Defining a Network Protocol in a Domain Specific Language

Anders Olav Candasamy

Supervisor
Edwin Brady
Network Protocol

Defines the **order** and **content** of messages
- An agreement

Used for:
- Communication
  - Browsing the web
- Establish secure and encrypted connections
  - Online authentication
Domain Specific Language

Language that solves the problem of a specific domain

- Structured Query Language (SQL)
- HyperText Markup Language (HTML)
Motivation

● Protocols are often complex
  ○ Many types of message
  ○ Many steps involved
● Mistakes have consequences
Apple’s GOTO bug

```c
if ((err = SecurityCheck1())) != 0)
    goto fail;

... goto fail; // duplicated line

if ((err = SecurityCheck2()) != 0) // dead code
    goto fail;
```
Solution

- Define a model inside the implementation
  - Implementation must obey the model
  - Stop communication on errors
  - Human readable syntax
Architecture
Defining a protocol

Echo Server
1. Receive message
2. Return message
Echo server specification

// EndPoint (immutable)
val ep = ProtocolBuilder

// Server protocol
val echoServer = ep receives anEcho sends anEcho
Validators

● Tests a message's properties
  ○ Type
  ○ Value

● Returns a Left or Right value
  ○ Left indicates an error
  ○ Right indicates protocol compliance
val echoServer = ep receives anEcho sends anEcho
Message handling

- Actors all have a **receive** method
  - Pattern matching
  - Event based

source: http://www.educationaltoysplanet.com/hid-n-seek-shape-sorter-wooden-toy.html
Secure Echo Server

1. Establish encrypted communication
   a. Diffie-Hellman-Merkel key exchange
      i. Prime and generator
      ii. Send and receive PublicKey

2. Echo messages received
Server specification

val diffieInit = ep receives primeAndGenerator
               sends aPublicKey receives aPublicKey

val fullProtocol = diffieInit next (echoServer loop())
Error detection

"protocol violated - sending when should be receiving"
ValidationError

“ValidationError(4 is not a prime number)"

“ValidationError(Msg could not be converted to a PrimeAndGenerator class: ,net.liftweb.json.MappingException: Do not know how to convert JString(239.0) into double)”
Performance

Echo server - 10,000 messages

- 20% performance loss
  - Compared to a “basic” system
Concurrency
Latency
Apple’s GOTO bug

● Would use of such a DSL prevent Apple’s GOTO bug?
  ○ Development team culture
    ■ Code duplications
    ■ No compiler warnings
    ■ Poor merging practices
  ○ PM depends on correct validators
Conclusion

- Manages complexity
  - Natural separation of concerns
- Detects errors in messages
  - Dependant on correct Validators
  - Uses dynamic checking
  - Does not detect lack of message
- Can be used for testing implementations
  - Define a Client to test a Server
  - Allows for any type of Server implementation
Future Work

- Static checking
- Make definition of Validators more human readable
- Syntax for defining multiple endpoints
  - Internal Protocol Monitors
?
Extra Andrii slides

1. Needham-Schroeder
2. External or Internal DSL
3. Tools & Actor model
4. Secure Chat Server
5. Validator code
Needham-Schroeder
External or Internal DSL

- **External**
  - Full syntactic freedom
  - Translation to compiled language
  - Domain experts

- **Internal**
  - Built in a host language
  - Syntax restricted
Tools

- **Scala**
  - Object Oriented & Functional
  - Scalable
    - Internal DSL

- **Akka**
  - Actors
    - Concurrency
    - Message passing
    - Actor Model
Actor Model
Secure Chat Server

1. Client connects to the chat server
2. Establish encrypted communication with server
3. Register a username with server
4. Establish encrypted communication with new users
5. Send encrypted greeting to new users
Server Protocol Definition

val secureChat =

diffiInit receives aUsername

next (ep anyone aChatMessage loop())
Server

[info] Running implementation.serverImpl.ServerMain
Hit ENTER to exit ...
Server bond to: /127.0.0.1:8888
New user connected: 175014914
New user connected: 1856862044
Client A

Terminal

Enter number: 1

[info] Running implementation.clientImpl.ClientMain
Hit ENTER to exit ...
We are 175014914
NewUser alert. The connected user is: 1856862044
Starting a secure Communication with user: 1856862044
Secure communication opened
Sending encrypted message...
Client B

<table>
<thead>
<tr>
<th>File</th>
<th>Edit</th>
<th>View</th>
<th>Search</th>
<th>Terminal</th>
<th>Help</th>
</tr>
</thead>
</table>

Enter number: 1

[info] Running implementation.clientImpl.ClientMain
Hit ENTER to exit ...
We are 1856862044
We got a message from 175014914
Encrypted: +gLCoGULjxUOloCZQAdsECHXIalu1fULQ618ydgcfVk=
Decrypted: Hello my dear friend
val anEcho = new Validator( input =>
    if isGood(input)
        Right(Echo(input))
    else
        Left(ValidationError(“Not an Echo message”))
)