Sistemi Informativi

Alessio Massetti

Politecnico di Milano - A.A. 2011/12 - II Semestre

Contents

Informazioni sul Corso:

Docente:

Mariagrazia Fugini

E-Mail:

mariagrazia.fugini@polimi.it

Libri di Testo:

D. Ardagna, M.G. Fugini, B. Pernici, P. Plebani, Sistemi informativi basati su web, serie Sistemi Informativi, Vol. VI, Franco Angeli, 2006

Pagina del Corso:

Corsi Metid

Orario:

giovedì

10.30 - 12.00 (B 2.2)

venerdì

10.30 - 13.00 (L.26.0.2)

1 Risorse e processi

1.1 Definizione

Risorsa è tutto ciò con cui l'azienda opera. Si suddividono in risorse esterne (ambiente, mercato, clienti) e risorse interne (di scambio, i struttura e di gestione).

1.2 Ciclo di vita

Il ciclo di vita di una risorsa:

- Pianificazione
- Gestione
- Acquisizione
- Manutenzione

Esempi: personale, denaro, informazione

1.3 Processo

Il processo è l'insieme di attività che l'organizzazione nel suo complesso svolge per gestire il ciclo di vita di una risorsa e per raggiungere un risultato definito e misurabile. Il processo per operare ha bisogno di informazioni.

1.4 Piramide di Anthony

I processi sono organizzati in modo gerarchico all'interno di una azienda:

- Decisioni strategiche
- Decisioni direzionali
- Decisioni operative

Anche i dati sono divisibili in strategici, direzionali e operativi.

1.5 Caratteristiche dei processi

1.5.1 Economicità

Fornitura di servizi ai più bassi costi possibili, in relazione ad una specifica qualità.

1.5.2 Efficienza ed Efficacia

Definiamo l'efficienza come output effettivo / input mentre l'efficacia l'output effettivo / output atteso. La prima quindi definisce quanto siano state "giuste" le cose fatte, la seconda quanto si avvicinino alle reali.

1.6 Sistema informativo vs Sistema informatico

Il sistema informativo è il flusso di gestione della libera informazione, può eventualmente fare uso della tecnologia del sistema informatico a cui vi si appoggia. Un sistema informativo basato su sistema informatico è detto sistema informatizzato.

2 I Sistemi informativi in rete

2.1 Classificazione UE

2.1.1 Servizi di informazione

Accesso a pagine statiche. L'utente può interagire tramite cookie ma non inserisce contenuto informativo nel sistema. Nessuna interazione.

2.1.2 Servizi di comunicazione

Servizi di interazione (es. e-mail) in cui l'utente è identificato e le sue comunicazioni possono rimanere memorizzate. Interazione in un senso.

2.1.3 Servizi transazionali

L'utente ottiene servizi modificando dati e interagendo con DBMS. Interazione in due sensi.

2.2 Classificazione per accesso

2.2.1 Sito internet

Accessibile da tutta la rete

2.2.2 Sito intranet

Accesso limitato all'interno dell'organizzazione

2.2.3 Sito extranet

Accessibile ad un gruppo di clienti ben identificato

2.3 Classificazioni di servizi on-line

2.3.1 Business to Consumer B2C

Servizi operativi per clienti

2.3.2 Business to Business B2B

Servizi operativi per aziende

3 Architetture Web Information Systems

3.1 Architetture dei SI

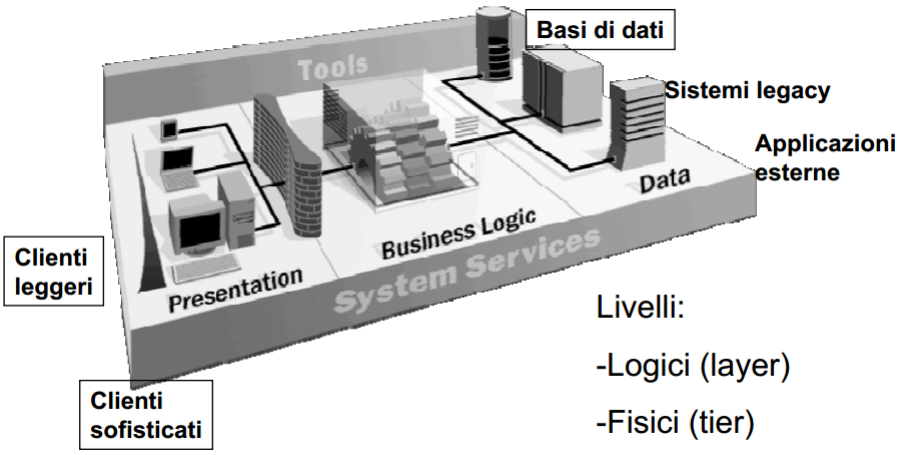
3.1.1 Proprietà delle architetture

- Affidabilità
  - Probabilità che il sistema sia operativo
- Scalabilità
  - Capacità di soddisfare richieste crescenti con aggiornamenti adeguati
- Tolleranza ai guasti
  - Implica disponibilità e sostituibilità

3.1.2 Sistema informativo distribuito

Un SI distribuito è un sistema complesso con più sorgenti informative, più basi di dati, collegamenti in rete e processi interconnessi.

3.2 Livelli logici e livelli fisici (tier)



3.3 Architetture a 1, 2, 3 livelli

I livelli in cui si può suddividere una architettura sono

- Presentazione
- Logica Applicativa

- Dati

Un SI può quindi essere impostato come

- Single-tiered: tre livelli assegnati ad una macchina
- Two-tiered: layer divisi tra stazione-lavoro e server che ospita i dati
- Three-tiered: ogni livello su una macchina
  - Nell'architettura a tre tier è comunque possibile incrementare il numero di tier fisici per aumentarne la scalabilità. Basi di dati e logiche applicative possono quindi trovarsi su più server.

3.4 Architetture WIS

3.4.1 Siti Web vs. Applicazioni Web

Il sito è un sistema di visualizzazione di documenti ipermediali, con il link che è un meccanismo che permette di navigare attraverso le risorse di un sistema. L'applicazione web è l'estensione di un sito Web in cui gli utenti possono accedere alla logica applicativa attraverso un browser.

3.4.2 Architettura di base

Browser -> Web Server -> Local File system. Vengono usati URL e protocolli HTTP (attenzione, il protocollo HTTP è connectionless). I linguaggi sono HTML e XML.

3.4.3 Web applications

Insieme di applicazioni basate su pagine generate dinamicamente. L'utente modifica lo stato dell'applicazione. Sono disponibili tre tecnologie:

- CGI
- Servlet
- Java Server Pages

3.4.4 Gateway

I server Web possono richiamare programmi, trasmettendo eventualmente anche altri parametri. Il CGI è quel meccanismo che consente al Web Server di eseguire applicazioni esterne in grado di creare pagine dinamicamente. Non è un linguaggio di programmazione, bensì un protocollo di comunicazione (scritto in perl) che permette la trasmissione di dati. Vediamone la sua esecuzione:

1. Il server riconosce dall'URl che la risorsa richiesta dal client è un eseguibile

2. Il server decodifica i parametri inviati dal client e riempie le variabili d'ambiente
3. Il server lancia in esecuzione l'applicazione richiesta
4. L'applicazione stampa la sua risposta sullo standard output
5. Il server ridireziona lo standard output sulla rete e quindi verso il client

3.4.5 Thin / Fat Client e Server intermedio

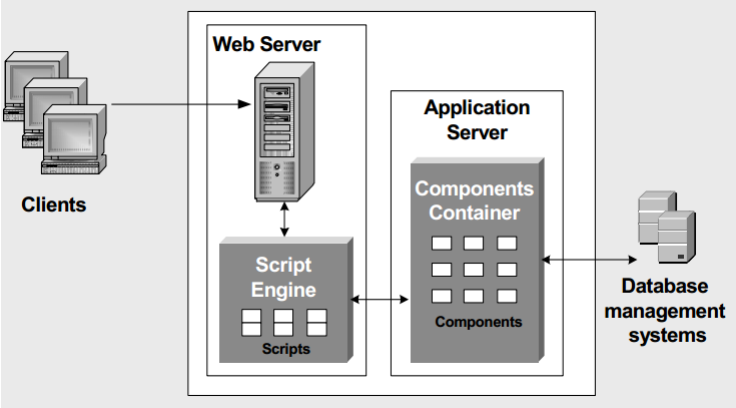
Un fat client ha, a livello utente, logica applicativa e accesso ai dati. Un thin client invece consente esclusivamente funzione di presentazione, alleggerendo le funzionalità della stazione utente. Il fat client necessita quindi di amministrare diverse macchine client.

3.4.6 Server intermedio

Consente di ridurre il carico al DBMS server. Il middle tier, infatti, mantiene le connessioni con il DBMS. Nelle applicazioni reali si usa più di un tier o si introducono tier dedicati a supporto della comunicazione detti middleware.

3.5 Server Farm

L'architettura a tre tier fisici, può essere realizzata anche tramite una server farm, ovvero un insieme di server che condividono un'unica risorsa. La soluzione più usata è spesso quella a cinque tier. Browser utente, Web server, Script engine mentre gli ultimi due tier sono quelli che si occupano del dbms.



3.5.1 Web server

Tra utente finale - che accede al sistema via browser (ruolo di primo tier applicativo) e il sistema informativo aziendale. Si occupa della presentazione delle informazioni verso il client: restituisce pagine HTML statiche.

3.5.2 Script Engine

Processo che genera una pagina dinamica interagendo con application server o DBMS server. Le pagine dinamiche vengono poi restituite al web server.

3.5.3 Application Server

Ha il ruolo di Middle Tier: implementa BL e in alcuni casi svolge il ruolo di contenitore di oggetti.

3.6 Progettazione architetturale

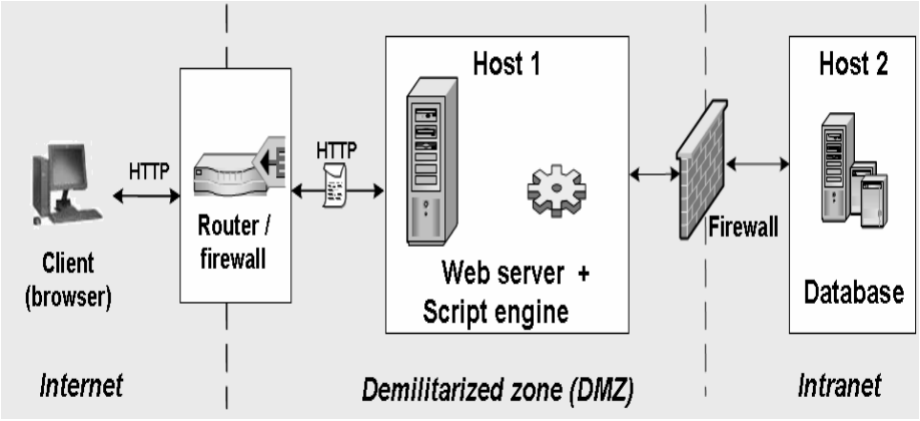
Il problema consiste nello stabilire quanti elementi ha il sistema e quanti tier fisici devono essere installati su macchine separate, quindi come collegare tra loro le diverse macchine.

3.6.1 2 tier - single host.

Una sola macchina fisica (Host 1) ospita:

- Web Server
- Script execution engine
- Database

3.6.2 3 tier - dual host.



- Web server e Script engine su stessa macchina
- DB eseguito su macchina dedicata.
- Un secondo firewall introduce un secondo dominio di sicurezza a protezione del DBMS.

La rete in cui è collegato il Web Server viene detta DMZ, consiste in un segmento, o insieme di segmenti, della rete localizzati tra reti protette e non protette. E' necessario tempo aggiuntivo per violare il secondo firewall e il DBMS, tempo che può essere utile al sistema di Intrusion Detection (IDS) e agli amministratori per rilevare e bloccare l'attacco. Il vantaggio è di avere un dimensionamento meno critico della macchina. Il Middle tier esegue operazioni CPU intensive mentre il DBMS esegue operazioni I/O intensive.

4 Tecnologie Web

- Pagine dinamiche
- Form/Script lato server
- Gestione dello stato: sessione

4.1 Architetture Web

- CGI
- Form
- Progettazione con Form
- Cenni ad architetture distribuite e Middleware

4.1.1 CGI

Invio dei parametri ad un programma CGI: esistono due possibilità, GET e POST. Nella get i parametri sono codificati all'interno dell'URL mentre nel post i parametri sono spediti al server usando il body del messaggio di richiesta HTTP.

4.1.2 Form HTML

Esempio:

```
<html>
<body>
<form action="http://www.mysrvr.it/cgi-bin/xyz.exe" method="post">
<p>Dimmi il tuo nome:  <input type="text" name="chisei" value="da inserire"></p> <input
</form>
</body>
</html>
```

Quando l'utente seleziona il bottone di tipo submit viene quindi formulata una richiesta HTTP. Se il metodo è GET, il parametro verrà visualizzato nell'url, altrimenti nel body.

Riepilogando:

- Input dall'utente
- Parametri (<INPUT type="tipoparam"):
  - text boxes (text)
  - buttons (submit, reset)
  - radio buttons (radio)
  - password (password)
  - hidden (valore fisso, non visibile all'utente)
- I parametri hanno un nome a cui si associa un valore compilando la form
- submit: si invia il contenuto del modulo al server come parte di un'altra richiesta HTTP di una pagina speciale (eseguibile)
- L'indirizzo della pagina eseguibile e' contenuto nella action

Cosa accade nel **lato server?**

Arriva una richiesta HTTP, eventualmente con dei parametri, si utilizzano delle variabili d'ambiente - l'applicazione elabora la richiesta e produce quindi la risposta.

4.2 Plug-in Server Web

Permettono di servire le richieste HTTP standard e rimandare le pagine eseguibili ad altro processo già in esecuzione (es. Microsoft Active Server Pages)

4.2.1 Moduli compilati (es. Java Servlet)

Estensioni al server Web di tipo filtro sulla normale elaborazione. Hanno accesso a delle API che forniscono i dati richiesti - i moduli producono poi output HTML che viene inviato al browser richiedente.

- **Vantaggi**: efficienti e adatti per applicazioni complesse; gestiscono anche autenticazione, autorizzazioni ed errori di login utente
- **Svantaggi**: mix di logica di business e presentazione; codice lungo e complesso; problemi di aggiornamento (specie su Intranet). Devono molto spesso essere registrati sul server.

4.2.2 Programmi eseguibili

Pagina HTML che contiene script per elaborare la logica di business. Uno script è un file nel file system del **server**. Lo script viene quindi interpretato dal server e che **alla fine** produce in output una pagina html. L'estensione del nome dice al server quale programma deve usare per interpretarla. **Offerte commerciali:** Java Server Pages (JSP), PHP, Microsoft Active Server Pages (ASPX).

La pagina viene quindi interpretata e localizzata nella directory specificata, il risultato è una pagina HTML formattata inviata al Client.

Utili per:

- Per generare pagine dinamicamente
- Per accedere a pagine riservate
- Per accedere a servizi
  - per accedere a basi di dati

4.2.3 Scripted page

E' un file memorizzato nel file system del Web server che contiene degli script interpretabili. Gli script interagiscono con oggetti sul server e alla fine producono un output HTML: una scripted page è simile a pagina HTML standard, ma include tag/token speciali interpretati dal server.

4.3 Gestione dello stato del Client

HTTP è connection-less: è necessario mantenere lo "stato" del client sul server

Il W3C ha introdotto nell'ambito di HTTP il meccanismo dei cookie

4.3.1 Cookie

I cookie sono una collezione di dati che un Web server puo' richiedere ad un browser per ogni richiesta HTTP. Parametri

- Nome
- Valore
- Scadenza
- Percorso
- Dominio
- Richiesta di connessione sicura

4.4 Sessioni

Si definisce sessione un singolo uso coerente del sistema. Le sessioni possono essere mantenute in quattro modi:

1. Inserimento di tutte le variabili di stato nei cookie
2. Inserimento di una chiave univoca in un cookie e suo utilizzo con un dizionario gestito dal server

3. Inclusione di tutti i valori delle variabili di stato come parametri inseriti in ogni URL

4. Inclusione di una chiave univoca come parametro in ogni URL

E' possibile memorizzare quindi le informazioni in un cookie oppure trasmettere la scelta effettuata ad ogni interazione con il sito web come Parametro Nascosto nelle form.

4.4.1 Utilizzo dei Cookie

Informazioni riguardanti le richieste dell'utente sono memorizzate in cookie.

Due alternative:

1. creare un identificativo di sessione e lato server memorizzare i dati relativi alle richieste dell'utente

2. memorizzare nei cookie i dati relativi alla richiesta.

4.4.2 Cookie e dati

Non viene memorizzata lato server alcuna informazione sulla sessione in corso. Nei cookie restituiti al client sono inserite le informazioni necessarie a proseguire la sessione (data e abbonamento), insieme alla pagina web che consente di effettuare le prenotazioni.

Utilizzo di parametri: Problema nella gestione dello stato di una applicazione tramite cookie: gli utenti possono scegliere di disabilitare il loro utilizzo, configurando il browser quindi la sessione non viene gestita correttamente dall'applicazione sul server.

E' possibile invitare gli utenti a abilitare i cookie, ma non è possibile forzare il loro utilizzo. Soluzione alternativa, realizzata gestendo i dati come sopra, è utilizzare parametri nascosti inseriti nelle form inviate all'utente, che verranno rimandati al server con le richieste successive.

4.4.3 Parametri e identificativo di sessione

Viene creato un identificativo di sessione e i dati relativi vengono memorizzati lato server. Viene quindi inserito un parametro nascosto nella prima form, questo parametro verrà inviato con i parametri inseriti esplicitamente dall'utente nella seconda form e così via...

4.4.4 Parametri e dati

Analogamente alla soluzione 2, anche i dati possono essere gestiti tramite parametri nascosti. Questo è però possibile solamente per applicazioni limitate altrimenti appesantirebbe il server.

4.5 Sistema distribuito

Come scambiare i dati tra sistemi diversi per tipo, dati, macchine (hw/sw), configurazioni ecc? Serve interoperabilità tra dati e applicazioni.

Un sistema distribuito è un sistema costituito da un insieme di applicazioni logicamente indipendenti che collaborano per il perseguimento di obiettivi comuni attraverso una infrastruttura di comunicazione hardware e software. Anche le applicazioni e le basi di dati sono distribuiti e risiedono su più nodi lavorativi.

4.5.1 Modello basato su oggetti

Oggetto

- Dati (stato)
- Funzioni (comportamento)
- Identità
- Istanziazione di una classe

Interfaccia

- Servizi messi a disposizione da un oggetto e invocabili da altri oggetti
- Nota: non è modello ORIENTATO agli oggetti (mancano alcune caratteristiche quali l'ereditarietà).

Il progettista può modellare un'applicazione individuando GLI OGGETTI, ovvero i componenti architetturali e i processi.

4.6 Middleware

Insieme di componenti software che realizzano una Macchina Virtuale = servizi coerenti "erogati" da "unica" macchina. Tipologie di middleware:

1. Middleware generalizzato: strumenti di comunicazione, servizi di security, indirizzamento, sincronizzazione, accodamento
2. Middleware orientati a specifiche classi di servizio
   (a) Middleware per accesso a DB: interfacce di programmazione per l'accesso a BD da applicazioni indipendente da caratteristiche fisiche dei singoli sistemi di gestione dei dati;
   (b) Middleware per transazioni Es. Distributed Transaction Processing – DTP (di X/Open): specifica come processi diversi possono collaborare per attuare transazioni distribuite.

5 ERP

5.1 Definizione

Un ERP fornisce l'**integrazione di tutti i flussi informativi** nell'ambito di un'azienda. Ad esempio:

- Informazioni Finanza/Accounting
- Informazioni Human Resources (HR)
- Informazioni Clienti
- nato per Supply ChainManagement – esteso ad altre aree

Un'organizzazione con ERP è un'organizzazione process-oriented, i dati sono il core dell'azienda.

5.2 Come funziona?

Sistemi ERP sono sistemi software progettati per automatizzare e standardizzare TUTTI i Processi organizzativi di Business ("organizational business processes"). L'obiettivo è organizzare **tutte** le funzioni aziendali e condividere le informazioni con i partner. Il data sharing avviene così senza **né ridondanza né duplicazione**.

ERP indica **una suite di moduli applicativi integrati** che supportano una ampia gamma di processi di un'impresa:

- processi interni (ERP core)
  - moduli core intersettoriali :
    - moduli istituzionali (amministrazione, gestione risorse umane)
    - moduli direzionali (pianificazione strategica, la programmazione ed il controllo del budget ecc.)
    - moduli di gestione progetti & investimenti
    - portale aziendale.
  - moduli core settoriali
    - che supportano le attività primarie tipiche di un settore (p.e. il ciclo di trasformazione di una azienda manifatturiera)
- transazioni interaziendali verso fornitori (extended ERP)

5.3 Progetti ERP

Le attività sono focalizzate sulla "customizzazione" e sul "setting" (non sulla programmazione). C'è un alto impatto sull'organizzazione, alti costi e lunghi tempi di implementazione. Perché l'ERP ha avuto successo?

- Unico paradigma IT per tutti i clienti/fornitori
- Superamento dei limiti strutturali delle precedenti generazioni di packages e di sistemi hoc (custom)

Punti cruciali dell'ERP:

1. Coerenza dell' informazione e del data sharing
2. Sistema estensibile e modulare
3. Modello prescrittivo

5.4 Principali benefici dell'ERP

1. Trasparenza e capacità informativa aumentata
2. Maggiore capacità decisionale e velocità di adeguamento alle variazioni
3. Saturazione delle risorse e dei materiali
4. Riduzione delle giacenze di magazzino
5. Miglior servizio e migliore informazione al cliente

5.5 Scelta della soluzione ERP

Nel momento in cui una azienda sceglie di affidarsi ad una soluzione ERP deve selezionare i pacchetti necessari. Per fare questo, deve:

- Individuare indicatori che consentano di descrivere e classificare i pacchetti sw disponibili sul mercato
- Evidenziare e analizzare eventuali correlazioni tra tali indicatori
- Utilizzare gli indicatori in fase di studio di fattibilità in una metodologia orientata ai costi che supporta la scelta di un sistema sw
- Mettere in evidenza eventuali criticità dovute al particolare contesto nel quale si opera.

5.6 Variabili descrittive

Le variabili che differenziano le diverse soluzioni di mercato e ne supportano la scelta in fase di **studio di fattibilità** sono:

- Variabili funzionali: funzionalità supportate dai diversi moduli che compongono il sistema sw in esame.
- Variabili architetturali: aspetti tecnologici critici al momento della scelta del sistema sw e soprattutto per la sua scalabilità nel tempo.

- Costi della soluzione: vengono considerati
  - i costi delle licenze,
  - i costi hardware,
  - i costi di progetto legati alle risorse umane,
  - i costi di manutenzione.

5.6.1 Indicatori funzionali

- Grado di completezza del programma
- Personalizzabilità
  - Con personalizzazioni standard (moduli aggiuntivi)
  - Con personalizzazioni dinamiche (ad hoc)

5.6.2 Indicatori architetturali

- Scalabilità dell'infrastruttura: fornire indicazioni a supporto della scelta di un sistema hardware scalabile nel tempo
- Grado di interoperabilità della soluzione: comprendere quanto l'applicativo possa essere considerato un'entità aperta verso il mondo esterno
- Livello di sicurezza del sistema

5.7 Paradigma Software as a Service

Si intende tutto un insieme di applicazioni web-native messe a disposizione dai proprietari ai propri clienti attraverso Internet. A differenza di ASP **molti client usano la stessa istanza**.

6 BPMN

6.1 Definizione

BP è un insieme di attività eseguite in coordinamento nell'ambito di un contesto organizzativo e tecnologico. Le attività concorrono ad un **business goal**. BPM(amangement) comprende concetti, metodi e tecniche che supportano il progetto, l'amministrazione, la configurazione, la messa in opera (enactment) e l'analisi di BP.

6.2 Simbologia, approfondimenti, esercitazioni

Si invita a vedere le slide dell'esercitatore

7 Sicurezza

7.1 Introduzione

La sicurezza si può dividere in tre tipi: fisica, logica e organizzativa. Circa il 90% delle intrusioni in rete non viene scoperta e gli attacchi hanno successo nell'80% dei casi. Si ha quindi una intrinseca debolezza dei sistemi: **è necessario predisporre opportune "Politiche di sicurezza": strumenti tecnici, regole organizzative e norme amministrative**.

C'è quindi la necessità di condividere e cooperare su larga scala ma nel contempo di preservare sistemi informativi e basi di dati preesistenti.

7.2 Minacce

Le minacce ad un sistema informativo possono essere di tre tipi

- **Fisiche**: es. furti, danneggiamenti intenzionali, eventi accidentali (rottura impianto di condizionamento) o da disastri naturali (inondazioni, incendi, terremoti)
- **Logiche**: es. sottrazione di dati (numeri di carta di credito memorizzati), creazione di punti di accesso nascosti al sistema (uso illegale delle risorse di calcolo e di comunicazione)
- **Accidentali**: es. errori di configurazione, malfunzionamenti di programmi, errori di data entry

Da chi difendersi quindi? Da attacchi di terzi, dal gestore di rete (che ha il controllo di tutte le basi di dati) e dall'accesso del personale non autorizzato ma anche dagli utenti legittimi (spionaggio industriale).

7.3 Proprietà di Sicurezza

- **Integrità**: certezza che le informazioni non vengano alterate da agenti non autorizzati (aggiunta, cancellazione o modifica)
- **Autenticità**: le informazioni sono integre e contengono al loro interno anche elementi che permettono di identificarne il creatore. Non ripudio
- **Autenticazione**: ogni agente viene identificato prima di poter interagire col sistema
- **Riservatezza**: le informazioni possono essere lette solo dagli agenti autorizzati
- **Disponibilità**: un sistema deve essere in grado di fornire i servizi per cui è stato progettato quando richiesti dagli utenti

7.4 Tipologie di attacco

- **Cause:** debolezze tecniche di progetto o di implementazione, procedure di gestione inadeguate o inapplicate
- **Sniffing:** vengono catturati pacchetti di rete non destinati al nodo che effettua l'attacco (lettura dati, ricerca di password).
  - **Soluzione:** crittografia
- **Address spoofing:** generazione di pacchetti di rete contenenti l'indicazione di un falso mittente.
  - **Soluzione**: meccanismi di autenticazione forte
- **Data spoofing:** vengono creati dati falsi oppure i dati vengono modificati
- **Hijacking:** è una forma molto sofisticata di data spoofing che consiste nel "dirottamento" di un canale virtuale
- **Denial-of-service(DoS):** rende non disponibile un servizio, tenendolo impegnato in una serie di operazioni inutili o bloccandone completamente l'attività (DDoS). Di difficile soluzione (a posteriori)
- **Infezioni informatiche:**
  - **Trojan:** programma, apparentemente utile, che nasconde invece una porzione di codice atta a realizzare azioni indesiderate o distruttive
  - **Virus:** presenta tutte le caratteristiche dei trojan, cui aggiunge la capacità di replicarsi e propagarsi
  - **Worm:** programma che utilizza le tecniche di connessione in rete, ed i relativi protocolli, per trasferirsi da sistema a sistema
- **Shadow Server:** creazione di un falso server in grado di sostituirsi a quello vero per fornire informazioni sbagliate oppure per catturare dati introdotti dagli utenti (Sniffing+Spoofing +DoS, DNS Spoofing)

7.5 Strategie di attacco

- **Attacchi da sistemi ponte**, ovvero da sistemi **già attaccati in precedenza**. Gli attacchi sfruttano problemi di configurazioni o debolezze **note** dei sistemi.
- **Social engineering** (es. corruzione del personale autorizzato) - detto anche livello non tecnico si è rivelato il punto più debole

Non esiste sistema sicuro in assoluto, dipende dal livello dell'informazione che si vuole proteggere.

7.6 Tecniche base e principi di crittografia

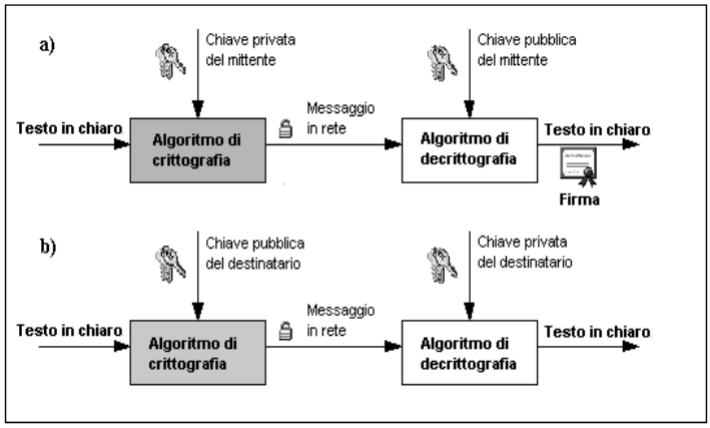
Crittografare: codificare l'informazione in modo da renderla intellegibile solo a chi possiede una (o più) CHIAVE(I). Ci si avvale sempre di algoritmi noti e resi pubblici. La chiave deve essere scelta tra un **vastissimo numero di combinazioni e CAMBIATA SPESSO.**

7.6.1 Algoritmi di crittografia

Sono a chiave **simmetrica** e a chiave **asimmetrica** (esiste una coppia di chiavi privata e pubblica). Esempi semplici di algoritmi a chiave simmetrica sono sostituzione e trasposizione.

- Crittografia **simmetrica**
  - Vantaggi
    - è molto veloce
  - Svantaggi
    - Le parti per comunicare devono essersi incontrate
    - Comunicazioni uno-molti: come tenere segreta la chiave?
    - Una volta che una chiave è compromessa, tutti i messaggi successivi sono compromessi
  - Maggiore è la lunghezza della chiave, maggiore è il livello di protezione offerto dall'algoritmo. Applicazioni commerciali oggi anche a 1024 bit, prima usate solo per applicazioni ad alta sicurezza.

- Crittografia **asimmetrica**



Uso di una funzione f semplice, la cui inversa è complessa. Vengono utilizzate due funzioni (chiave pubblica e chiave privata) per crittare/decrittare rispettivamente. Data una coppia di chiavi asimmetriche, **una chiave viene resa pubblica mentre l'altra viene assegnata ad una singola entità che**

deve tenerla riservata. Richiede l'uso di due chiavi diverse per cifrare e decifrare i dati; se una chiave è stata usata per cifrare i dati allora per decifrarli occorre usare solo ed esclusivamente l'altra chiave che fa parte della coppia. E' una procedura **lenta**: viene utilizzata solamente per passare le chiavi.

7.7 Impronta

I dati cifrati da soli non sono sufficienti: non possono essere letti da chi non conosce la chiave di decifratura, ma ciò non è sufficiente a garantire la protezione di questi dati. Anche se le informazioni sono cifrate, **è sempre possibile alterarle e quindi renderle inutilizzabili** anche da parte dei legittimi destinatari. E' importante disporre di opportune tecniche per evidenziare che un dato è stato manipolato: l'impronta dei dati (digest) può essere utilizzata.

7.7.1 Firma digitale

Come funziona? Viene creata una **firma con la chiave privata** del firmatario basata sul digest. Questa firma viene poi inviata al destinatario che **attraverso la chiave pubblica del firmatario potrà verificare** che la firma è valida. Il destinatario avrà inoltre, separatamente, ricevuto il documento elettronico - **confrontando i digest potrà capire se quel documento è effettivamente quello inviato.**

La firma digitale richiede un tempo relativamente lungo (per questo viene fatta sul digest e non sul documento). Dipende non solo dal firmatario, ma **anche dai dati** che si sta firmando. Non solo identifica chi ha generato i dati ma **dimostra anche che i dati non sono stati modificati dopo essere stati firmati**.

7.8 Gestione delle chiavi crittografiche

Per applicare gli algoritmi di crittografia (talvolta anche quelli di digest) occorre che le parti interessate posseggano e condividano le chiavi. Questo problema è detto Gestione delle Chiavi e richiede la soluzione di vari sottoproblemi:

- Generazione delle chiavi
  - Deve essere fatta direttamente da chi effettuerà le operazioni crittografiche
- Conservazione delle chiavi
  - se le chiavi sono note agli attaccanti, tutte le altre misure di sicurezza possono essere aggirate. Le chiavi vengono quindi memorizzate o in file cifrati protetti da password o, in alternativa, su dispositivi elettronici.
- Identificazione del possessore di una chiave

- Scambio delle chiavi ("key exchange" o "key distribution")
  - Tecnica OOB (Out-Of-Band): chiave distribuita tramite canale diverso da quello su cui transitano i dati
  - In alternativa, se il destinatario dei dati cifrati dispone di una coppia di chiavi asimmetriche, **il mittente può inviargli la chiave segreta cifrandola con la chiave pubblica del destinatario**.
  - Se lo scambio è di chiavi pubbliche, **non è importante la loro segretezza ma solo stabilire l'identità del possessore**. Viene quindi usato un **certificato** che lega una persona ad una chiave pubblica. Il certificato ha scadenza temporale ed è revocabile sia dall'utente che dall'emettitore.
    * Per garantire che il certificato non possa essere alterato, esso viene protetto con la firma digitale dell'"autorità di certificazione" che lo ha emesso. Esiste una infrastruttura a supporto delle chiavi asimmetriche, "infrastruttura a chiave pubblica" o PKI.

7.9 Sicurezza nelle basi di dati

7.9.1 Controlli di sicurezza

- **Controllo di flusso**: controllo da oggetto X su oggetto Y quando sono eseguite READ e WRITE
- **Controlli di inferenza**
- **Controlli di accesso**

7.9.2 Regole di accesso

Concetto di **Ownership** e funzione di controllo. Gli utenti sono suddivisi in **gruppi** o **classi** che hanno accesso a diverse transazioni di applicativi.

7.9.3 Modello a 4 componenti constraint:

Espresso mediante Protection View (protezione delle viste)

- il modello fornisce:
  - controllo dell'accesso dipendente dal nome
  - controllo dell'accesso dipendente dal contenuto
- Ancora non fornisce:
  - controlli su amministrazione del set di regole di accesso
  - delega dei privilegi di accesso

Vengono create regole per limitare gli accessi a seconda che il sistema sia aperto o chiuso. In un sistema chiuso, **se esiste una regola d'accesso, l'accesso è consentito altrimenti è negato** (nel sistema aperto vale l'opposto).

7.9.4 Controlli d'accesso

- Di tipo **discrezionale**
  - Gli utenti amministrano i dati che **possiedono**. Su questi dati possono autorizzare il tipo di accesso, autorizzare altri utenti e definire accessi selettivi.
- Di tipo **mandatorio**
  - Per basi di dati con dati sensitibili (governative, militari, sanitarie, ..) DAC non è sufficiente. Quindi vengono utilizzate politiche basate su classificazione di dati e utenti: MAC (Mandatory Access Control)

7.9.5 MAC

Vengono classificati tutti i dati secondo un livello di sicurezza e tutti i soggetti. I meccanismi di sicurezza **devono garantire che tutti i soggetti abbiano accesso solo ai dati per cui possiedono la clearance appropriata**.

Classificazione MAC: **Unclassified, Confidential, Secret, Top Secret**

7.9.6 Tipi di controllo d'accesso alle basi di dati

L'accesso può essere basato sul **nome** (a seconda della tipologia è possibile vedere solamente diversi attributi di una tabella) o sul **contenuto** (a seconda della tipologia è possibile vedere solamente diverse tuple di una tabella). Il tipo di accesso può essere poi sia in lettura che in scrittura.