

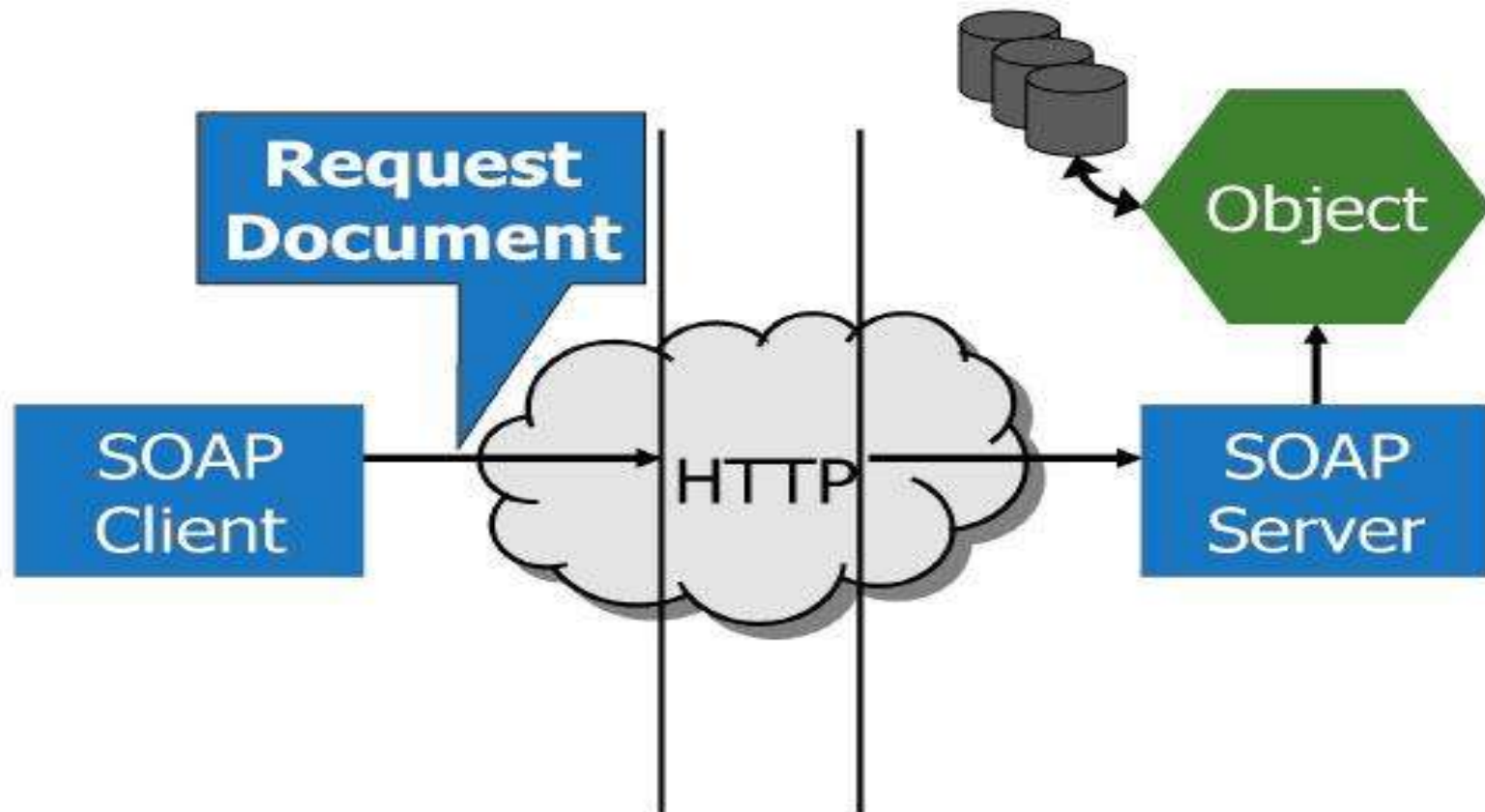
# Introduzione a SOAP e SOAP::Lite

- Introduzione a SOAP e SOAP::Lite
- Definizione di un Servizio
- Creazione Client e Server
- Autenticazione
- Gestione Errori
- Debugging
- Descrizione del Servizio tramite WSDL

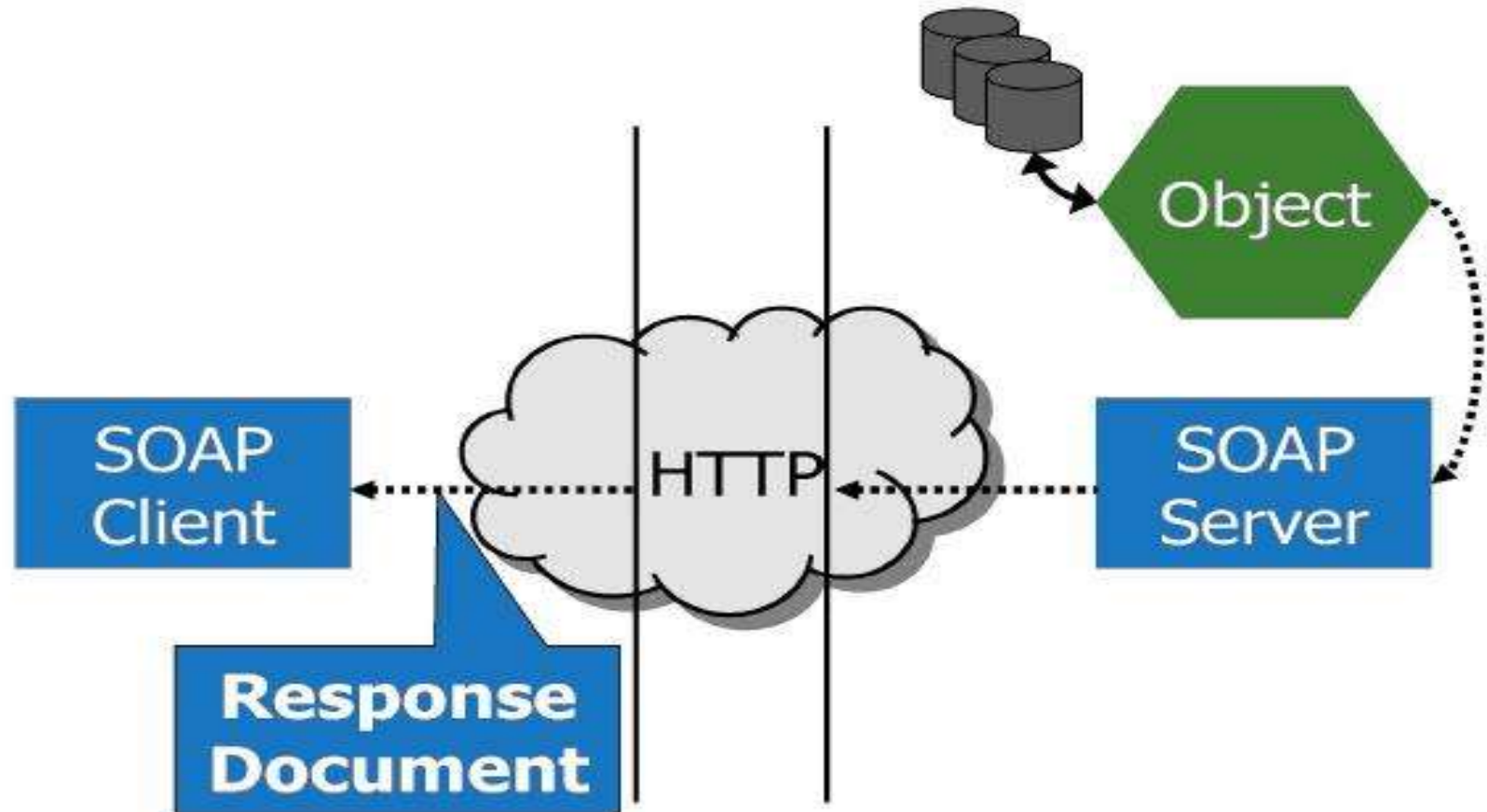
# Introduzione al SOAP

WSDL	Contract
SOAP	Envelope
HTTP, SMTP, FTP	Mailman
TCP/IP, UDP	Post office
Programming: DOM, SAX	Speech
Schema: DTD, XSD	Vocabulary
XML 1.0	Alphabet

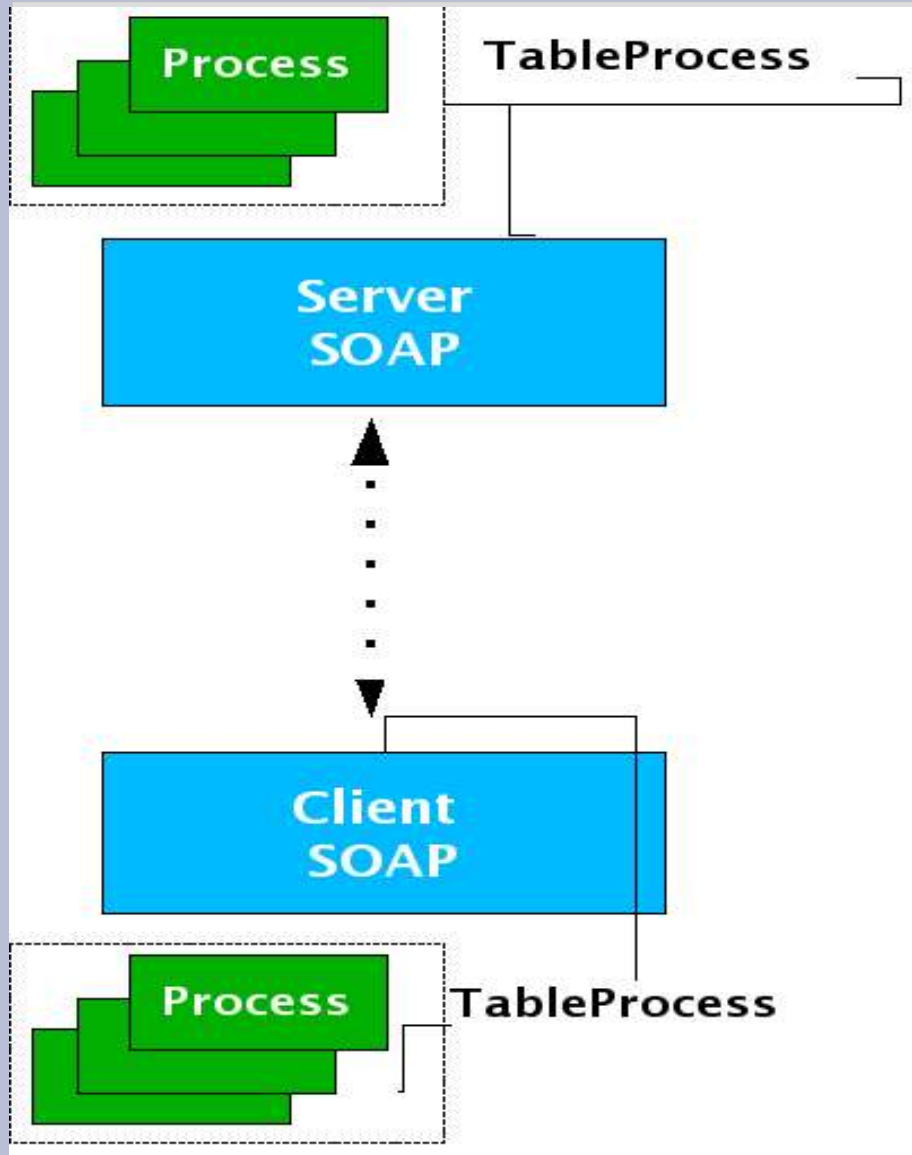
# Introduzione al SOAP



# Introduzione al SOAP



# Process Info Server



- **Controllo Processi remoti**
- **Accesso Limitato**
- **Protocollo trasporto HTTP**

# Server SOAP::Lite: Dispatching

- Dispatch Statico

```
-> dispatch_to ( 'Login', 'Info' );
```

- Dispatch Dinamico

```
-> dispatch_to ( '/home/modules' );
```

- Dispatch Misto

```
-> dispatch_to( '/home/modules', 'Login', 'Info' );
```

# Server SOAP::Lite

```
#!/usr/bin/perl -w

use strict;

use lib qw('../lib');

use Login;

use Info;

use SOAP::Transport::HTTP;

my $daemon = SOAP::Transport::HTTP::Daemon-> new (LocalPort =>
    10000, Reuse => 5);

$daemon->dispatch_to('Login', 'Info');

$daemon->handle;
```

# Client SOAP::Lite

- **STANDARD**

```
my $s = SOAP::Lite
    ->uri ( 'urn:Info' )
    ->proxy( 'http://myhost:10000/' );

my $stableprocess = $s -> getProcess() -> result;
```

- **WSDL**

```
my $s = SOAP::Lite -> service( 'http://myhost/info.wsdl' ) ;
my $stableprocess = $s -> getProcess();
```

# Client SOAP::Lite

```
Use TableProcess ;

my $s = SOAP::Lite -> service( 'http://myhost/info.wsdl' ) ;

# autenticazione
my $ticket = $s -> login ( 'User', 'Password' );

# lista dei processi
my $stableprocess = $s -> getProcess ( $ticket );

# restituisco pid
print $stableprocess->getAllPids ( ) ;
```

# Client Soap::Lite Autodispatch

```
Use TableProcess ;

Use SOAP::Lite + autodispatch =>

    service => ( 'http://myhost/info.wsdl' );

# autenticazione
my $ticket = login ( 'User', 'Password' );

# lista dei processi
my $stableprocess = getProcess ( $ticket );

# restituisco pid
print $stableprocess->getAllPids ( ) ;
```

# Autenticazione: Ticket Based

```
sub login {  
    my ($self, $user, $password) = @_;  
  
    unless ( $self->_check( $user, $password ) ) {  
  
        my $ticket = $self->_getticket( $user );  
  
        return SOAP::Data->name('ticket')->type('string')->value($ticket) ; }  
  
    return 0;  
}
```

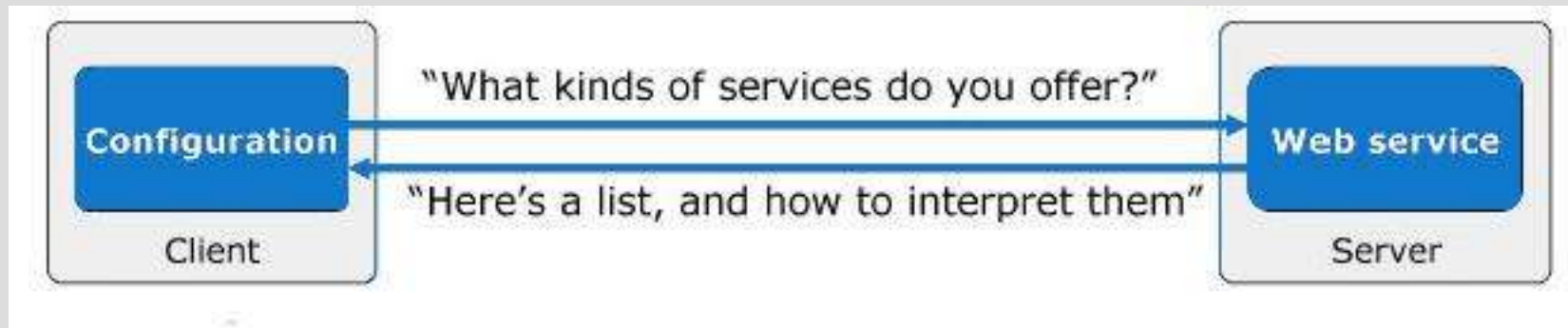
# Gestione Errori : Server

```
...  
sub getProcess{  
    my ( $self, $ticket, $name ) = @_ ;  
    die _error('Client', 'Ticket Non Valido' ) unless ( validate( $ticket) );  
...  
sub _error($) {  
    my ( $faultcode, $faultstring ) = @_ ;  
    return SOAP::Fault  
        ->faultcode($faultcode)  
        ->faultstring($faultstring);  
}
```

# Gestione Errori : Client

```
my $s = SOAP::Lite
    ->service('http://myhost/info.wsdl')
    ->on_fault(
        sub { my($soap, $res) = @_;
            die ref $res ? $res->faultstring : $soap->transport->status, "\n";
        });
```

# Definizione di un WebService tramite WSDL



- Web Service Description Language
- Standardizzazione delle descrizione di un webservice

# WSDL: Definizione dei Tipi

```
<xsd:complexType name="Process">
  <xsd:sequence>
    <xsd:element maxOccurs="1" minOccurs="1" name="name"
      type="xsd:string" />
    <xsd:element maxOccurs="1" minOccurs="1" name="pid"
      type="xsd:int" />
  </xsd:sequence>
</xsd:complexType>

...

<xsd:complexType name="TableProcess">
  <xsd:complexContent>
    <xsd:restriction base="SOAP-ENC:Array">
      <xsd:attribute ref="SOAP-ENC:arrayType"
        wsdl:arrayType="xsd1:Process[]" />
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

# Definizione delle Operazioni

- Definizione dei Messaggi

```
<message name="getProcessRequest" >
```

```
    <part name="ticket" type="xsd:string" /> </message>
```

```
<message name="getProcessResponse" >
```

```
    <part name="table" type="xsd1:TableProcess" /> </message>
```

- Definizione dell'Operazione

```
<operation name="getProcess" >
```

```
    <input message="tns:getProcessRequest" />
```

```
    <output message="tns:getProcessResponse" />
```

```
</operation>
```

# SOAP::Schema

- **Caricamento di WSDL**

```
use SOAP::Lite -> service ('http://myhost/info.wsdl');
```

- **Generazione di stub**

```
perl stubmaker.pl 'http://myhost/info.wsdl'
```

# Debugging

- Debug Interno

```
use SOAP::Lite + trace ;
```

```
use SOAP::Lite + trace => transport ;
```

```
use SOAP::Lite + trace => [ debug => { 'do_what_I_want_here' } ] ;
```

```
my $soap = SOAP::Lite -> on_debug ( sub{ print @_ } ) ;
```

- Debug esterno

```
use Data::Dumper ;
```

```
print Dumper( $soap ) ;
```