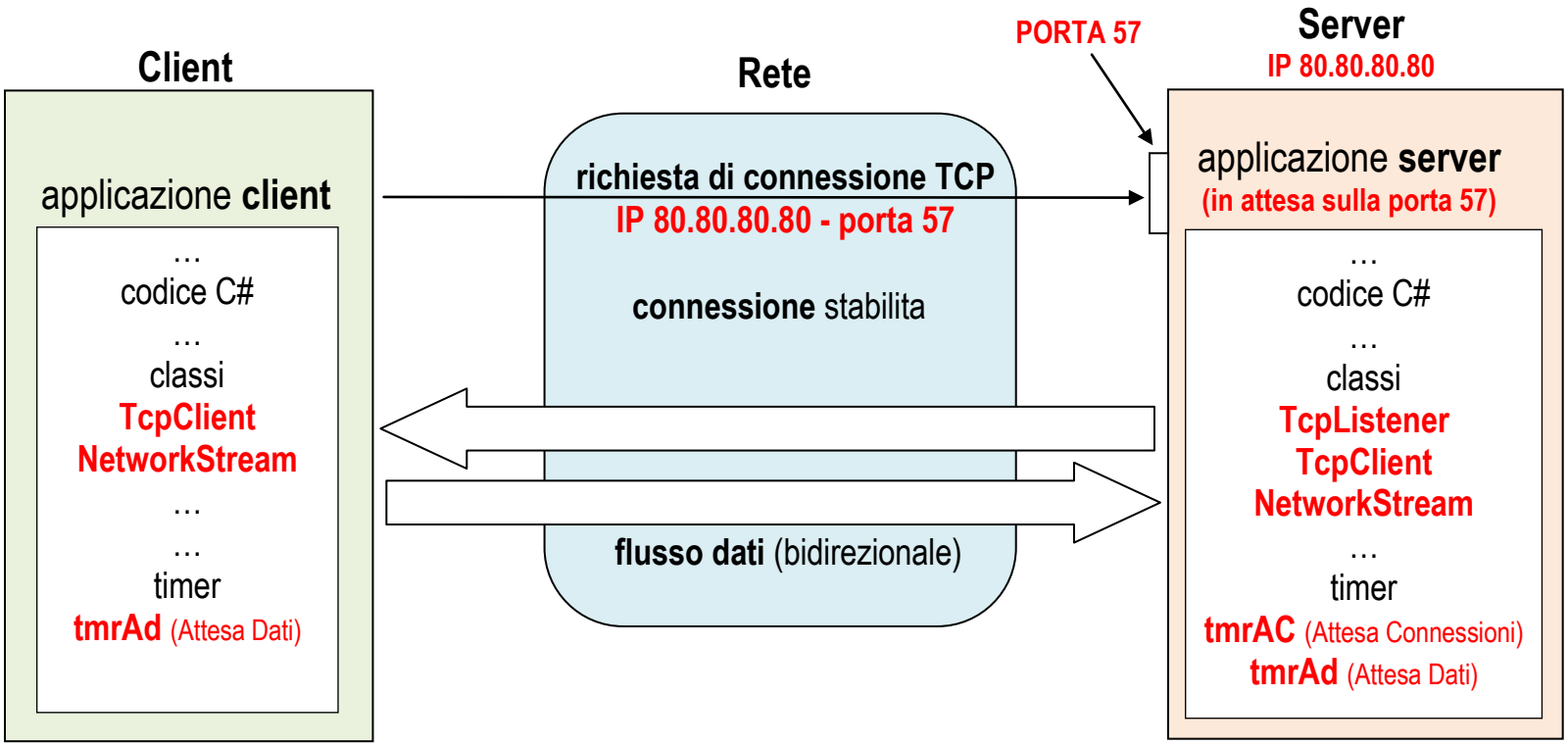
Schema di una Applicazioni di Rete C# basata su una Connessione TCP



SOCKET: Protocollo - IP Sorgente - IP Destinazione - Porta Sorgente - Porta Destinazione

esempio: TCP - 60.60.60.60 - 80.80.80.80 - 1024 - 57

APPLICAZIONI DI RETE

in ambiente Visual Studio (linguaggio C#)

Classi da Utilizzare	Libreria (*namespace*) da utilizzare: **using System.Net.Sockets** classe **TcpListener**: gestisce l'attesa per le Richieste di Connessione (*solo sul Server*) classe **TcpClient**: gestisce una Connessione e le relative operazioni classe **NetworkStream**: gestisce il Flusso di Dati di una Connessione	
Classe TcpListener	costruttore **TcpListener (<porta>)** metodo **.Start( )** avvia attesa x connessioni metodo **.Stop( )** ferma attesa x connessioni metodo **.Pending( )** *… restituisce bool* *… true se è arrivata una richiesta di conness.* metodo **.AcceptTcpClient( )** *… rest. TcpClient* *… restituisce la connessione (classe TcpClient) instaurata a seguito delle richiesta del client*	*Esempi:* TcpListener LST = new **TcpListener (57)**; LST**.Start ( )**; LST**.Stop ( )**; if (LST**.Pending( )**) … TcpClient CN = LST**.AcceptTcpClient( )**

Classe TcpClient	costruttore **TcpClient ( )**
	
	metodo **.Connect(<ip-server>, <porta>)**
	*… inoltra una richiesta di connessione al server con l'**IP indicato** e sulla **porta specificata**.*
	
	metodo **.GetStream( )** *… rest. NetworkStreamt*
	… restituisce il Flusso Dati (classe NetworkStream) relativo alla connessione indicata
	
	proprietà **.Available** *… di tipo intero*
	.. indica il numero di byte ricevuti sulla connessione e che sono pronti per essere letti dal buffer.
	
	proprietà **.Connected** *… di tipo bool*
	... indica se, a seguito di una operazione di rete, la connessione è ancora attiva o no.
	
	metodo **.Close( )** *… chiude la connessione*
		Esempi:
		
		TcpClient CN = new **TcpClient( )**;
		
		CN**.Connect("127.0.0.1", 57)** *(…sulla app.client)*
		
		NetworkStream NS = CN**.GetStream( )**;
		
		if (CN**.Available** > 0)
		{ *… leggi i dati disponibili …* }
		
		If (CN**.Connected**)
		{ *... operazioni sulla connessione …* }
		
		CN**.Close( )**;

Classe NetworkStream	L'oggetto NetworkStream si crea a partire da una connessione, con il **metodo CN.GetStream( )**
	
	metodo **.Read (** *<vettore-di-byte>*, **0,** *<numero-byte-da-leggere>* **)**
	… legge dal buffer, il numero di byte indicato e memorizza i byte letti nel vettore specificato.
	
	metodo **.Write (** *<vettore-di-byte>*, **0,** *<numero-byte-da-scrivere>* **)**
	… invia il contenuto del vettore di byte specificato, ma limitatamente al numero di byte indicato.
	
	metodo **.ReadByte( )** *… rest. un valore di tipo int*
	… legge un singolo byte dal buffer dei dati ricevuti.
	
	metodo **.WriteByte(** *<byte>* **)**
	… invia il byte indicato, sul flusso dati della connes.
		Esempi:
		NetworkStream NS = CN.**GetStream( )**;
		
		… per leggere tutti i dati ricevuti …
		int NumByte = CN.**Available**;
		byte[ ] Dati = new byte [NumByte];
		if (NumByte > 0)
		{ NS.**Read** (**Dati**, **0**, **NumByte**); ……
		
		… per inviare tutti i dati di un vettore di byte …
		byte[ ] Dati = { 15, 25, 35, 75, 65, 87 };
		NS.**Write** (**Dati**, **0**, **Dati.Length**);
		
		byte Dato = (byte)NS.**ReadByte( )**;
		
		byte Dato = 50;
		NS.**WriteByte**(**Dato**);

come è strutturato il SOCKET di rete	il **Socket individua univocamente una connessione di rete**. E' costituito dai seguenti dati riguardanti la connessione: **protocollo**, **ip-sorgente**, **porta-sorgente**, **ip-destinazione**, **porta-destinazione** **protocollo:** è il protocollo su cui si basa la connessione (ad esempio: TCP) **ip-sorg.:** indirizzo IP dell'host che ha richiesto la connessione (client) **ip-dest.:** indirizzo IP dell'host che ha accettato la connessione (server) **porta-dest.:** numero di porta che specifica quale servizio è richiesto dal client **porta-sorg.:** numero di porta generato casualmente dal client: è necessario per distinguere fra loro più connessioni indirizzate verso lo stesso server e verso lo stesso servizio.	
	Esempio: un CLIENT con indirizzo IP (200.10.20.30) apre una connessione per navigare (servizio web HTTP, porta 80) verso un server remoto avente IP (90.90.90.90). Il socket, per il Client, potrebbe essere così strutturato: **TCP, 200.10.20.30, 1025, 90.90.90.90, 80** *… la porta-sorg.1025 è generata casualmente …* **Attenzione … la stessa connessione, per il Server, avrebbe il socket inverso …** **TCP, 90.90.90.90, 80, 200.10.20.30, 1025**	
come convertire da Stringa a Bytes (e viceversa) secondo il codice ASCII	I dati ricevuti/inviati su una connessione di rete, devono essere sempre convertiti in **sequenze di byte**. Libreria (*namespace*) da utilizzare: **using System.Text** byte[ ] *mieiBytes* = **ASCIIEncoding.ASCII.GetBytes**(*miaStringa*) … converte la stinga miaStringa in una sequenza di byte (ASCII), memorizzandola nel vettore mieiBytes string *miaStringa* = **ASCIIEncoding.ASCII.GetString**(*mieiBytes*) … converte il vettore di byte (ASCII) *mieiBytes* in una stinga, memorizzandola nella variabile miaStringa	