaca, l'Italia è un paese fornitore di componenti per i droni Usa.

[54] Fonte principale: articolo di Peter Bergen della New America Foundation, dal titolo 'Drone Wars', del 24 Aprile 2013.

[55] L'argomentazione dei militari è che dietro ogni civile può nascondersi un guerrigliero terrorista, in altri termini che non esistono veri e propri civili in contesti come l'Afghanistan o il Pakistan.

[56] Tecnicamente vengono chiamati *remotely piloted vehicles* (RPV), in uso fin dalla guerra nel Vietnam. Nel Febbraio 2013 è stata istituita la medaglia al merito per soldati che combattono dietro una tastiera o un joystick. Fonte: Washington Guardian, 'Pentagon creats new medal for cyber, drone wars', 13 Febbraio 2013.

[57] Fonte: articolo di Lama Hasan su Abc News, 'Prince Harry's War Games Trigger Backlash', 23 Gennaio 2013.

[58] Si tenga conto che l'industria dei droni è in rapida espansione: le stime di Chris Anderson, direttore di *Wired*, parlano di 30 miliardi di dollari nel 2015.

Terzo caso: Data mining & privacy

Cosa può succedere alla riservatezza quando i dati personali ricavati da diversi database vengono incrociati? Il caso di Target e il *pregnancy prediction score*.[59] Nel 2002 lo statistico Andrew Pole comincia a lavorare per Target Corporation, una società di elaborazione dati. Dal settore marketing arriva la seguente richiesta: "vogliamo sapere se una cliente è incinta, anche se non vuole che lo sappiamo, dai suoi acquisti. Lavoraci su."[60] Al marketing spiegano che mentre i consumatori normalmente comprano il pane in panetteria e giocattoli in un negozio di giocattoli e sono abitudinari, le donne incinte tendono a rimettere in discussione la routine, e nel secondo trimestre di attesa comprano un po' di tutto. Poiché Target vende di tutto, sapere se una consumatrice è in attesa permette di mandare delle offerte mirate (vitamine premamam, vestiti da incinta, ecc.) e di catturare la cliente che poi rimane fedele per anni.

Da anni la politica aziendale di Target è la profilazione dettagliata del cliente, tenendo traccia di informazioni demografiche e altro tramite carte fedeltà, email, clic sul sito web, questionari e cosí via. Mediante tecniche di *analytics* vengono inferiti dati come il reddito, l'appartenenza etnica, gli interessi, ecc. Pole analizza i dati e trova 25 prodotti che, analizzati insieme, danno un *pregnancy prediction score*.

Un giorno del 2012, il signor M., di Minneapolis, arriva infuriato in un punto vendita di Target, urlando che sua figlia adolescente ha ricevuto nella cassetta delle lettere buoni sconto su vestiti per neonati e altri articoli premamam. Il signor M. minaccia di fare causa a Target per "istigazione alla maternità precoce" alla direzione, la quale decide di scusarsi ufficialemente per l'errore. Al momento della telefonata dell'azienda, il signor M. si scusa, ammettendo che Target aveva ragione: sua figlia è incinta, esattamente al terzo mese.

In seguito alla vicenda Target cambia la politica di buoni sconto, mescolando a quelli premamam altri buoni sconto per il padre o comunque per la famiglia, perché si rende conto che in questo modo le clienti in attesa li usano di piú perché rischiano meno di essere scoperte prima del tempo.

Il comportamento di Target può essere considerato eticamente corretto?[61] E quello del signor M.?

[59] Fonte principale: articolo di Charles Duhigg del NY Times dal titolo 'How Companies Learn Your Secrets', pubblicato il 16 febbraio 2012.

[60] Nell'articolo Pole si definisce *nerd* e *analytics evagelist*.

[61] Si noti che, allo stato attuale delle conoscenze, nessuna legge è risultata violata.

PROVA SCRITTA FINALE: approfondimento di circa 20.000 battute (spazi e bibliografia compresi), di cui due terzi di analisi di un caso studio dalla propria esperienza didattica o comunque informatica, applicando il metodo dei livelli di astrazione di Floridi visto nelle prime due lezioni, e un terzo caso di analisi di un caso etico tra i tre presentati in classe (oppure uno extra, se concordato con il docente), mediante il metodo dell'analisi dei casi etici di Moor-Barger.

Riferimenti bibliografici

[1] Chris Anderson. *La coda lunga: Da un mercato di massa a una massa di mercati*. Codice Edizioni, 2010.

[2] Reinier B. Bakels. Software patentability: what are the right questions? *European Intellectual Property Review*, 10:514–522, 2009.

[3] Robert N. Barger. *Computer Ethics*. Cambridge University Press, 2008.

[4] Martin Davis. *Il calcolatore universale*. Adelphi, 2012.

[5] John Derbyshire. *Unknown Quantity: A Real and Imaginary History of Algebra*. Plume, 2007.

[6] Nathan Ensmenger. *The computer boys take over: computers, programmers, and the politics of technical expertise*. The MIT Press, 2010.

[7] Luciano Floridi. *La rivoluzione dell'informazione*. Codice edizioni, 2012. Prefazione di Juan Carlos De Martin.

[8] Giustino Fumagalli. *La tutela del software nell'Unione Europea. Brevetto e diritto d'autore*. Nyberg, 2 edition, 2005.

[9] Federico Gobbo and Marco Benini. From Ancient to Modern Computing: A History of Information Hiding. *IEEE Annals of the History of Computing*, 99(PrePrints), 2013.

[10] Federico Gobbo and Marco Benini. The Minimal Levels of Abstraction in the History of Modern Computing. *Philosophy & Technology*, pages 1–17, 2013.

[11] Federico Gobbo and Marco Benini. Why zombies can't write significant source code: The Knowledge Game and the Art of Computer Programming. *Journal of Experimental & Theoretical Artificial Intelligence*, 2013. forthcoming.

[12] Deborah G. Johnson. *Computer Ethics*. Prentice Hall, 1985. 2nd ed 1994; 3rd ed 2001; 4th ed 2009.

[13] Lawrence Lessig. *Cultura libera. Un equilibrio fra anarchia e controllo, contro l'estremismo della proprietà intellettuale*. Apogeo, 2005.

[14] Steven Levy. *Hackers. Gli eroi della rivoluzione informatica*. Shake edizioni, 2002.

[15] Leonard Mlodinow. *Euclid's windows: the story of geometry from parallel lines to hyperspace*. Free Press, 2002.

[16] James H. Moor. Just consequentialism and computing. *Ethics and Information Technology*, 1:65–9, 1999.

[17] Eric Steven Raymond. *La cattedrale e il bazaar*. Apogeo Online, 1998.

[18] Richard Stallman. *Software libero, pensiero libero*. Nuovi equilibri, 2003.

[19] Mariarosaria Taddeo. Information warfare: A philosophical perspective. *Philosophy & Technology*, 25(1):105–120, 2012.

[20] John von Neumann. First Draft of a Report on the EDVAC. *IEEE Ann. Hist. Comput.*, 15(4):27–75, October 1993.