Sviluppo dei sistemi informatici visti da uno studente di informatica durante l'evoluzione dell'ultimo ventennio

01/12/2006

Basandosi su informazioni personali ottenute o vissute, si vuole analizzare il software negli ultimi anni, in ordine cronologico, facendo opportune riflessioni. Questo scritto è organizzato su questi argomenti:

1. Capacità di calcolo;
2. Utilizzo delle risorse;
3. Tipologia di rete
4. Linguaggi di programmazione;
5. Programmi ad uso commerciale (giochi, applicazioni di elaborazione semplici, testo, calcolo);
6. Interesse peculiare nello sviluppo del software;
7. Sicurezza dei sistemi.

Ovviamente questo testo è stato ottenuto riflettendo sull'esperienza di uno studente, quindi sono direttamente collegate alla carriera scolastica, passando dalle elementari, passando per le scuole medie inferiori e superiori, fino ovvimaente all'esperienza universitaria.

1 Prima del 1988

Dalle informazioni ricevute:

1. Il problema era il costo, direttamente proporzionale alla computazione, ovvero alla capacità di calcolo: i computer erano lenti e costosi, i programmi piccoli ed efficienti, il software esattamente ritagliato sul dispositivo, completamente non estendibile;

2. Le risorse erano centellinate; essendo poche e molto costose, venivano utilizzate al meglio senza poter minimamente sprecare nulla, al limite del singolo bit.

3. Le reti erano di tipo stella, un potente, quindi molto costoso, elaboratore centrale al quale collegare i terminali degli utenti-utilizzatori, che non possiedono altro che un monitor e una tastiera. Il problema era quindi sincronizzare i vari utenti per ottimizzare il flusso del calcolo, rendendolo il più omogeneo possibile, senza dare priorità ad un utente piuttosto che a un altro.

4. Il linguaggio di programmazione era il linguaggio macchina, quindi l'astrazione era un concetto ancora lontano. Questo dava la possibilità al programmatore di operare a un basso livello rispetto alla macchina, cosa da un lato molto potente poichè non ci sono limitazioni al programmatore, dall'altro molto pericoloso, perché queste limitazioni possono proteggere la macchina e quello che contiene.

5. (5) I programmi erano ritagliati per l'applicazione da offrire, non erano estendibili, e poco riutilizzabili. Non si può ancora parlare di applicazioni commerciali in quanto il computer non era qualcosa di commerciabile su vasta scala, i sistemi di elaborazione sono solo per le ditte che dispongono di sufficiente denaro. Erano presenti le prime console per l'intrattenimento, con le quali si potevano abbozzare dei primordiali programmi con semplici linguaggi di programmazione come il Basic.

6. L'interesse è la computazione, ogni applicazione sfruttava tutte le potenzialità degli elementi forniti, Per intenderci, se si vuole fare un contatore di 200 elementi possibili, si preferisce un byte a un integer (notazione Pascal) per non sprecare 1 byte in memoria statica. In pratica, utilizzando la metafora degli strati implementativi sopra alla macchina, si utilizzano le funzioni proprie della macchina sulla quale si lavora. Non si parla ancora di tecnologie, ma solo di funzionalità, e l'astrazione è ancora molto lontana da venire.

7. (7) La sicurezza stava nel fatto che non esistevano applicazioni on-line che necessitassero sicurezza. Tutti i dati privati erano su supporto cartaceo.

2 Anni 1988-2000

Dalle esperienze vissute:

1. La computazione è migliorata, si cominciano a sprecare risorse visto che ormai i prezzi, rispetto alla capacità di calcolo, sono drasticamente

crollati. Si pensa alla multi-medialità, i cui pionieri costruiscono apparati che sfruttano i sistemi per fornire un prodotto gradevole alla vista, con suoni e video, ma in realtà, non avendo una base solida di progettazione e operando solo aggiungendo funzionalità ai sistemi già esistenti, si creano delle applicazioni anche troppo pesanti sotto il profilo della computazione.

2. Le risorse sono in piena evoluzione, ma ancora si tiene ancora molto in conto il possibile spreco di queste.

3. La rete è ancora a stella, ma l'elaborazione non è più riservata al solo elaboratore centrale, anzi ogni terminale è una capacità intrinseca di calcolo, notevolmente inferiore a quello centrale ma comunque rilevante.

4. I linguaggi di programmazione sono stati estesi per supportare gli oggetti, ma continuano a essere utilizzati in maniera statica. Anche i primi linguaggi veramente ad oggetti, vengono utilizzati solo per le loro capacità grafiche e non per le loro potenzialità ad oggetti.

5. I programmi sono molto meno specializzati, difficilmente estendibili, ma estesi, coprono molte più parti di quelle utilizzate. Solamente i veri esperti utilizzano tutte le parti offerte. Si cominciano a vedere gli oggetti.

6. Particolare interesse hanno ora i dati, risulta di particolare interesse lo 'storico' di un DB. I sistemi sono improntati al mantenimento dei dati.

7. La sicurezza è amministrata tramite il solo utilizzo di password conferite da sistemi considerati garanti, ma effettivamente la sicurezza è lasciata alla fiducia dei dati forniti dall'utente. Soprattutto ai sistemi viene applicata una sicurezza, anziché analizzarla, progettarla e implementarla in fase di sviluppo del software, come si è dimostrato la metodologia migliore.

3 Anni 2000-2005

Dalle esperienze vissute e dalla messa in pratica degli insegnamenti precedenti:

1. la computazione è ancora più potente, quindi si può astrarre molto dalla tecnologie che si utilizzano. Una nuova metodologia di programmazione è il trend del momento, ovvero la metodologia ad oggetti. Grazie a questa metodologia si creano sistemi estremamente modulari, facilmente estendibili, anche perché sono caratteristiche proprie dell'entità oggetto alla base della metodologia.

2. L'utilizzo delle risorse, coerentemente con la metodologia ad oggetti, è indubbiamente votata allo spreco delle risorse, sia perché ora sono in grande quantità, sia perché si costruiscono software in grado di recuperare queste risorse quando vengono inutilizzate (Garbage Collector), in modo completamente trasparente rispetto al programmatore.

3. La rete è quasi globale, ora non si delega la computazione ad un computer centrale, ma è l'esatto opposto, ovvero la capacità di calcolo è distribuita, le grandi computazioni vengono organizzate su più terminali, perché si è capito che la forza della computazione sta nell'organizzazione, non nella velocità del calcolo. Un esempio lampante è dato dal motore di ricerca Google che non è altro che una grande quantità di computer mediamente potenti che, lavorando in parallelo, riescono a velocizzare notevolmente la computazione fornendo un servizio quasi in real time. Il cellulare è il simbolo di questi anni, vengono prodotti telefoni sempre più piccoli e con sempre maggiori funzionalità che però vengono sfruttate da pochi.

4. I linguaggi di programmazione sono in un momento di forte transizione, si parla soprattutto di oggetti, che sono il concetto trainante, e rendono le applicazioni facilmente estendibili e riutilizzabili. Il grosso problema è che, ora più che mai, si cerca di rendere l'organizzazione del codice il più possibile modulare ed estendibile. Inoltre si trovano nuovi modi per la documentazione automatica del codice e il testing risulta uno dei problemi di maggior interesse: non si scrive più il codice per poi testarlo, ma si testa il codice perché rispetti a priori le specifiche date, ovvero quelle definite nelle interfacce. Si utilizzano largamente linguaggi di programmazione che descrivono la struttura delle applicazioni e le loro funzionalità; nei tool più avanzati si può tranquillamente costruire un'applicazione partendo dalla struttura e poi riempiendo gli spazi vuoti dei metodi imposti. I sistemi sono sempre più complessi e vanno giustamente ingegnerizzati.

5. I programmi sono sempre meno specializzati, ma nelle applicazioni di uso comune, ci si riferisce a programmi di editing di testo, fogli di calcolo, addirittura compilatori. Essi, nelle applicazioni per le industrie, sono sempre ottimizzati per l'applicazione a cui sono rivolti, quindi sono poco o addirittura non riutilizzabili e quindi vanno ri-progettati ogni volta.

6. L'interesse peculiare sta nell'organizzazione delle applicazioni, che devono essere estendibili, riutilizzabili e modulari Sempre rispettando la metodologia degli oggetti e l'ingegnerizzazione del codice, si producono compilatori costruiti in modo modulare, fatti di plugins che offrono le funzionalità più disparate.

7. Sicurezza: data l'estensione della rete, la si vuole sfruttare per fornire agli utenti servizi che necessitano di sicurezza, quali gestione multimediale di conti correnti, acquisti on-line o semplicemente posta sicura. Vengono applicati metodi che 'garantiscono' la sicurezza sempre più evoluti, ma vengono utilizzate le piattaforme già esistenti, solamente aggiungendo funzionalità. Questo rende le cose difficilmente estendibili e comunque la sicurezza richiesta è sempre limitata. Ci si rende conto che l'unico modo per garantire la sicurezza è inserirla nella progettazione delle applicazioni, che devono averla come uno degli obiettivi da raggiungere.

4 Anno 2006 verso il futuro

Dai nuovi insegnamenti proposti:

1. la computazione è ovunque e va giustamente sfruttata, riprendendo uno slogan di una nota marca, 'la computazione è tutta intorno a noi'. Telefonini, agente elettroniche, palmari, navigatori satellitari per auto, sono tutti esempi di dispositivi dalle capacità computazionali,

2. non ci si rende ormai conto che le risorse hanno comunque un limite, vengono prodotte applicazioni anche troppo pesanti, con funzionalità impressionanti, ma che in pratica non vengono molto spesso utilizzate.

3. la rete ormai è ovunque, viste le tecnologie wireless che ormai ricoprono il globo, e grazie, a questo, ogni utente è virtualmente in rete, o meglio ha le potenzialità per esserlo.

4. I linguaggi di programmazione nascono concettualmente ad oggetti, ma si vuole passare alla programmazione ad agenti, inizialmente utilizzando tutte le risorse precedenti in quanto compilatori

5. I programmi sono tornati ad essere abbastanza ritagliati sul sistema nel quale devono operare, per quanto riguarda le applicazioni che devono funzionare sui nuovi dispositivi mobili i quali hanno molti problemi sulle risorse, in quanto possiedono una batteria che non è eterna, e quindi la computazione non può sprecare energia.

6. L'interesse particolare sta nel controllo, ovvero ogni entità capace di elaborare deve poter elaborare, utilizzando la struttura che la contiene per interagire pro-attivamente con gli altri sistemi di elaborazione

7. La sicurezza è un punto nodale, si è capito che per garantire la sicurezza in un sistema, non la si deve aggiungere, ma essa deve essere portata avanti durante tutto l'iter di sviluppo del codice, partendo dall'analisi e in tutta la parte di progetto. Quindi si progettano e implementano sistemi che incorporano la sicurezza al loro interno.

5 Conclusioni

Come si è notato, lo sviluppo del software non è stato al passo con lo sviluppo dell'hardware e questo ha portato inevitabilmente alla produzione di programmi molto 'pesanti' in senso computazionale. Inoltre l'organizzazione del software, in quanto fondamento, si è imposta solo molto tardi rispetto alla sua evoluzione. A questo punto, ci si deve chiedere dove devono essere spostati gli insegnamenti che ormai devono coprire un insieme troppo vario di concetti. Insegnare subito gli oggetti e la loro organizzazione? Potrebbe essere una soluzione, ma poi le nozioni di base dell'informatica, come l'ordinamento o lo scambio di due valori, verranno messi in ombra. Ormai il teorema di Bohm e Jacopini, ovvero della programmazione strutturata, è diventato uno standard, avendo ormai eliminato l'istruzione dei vecchi linguaggi di programmazione che faceva svanire questo teorema, il GOTO <label>. Ma molte altre cose andranno 'perdute' se si vuole progredire con i sistemi complessi.