Reti e Laboratorio III
Modulo Laboratorio III
AA. 2023-2024

docente: Laura Ricci

laura.ricci@unipi.it

Lezione 12
Conclusioni
12/12/2023

SSDP SIMPLE SERVICE DISCOVERY PROTOCOL (SSDP)

- su una rete locale attivi molti protocolli che usano il multicast
- SSDP attivo sulla porta 1900 associato all'indirizzo di multicast 239.255.255.250
- un protocollo di discovery usato per scoprire quali servizi sono disponibili in una rete
- ogni servizio è definito mediante specifiche UPnP
- il server SSDP invia pacchetti di advertisement sul gruppo di multicast
- formato di un pacchetto di advertisement

```
HTTP/1.1 200 OK
CACHE-CONTROL: max-age=120
ST: urn:schemas-upnp-org:device:WANDevice:1
USN: uuid:fc4ec57e-b051-11db-88f8-
0060085db3f6::urn:schemas-upnp-org:device:WANDevice:1
EXT:
SERVER: Net-OS 5.xx UPnP/1.0
LOCATION: http://192.168.0.1:2048/etc/linuxigd/gatedesc.xml
```

SSDP SIMPLE SERVICE DISCOVERY PROTOCOL (SSDP)

Messaggio di advertisement

- USN, Unique Service Name
  - UUID (Universally Unique Identifier) che identifica una device o un servizio
- LOCATION
  - punta ad una URL
  - a quella URL l'utente può trovare una descrizione XML delle capability del servizio

```
HTTP/1.1 200 OK
CACHE-CONTROL: max-age=120
ST: urn:schemas-upnp-org:device:WANDevice:1
USN: uuid:fc4ec57e-b051-11db-88f8-
0060085db3f6::urn:schemas-upnp-org:device:WANDevice:1
EXT:
SERVER: Net-OS 5.xx UPnP/1.0
LOCATION: http://192.168.0.1:2048/etc/linuxigd/gatedesc.xml
```

SNIFFING SSDP IN JAVA

```
import java.io.*; import java.net.*;
public class MulticastSniffer {
 public static void main(String[] args) {
    InetAddress group = null;
    int port = 0;
    // read the address from the command line
    try {
      group = InetAddress.getByName(args[0]);
      port = Integer.parseInt(args[1]);
    } catch (ArrayIndexOutOfBoundsException | NumberFormatException |
                                                    UnknownHostException ex)
        {
           System.err.println( "Usage: java MulticastSniffer multicast_address
                                                    port");
           System.exit(1); }
    MulticastSocket ms = null;
```

SNIFFING SSDP IN JAVA

```
try {ms = new MulticastSocket(port);
     ms.joinGroup(group);
     byte[] buffer = new byte[8192];
     while (true) {
        DatagramPacket dp = new DatagramPacket(buffer, buffer.length);
        ms.receive(dp);
        String s = new String(dp.getData(), "8859_1");
        System.out.println(s); }
 } catch (IOException ex) { System.err.println(ex);
 } finally {
   if (ms != null) {
     try {
       ms.leaveGroup(group);
       ms.close(); } catch (IOException ex) { } } } } }
```

SNIFFING SSDP IN JAVA: OUTPUT GENERATO

```
$ Java MulticastSniffer 239.255.255.250 1900

NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1801
NTS: ssdp:alive
LOCATION: http://192.168.1.1:49152/gKgq6lu34h/wps_device.xml
SERVER: Unspecified, UPnP/1.0, Unspecified
NT: urn:schemas-wifialliance-org:service:WFAWLANConfig:1
USN: uuid:03fef68b-319f-5ea7-80a8-2aea5bb7e216::urn:schemas-
wifialliance-org:service:WFAWLANConfig:1
…….
```

COLLECTIONI ED ITERATORI

- le `Collection JAVA` supportano diversi tipi di iteratori
  - si distinguono riguardo al comportamento di una collezione in presenza di "concurrent modification"
  - cosa accade quando la collezione viene modificata, mentre un iteratore la sta scorrendo, e questa modifica arriva "dall'esterno" dell'iteratore?
- fail-fast
  - se c'è una modifica strutturale (inserzione, rimozione, aggiornamento),

    dopo che l'iteratore è stato creato, l'iteratore la rileva e solleva una `ConcurrentModificationException`
  - fallimento immediato dell'operatore, per evitare comportamenti non deterministici
  - la maggior parte delle collezioni "non-concurrenti" sono fail-fast

    `Vector`, `ArrayList`, `HashMap`, ed altre....

FAIL FAST: HASHMAP

```
import java.util.HashMap;  import java.util.Iterator;import java.util.Map;
public class FailFastExample {
    public static void main(String[] args)
     { Map<String, String> cityCode = new HashMap<String, String>();
       cityCode.put("Delhi", "India");
       cityCode.put("Moscow", "Russia");
       cityCode.put("New York", "USA");
       Iterator iterator = cityCode.keySet().iterator();
       while (iterator.hasNext()) {
          System.out.println(cityCode.get(iterator.next()));
          cityCode.put("Istanbul", "Turkey"); }}}
```

```
India
Exception in thread "main"
java.util.ConcurrentModificationException
at java.util.HashMap$HashIterator.nextNode(Unknown Source)
at java.util.HashMap$KeyIterator.next(Unknown Source)
at FailFastExample.main(FailFastExample.java:19)
```

COLLECTIONI ED ITERATORI

- fail-safe ("snapshot") introdotti in `JAVA 1.5` con le concurrent collections
  - creano una copia della collezione, al momento della creazione dell'iteratore e lavorano su questa copia
  - non sollevano `ConcurrentModificationException`
  - l'iteratore accede ad una versione non aggiornata della collezione
  - `CopyOnWriteArrayList`
- weakly consistent introdotti in `JAVA 1.5` con le concurrent collections
  - l'iteratore e modifiche operano sulla stessa copia
  - no `ConcurrentModificationException`, comportamento fail-safe
  - l'iteratore considera gli elementi che esistevano al momento della costruzione dell'iteratore e può riflettere le modifiche che sono avvenute dopo la costruzione dell'iteratore, anche se non è garantito
  - `ConcurrentHashMap`, ...

WEAK CONSISTENCY: CONCURRENT HASH MAP

da `JavaDocs;`

"The view's iterator is a "weakly consistent" iterator that will never throw ConcurrentModificationException, and guarantees to traverse elements as they existed upon construction of the iterator, and may (but is not guaranteed to) reflect any modifications subsequent to construction."

- l'iteratore
  - non clona la struttura al momento della creazione
  - la collezione può catturare le modifiche effettute sulla collezione dopo la sua creazione
    - non solleva `ConcurrentModificationException`
- alcuni metodi, `size()` e `isEmpty()`
  - possono restituire un valore "approssimato"
  - "weakly consistent behaviour"

UN ITERATORE WEAKLY CONSISTENT

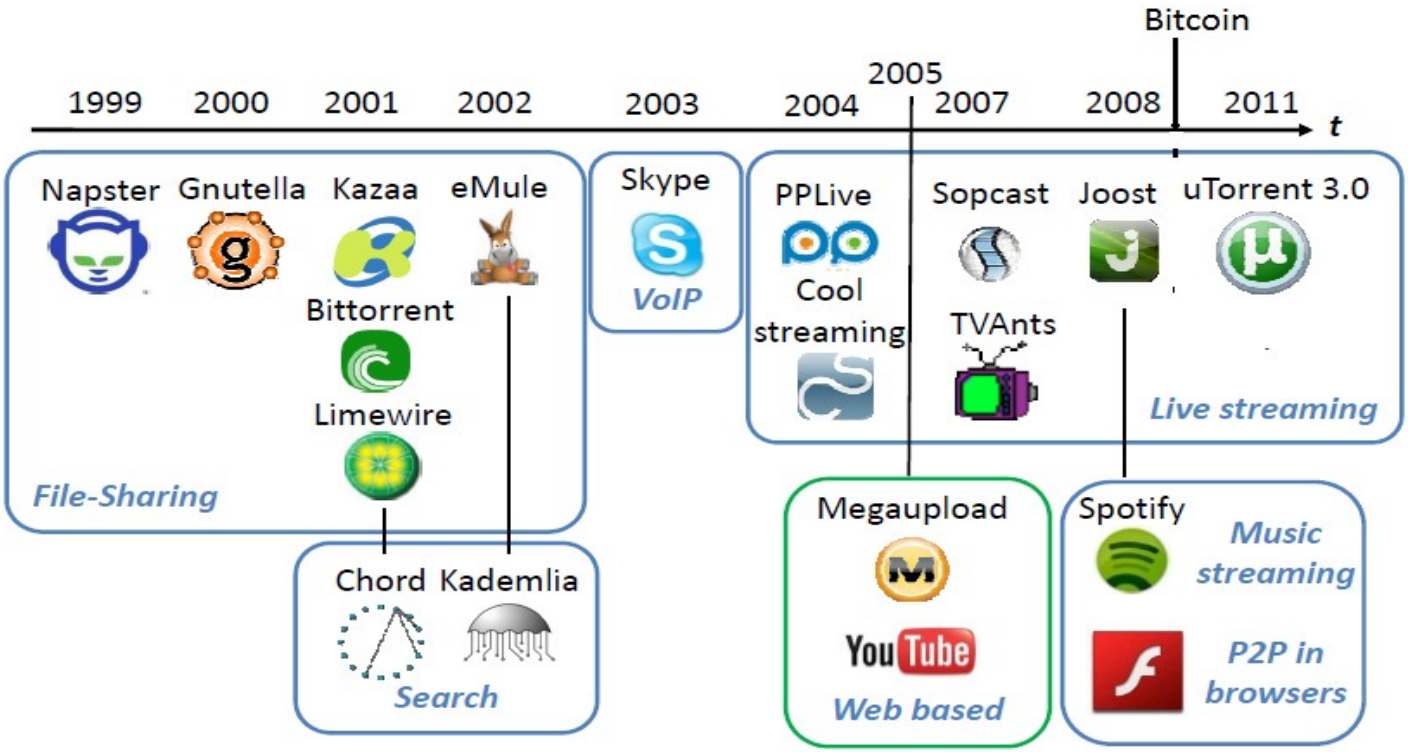
```
import java.util.concurrent.ConcurrentHashMap;
import java.util.Iterator;
public class FailSafeItr {
    public static void main(String[] args)
    {ConcurrentHashMap<String, Integer> map = new ConcurrentHashMap<String, Integer>();
     map.put("ONE", 1);
     map.put("TWO", 2);
     map.put("THREE", 3);
     map.put("FOUR", 4);
     Iterator <String> it = map.keySet().iterator();
     while (it.hasNext()) {
            String key = (String)it.next();
            System.out.println(key + " : " + map.get(key));
            // Notice, it has not created separate copy
            // It will print 7
            map.put("SEVEN", 7); }}}
       // the program prints ONE : 1 FOUR : 4 TWO : 2 THREE : 3 SEVEN : 7
```

FAIL SAFE: COPYONWRITEARRAYLIST

- come risulta evidente dal nome, effettua una copia dell'array tutte le volte che viene effettuata una operazione di modifica (add, set, etc..)
- "snapshot style iterator": usa un riferimento ad una copia dello stato dell'array nel momento in cui l'iteratore è creato
  - l'array riferito non viene mai modificato durante la vita dell'iteratore: l'iteratore non cattura le modifiche effettuate dopo la sua creazione
  - thread-safe: ogni thread lavora su una propria copia
  - fail safe: non solleva `ConcurrentModificationException`
- operazione di copia molto costosa
  - è adatto quando ci sono più accessi in lettura che modifiche

IL MODELLO PEER TO PEER

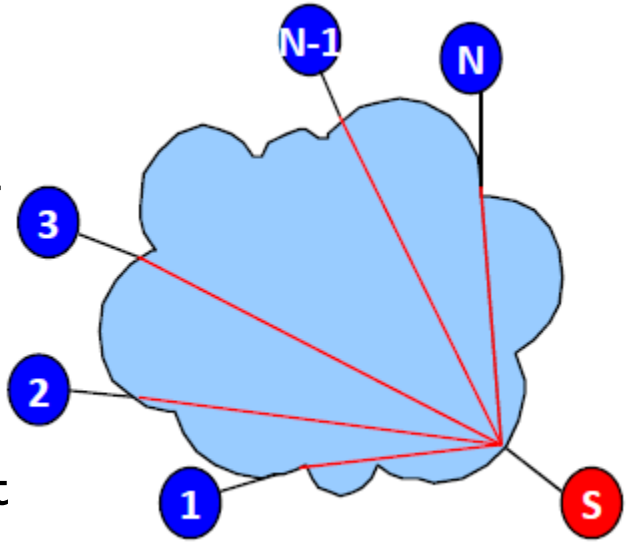
- in questo corso abbiamo analizzato il modello client server, come base per costruire applicazioni di rete
- molte applicazioni sono sviluppate secondo un modello alternativo: il modello peer to peer: nato nel 2000 per applicazioni di file-sharing



IL PARADIGMA CLIENT SERVER

- in esecuzione sugli end host
- comportamento on/off
- utilizza servizi
- inoltra richieste
- nessuna interazione tra client
- deve conoscere un riferimento al servizio

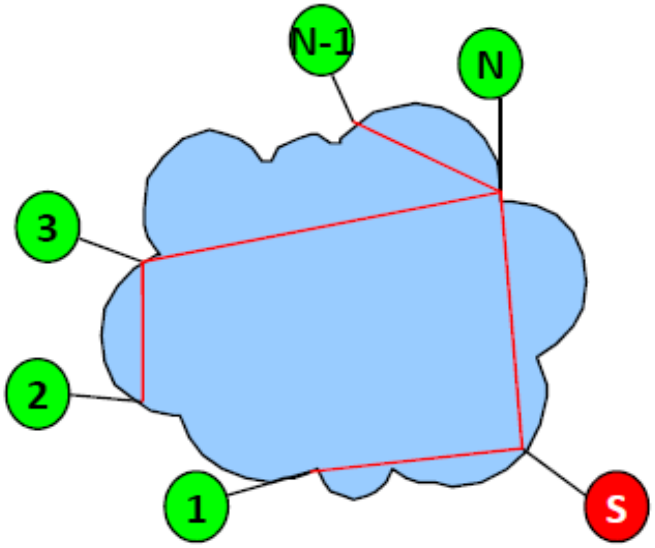
- in esecuzione su un host dedicato
- "always on"
- fornisce servizi
- riceve richieste dai client
- soddisfa le richieste dell'utente
- deve avere un IP fisso (o DNS name)



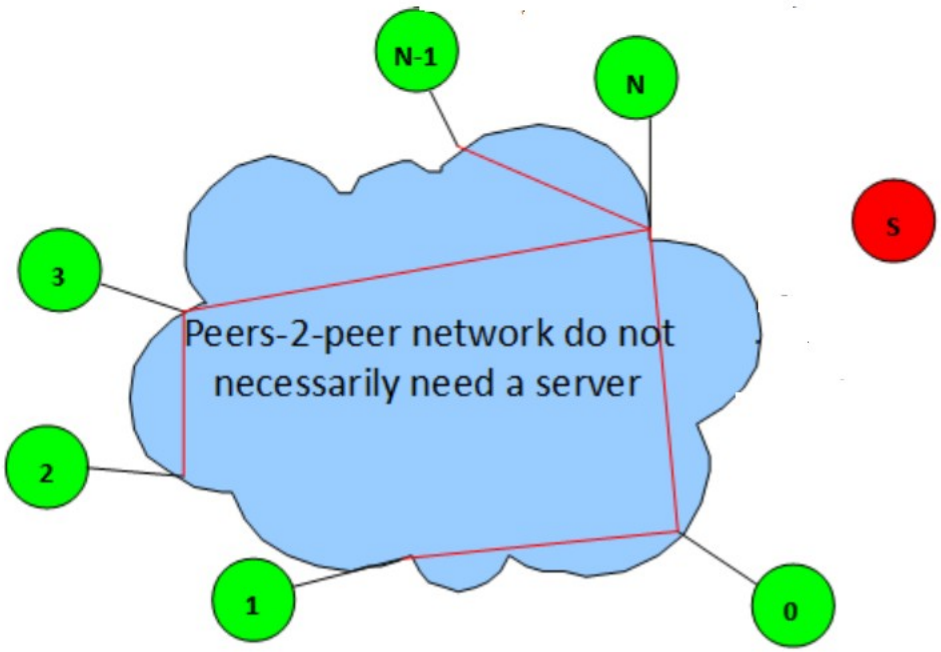
IL PARADIGMA P2P

Peer

- in esecuzione sugli end-hosts
- produttori e consumatori di servizi
- comportamento on/off
- alto livello di dinamicità (churn)
- necessaria una fase di bootstrap
- necessari meccanismi per la scoperta di peer
- comunicano tra di loro
- necessari protocolli a livello applicazione per
  - evitare il fenomeno dei `free riders`
  - incentivare partecipazione e collaborazione



IL PARADIGMA P2P



notare che:

- un server è sempre presente, ma serve solo nella fase di bootstrap
- i server non sono necessari per la condivisione delle risorse

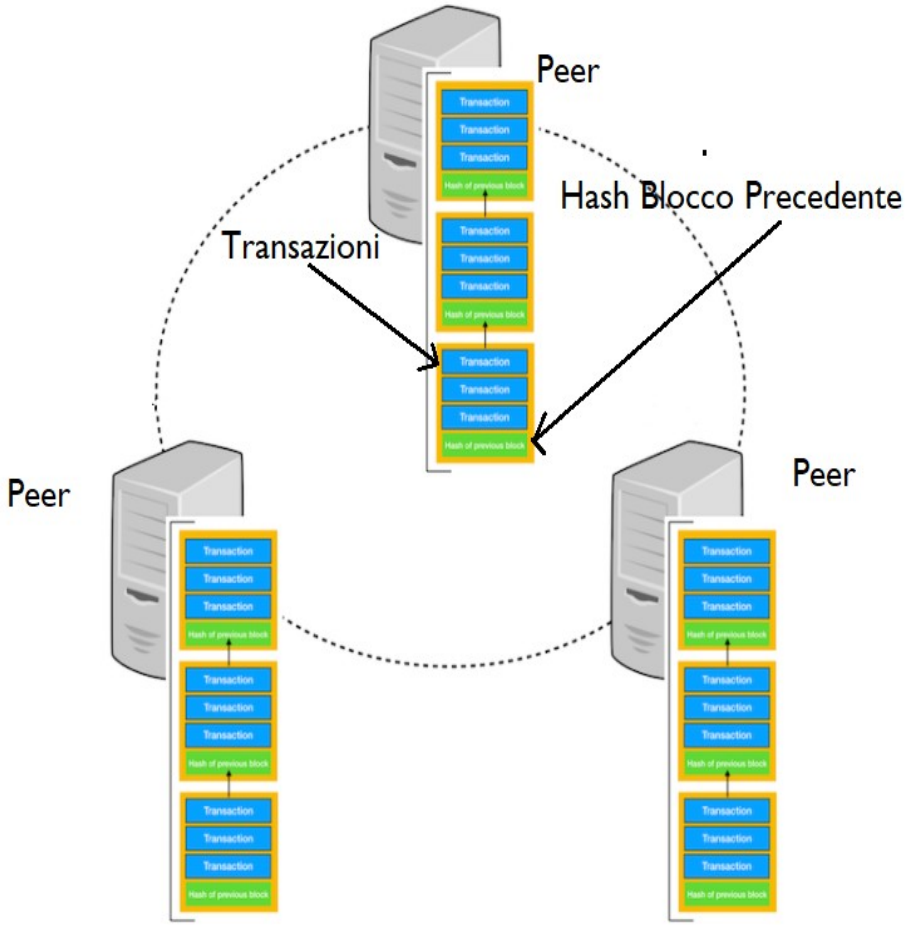
P2P: LE APPLICAZIONI

- P2P file sharing
  - file sharing: light weight/ best effort
  - la persistenza e la sicurezza non sono l'obiettivo principale
  - l'anonimato è importante
  - applicazione
    - Napster
    - Gnutella, KaZaa
    - eMule
    - BitTorrent
- P2P Media Streaming
  - Skype (first versions)
- Cyptocurrencies and blockchains
- Distributed file System: Internet Planetary File System (IPFS)

BLOCKCHAIN IN BREVE

- database distribuito e replicato sui nodi di un sistema peer to peer (P2P)
  - un insieme di blocchi collegati mediante puntatori hash
  - in ogni blocco è presente l'hash blocco precedente
  - ogni peer possiede una copia consistente dell'intero database

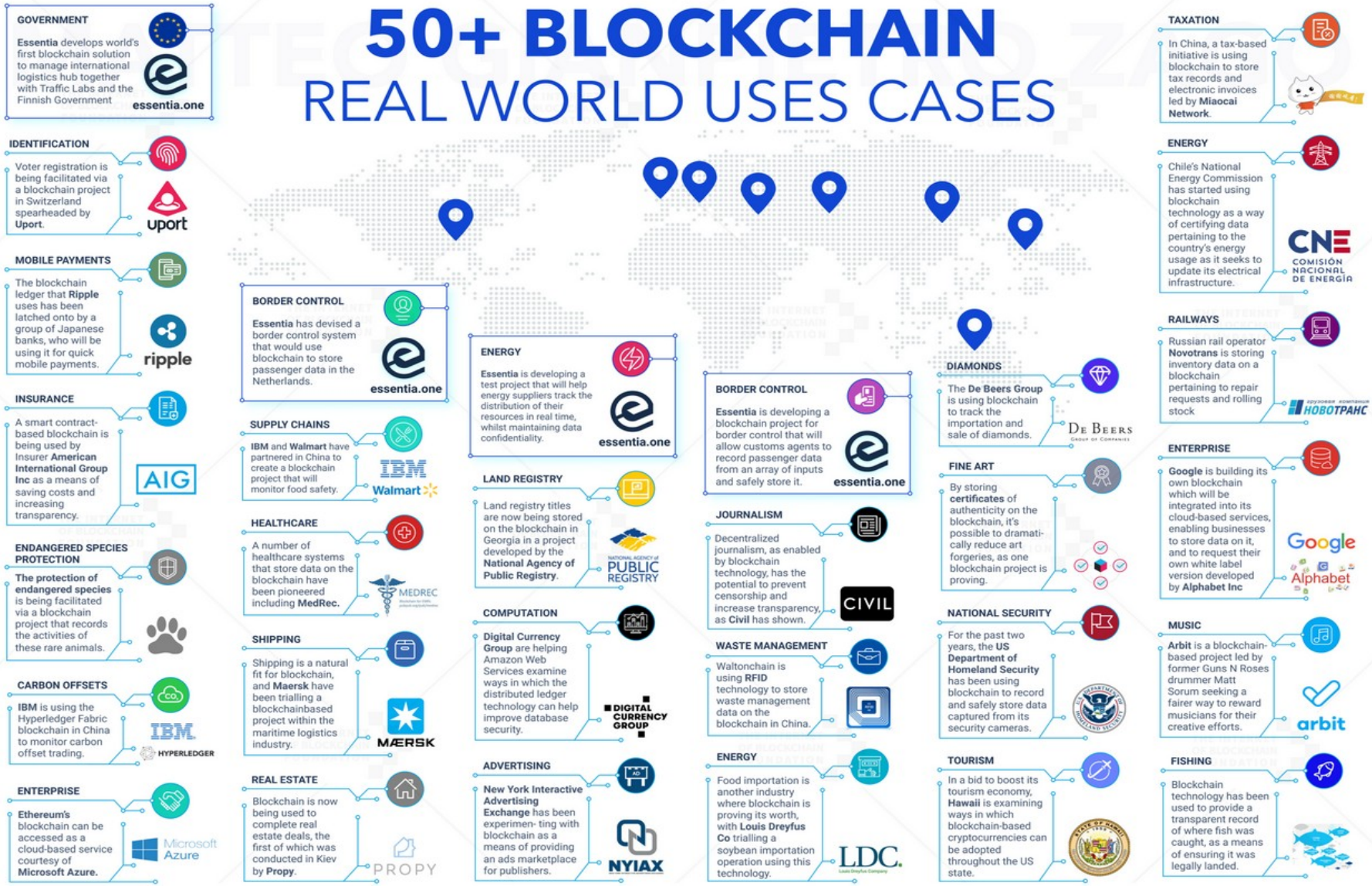
- operazioni
  - append only: accodare progressivamente registrazioni organizzate in blocchi
  - leggere il contenuto di una qualsiasi registrazione



LE BLOCKCHAIN: EVOLUZIONE

L'ECOSISTEMA DI BITCOIN

- non solo blockchain, ma diversi attori coinvolti
  - `Exchangers`, `Wallets`, `NFT Marketplaces`
- una vera e propria economia basata su cryptocurrencies


50+ BLOCKCHAIN
REAL WORLD USES CASES
GOVERNMENT
Essentia develops world's first blockchain solution to manage international logistics hub together with Traffic Labs and the Finnish Government
essentia.one
IDENTIFICATION
Voter registration is being facilitated via a blockchain project in Switzerland spearheaded by Uport.
uport
MOBILE PAYMENTS
The blockchain ledger that Ripple uses has been latched onto by a group of Japanese banks, who will be using it for quick mobile payments.
ripple
INSURANCE
A smart contract-based blockchain is being used by Insurer American International Group Inc as a means of saving costs and increasing transparency.
AIG
ENDANGERED SPECIES PROTECTION
The protection of endangered species is being facilitated via a blockchain project that records the activities of these rare animals.
CARBON OFFSETS
IBM is using the Hyperledger Fabric blockchain in China to monitor carbon offset trading.
IBM
HYPERLEDGER
ENTERPRISE
Ethereum's blockchain can be accessed as a cloud-based service courtesy of Microsoft Azure.
Microsoft Azure
BORDER CONTROL
Essentia has devised a border control system that would use blockchain to store passenger data in the Netherlands.
essentia.one
SUPPLY CHAINS
IBM and Walmart have partnered in China to create a blockchain project that will monitor food safety.
IBM
Walmart
HEALTHCARE
A number of healthcare systems that store data on the blockchain have been pioneered including MedRec.
MEDREC
SHIPPING
Shipping is a natural fit for blockchain, and Maersk have been trialling a blockchainbased project within the maritime logistics industry.
MÆRSK
REAL ESTATE
Blockchain is now being used to complete real estate deals, the first of which was conducted in Kiev by Propy.
PROPY
ENERGY
Essentia is developing a test project that will help energy suppliers track the distribution of their resources in real time, whilst maintaining data confidentiality.
essentia.one
LAND REGISTRY
Land registry titles are now being stored on the blockchain in Georgia in a project developed by the National Agency of Public Registry.
PUBLIC REGISTRY
COMPUTATION
Digital Currency Group are helping Amazon Web Services examine ways in which the distributed ledger technology can help improve database security.
DIGITAL CURRENCY GROUP
ADVERTISING
New York Interactive Advertising Exchange has been experimen- ting with blockchain as a means of providing an ads marketplace for publishers.
NYIAX
BORDER CONTROL
Essentia is developing a blockchain project for border control that will allow customs agents to record passenger data from an array of inputs and safely store it.
essentia.one
JOURNALISM
Decentralized journalism, as enabled by blockchain technology, has the potential to prevent censorship and increase transparency, as Civil has shown.
CIVIL
WASTE MANAGEMENT
Waltonchain is using RFID technology to store waste management data on the blockchain in China.
ENERGY
Food importation is another industry where blockchain is proving its worth, with Louis Dreyfus Co trialling a soybean importation operation using this technology.
LDC.
DIAMONDS
The De Beers Group is using blockchain to track the importation and sale of diamonds.
DE BEERS
FINE ART
By storing certificates of authenticity on the blockchain, it's possible to dramatically reduce art forgeries, as one blockchain project is proving.
NATIONAL SECURITY
For the past two years, the US Department of Homeland Security has been using blockchain to record and safely store data captured from its security cameras.
TOURISM
In a bid to boost its tourism economy, Hawaii is examining ways in which blockchain-based cryptocurrencies can be adopted throughout the US state.
TAXATION
In China, a tax-based initiative is using blockchain to store tax records and electronic invoices led by Miaocai Network.
ENERGY
Chile's National Energy Commission has started using blockchain technology as a way of certifying data pertaining to the country's energy usage as it seeks to update its electrical infrastructure.
CNE
COMISIÓN NACIONAL DE ENERGÍA
RAILWAYS
Russian rail operator Novotrans is storing inventory data on a blockchain pertaining to repair requests and rolling stock
НОВОТРАНС
ENTERPRISE
Google is building its own blockchain which will be integrated into its cloud-based services, enabling businesses to store data on it, and to request their own white label version developed by Alphabet Inc
Google
Alphabet
MUSIC
Arbit is a blockchain-based project led by former Guns N Roses drummer Matt Sorum seeking a fairer way to reward musicians for their creative efforts.
arbit
FISHING
Blockchain technology has been used to provide a transparent record of where fish was caught, as a means of ensuring it was legally landed.

P2P FILE SYSTEM: IPFS

Your file, and all of the **blocks within it**, is given a **unique fingerprint** called a **cryptographic hash**.

IPFS **removes duplications** across the network.

Each **network node** stores only content it is interested in, plus some indexing information that helps figure out which node is storing what.

When you **look up a file** to view or download, you're asking the network to find the nodes that are storing the content behind that file's hash.

You don't need to remember the hash, though — every file can be found by **human-readable names** using a decentralized naming system called **IPNS**.

P2P FILE SYSTEM: IPFS

- IPFS: integrazione file system distribuito con blockchain
- dati di grande dimensione memorizzati su IPFS, hash dei dati su blockchain
- implementato in LibP2P
  - una libreria per lo sviluppo di applicazioni P2P
  - nuove tecnologie: Multiprotocol Project
  - Distributed Hash Table: Kademlia

LABORATORIO DI WEB SCRAPING

- complementare Laurea Triennale in Informatica

Programma

- *Python*
  - introduzione mediante esempi
- *Reperimento di Dati*
  - API: user authentication, costruzione di queries, esempi: Google Big Queries
  - Web Scarping: pagine statiche : BeautifulSoup. Pagine dinamiche : Selenium
- *Analisi dei Dati*
  - Dati Tabellari, la libreria Pandas: Selezione di elementi per indice e per condizioni, Group_by, Merging e Concatenating, Pivoting
  - Visualizing: Matplotlib
  - Data Clustering
- *Dati Relazionali*
  - la libreria NetworkX: graph models: random graphs, scale free networks, small worlds.
  - proprietà caratteristiche: diametro, centrality, clustering coefficient,communities
- Casi d'uso: Blockchain
  - struttura delle transazioni (Bitcoin, Ethereum)
  - blockchain exploreres
  - NFT Markets
  - Grafi di transazioni

PEER TO PEER E BLOCKCHAINS

- corso fondamentale, Laurea Magistrale in Informatica, Curriculum ICT

Programma

- Introduzione al corso
- Sistemi p2p: concetti generali, unstructured overlays: Flooding, Random Walks, Epidemic Diffusion, structured overlays: Distributed Hash Tables (DHT), Routing on a DHT, IPFS
- Sistemi p2p, Applicazioni: BitTorrent, una Content Distribution Network,uso della DHT Kademlia
- Blockchains: concetti di base, strumenti crittografici di base (digital signatures, cryptographic hash, Zero Knowledge,.,..), structure dati: Merkle trees, Merkle Patricia tries
- Bitcoin
  - struttura delle transazioni e dei blocchi. Meccanismo di Mining : Proof of Works, ricompense
  - attacchi, 51%, double spending
  - tracciabilità e mixing. Struttura della rete P2P
- Ethereum:
  - smart contract, gas. Meccanismo di consenso: dalla PoW alla PoS
  - Solidity, programmazione di smart contracts. Smart contract security e attacks
- Applicazioni della tecnologia dei Distributed Ledgers: tokens (fungibili e non fungibili), Self Sovereign Identity, Supply-chains
- Meccanismi per aumentare la scalabilità: Zero Knowledge rollup e optimistic rollup, side chain, protocolli inter-chain, channel network (Lightning Network)

TESI DI LAUREA TRIENNALI E MAGISTRALI

- applicazione di tecniche crittografiche in collaborazione con la Prof. Bernasconi
  - zero-knowledge
  - authenticated data structures
- analisi di blockchain transactions
  - scraping di marketplace: esempio Fragment per compravendita NFT legati a Telegram
  - analisi di smart contracts
- applicazioni della blockchain
  - identità digitale: Self Sovereign Identity (SSI)
  - collegamento blockchain/mondi virtuali (verso il metaverso), Decentraland
- tecnologie legate a blockchain
  - cross-chain system, Polkadot