Embedded Systems

Introduction

In this document we will give an example for modelling with the SDL language and then extend it in two steps.

Description of the initial scenario

Chinook is a program which plays checkers agains several clients at the same time over the internet. These clients are controlled by human players. We model the following aspects of behaviour:

The Chinook server should be able to

- receive board configurations
- calculate moves from it
- send these moves to the respective client

Clients should be able to

- calculate the initial board configuration
- wait for a players move
- check the players move for correctness and reject incorrect moves
- calculate a new board configuration from the former one plus the players move
- display the board to the player
- send the board (plus the clients ID) to the chinook server
- receive chinook moves
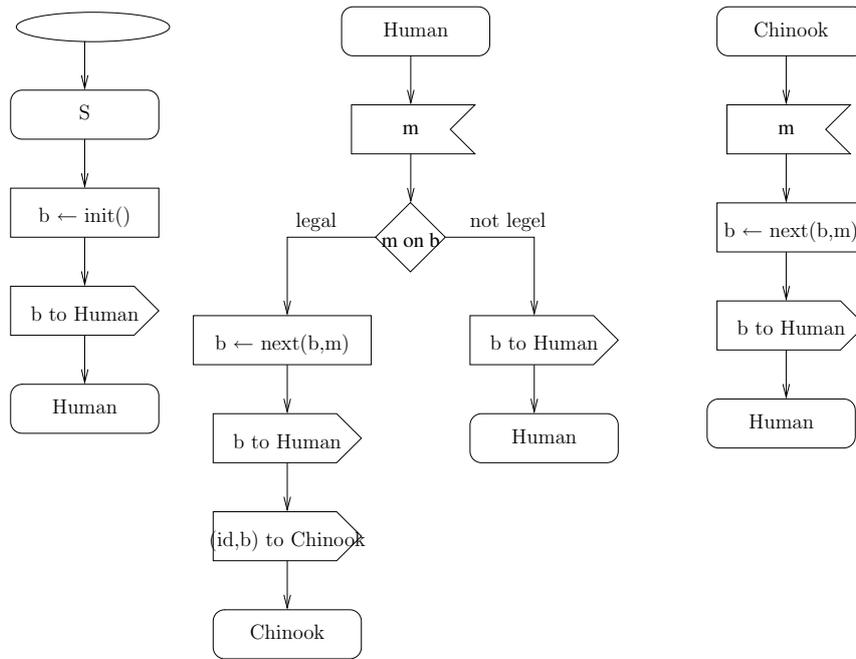- calculate new board configurations from the former one plus the chinooks move

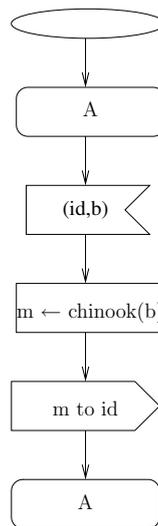Notice that we assume that clients are trustworthy to the server.
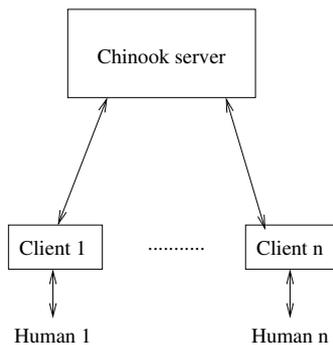
Modelling the initial scenario in SDL

The SDL descrition of the initial scenario consists of the parts for the client and the server

Client

```
      (  )
       │
       ▼
     ┌─────┐
     │  S  │
     └─────┘
       │
       ▼
  ┌──────────┐
  │ b ← init() │
  └──────────┘
       │
       ▼
  ┌──────────┐
  │ b to Human │>
  └──────────┘
       │
       ▼
  ┌──────────┐
  │  Human   │
  └──────────┘
```

```
  ┌──────────┐
  │  Human   │
  └──────────┘
       │
       ▼
  ┌──────────┐
  │    m     │<
  └──────────┘
       │
       ▼
   legal ◇ m on b ◇ not legel
```

legal →
```
  ┌──────────────┐
  │ b ← next(b,m) │
  └──────────────┘
       │
       ▼
  ┌──────────┐
  │ b to Human │>
  └──────────┘
       │
       ▼
  ┌──────────────┐
  │ (id,b) to Chinook │>
  └──────────────┘
       │
       ▼
  ┌──────────┐
  │ Chinook  │
  └──────────┘
```

not legel →
```
  ┌──────────┐
  │ b to Human │>
  └──────────┘
       │
       ▼
  ┌──────────┐
  │  Human   │
  └──────────┘
```

```
  ┌──────────┐
  │ Chinook  │
  └──────────┘
       │
       ▼
  ┌──────────┐
  │    m     │<
  └──────────┘
       │
       ▼
  ┌──────────────┐
  │ b ← next(b,m) │
  └──────────────┘
       │
       ▼
  ┌──────────┐
  │ b to Human │>
  └──────────┘
       │
       ▼
  ┌──────────┐
  │  Human   │
  └──────────┘
```

Server

```
      (  )
       │
       ▼
     ┌─────┐
     │  A  │
     └─────┘
       │
       ▼
  ┌──────────┐
  │  (id,b)  │<
  └──────────┘
       │
       ▼
  ┌──────────────┐
  │ m ← chinook(b) │
  └──────────────┘
       │
       ▼
  ┌──────────┐
  │ m to id  │>
  └──────────┘
       │
       ▼
     ┌─────┐
     │  A  │
     └─────┘
```
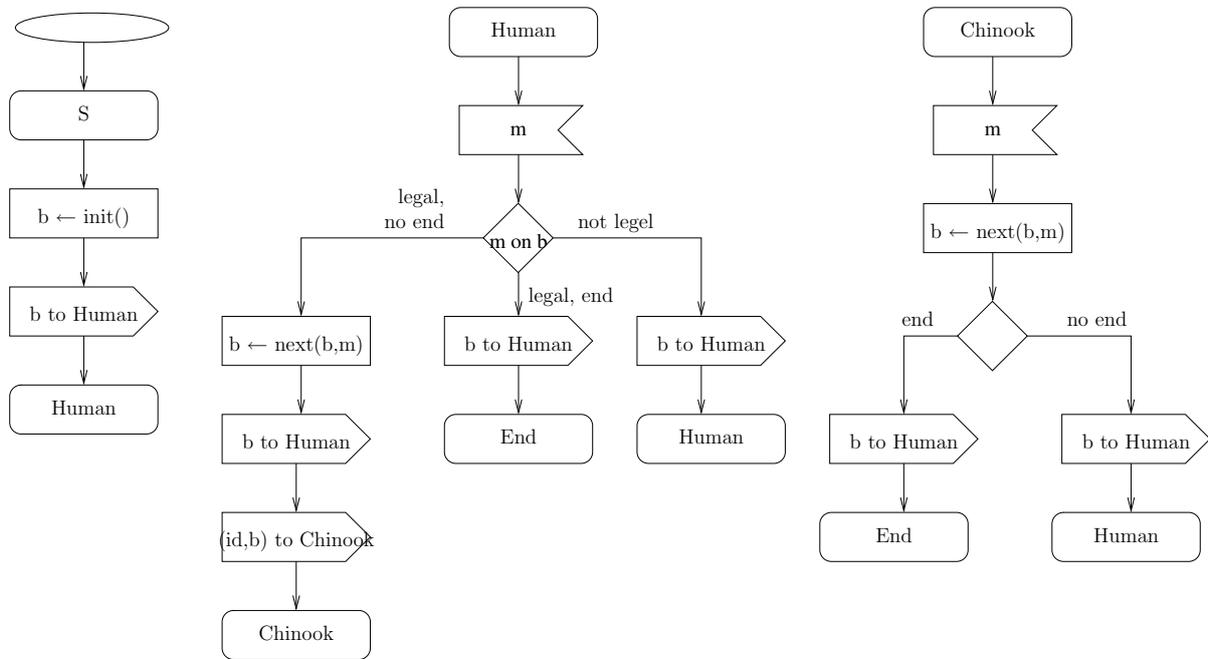
First extension: Checking whether game has ended

It shall be checked whether the game was ended. If the game ends, the client moves into an end state. We only have to modify the client.

S

b ← init()

b to Human

Human

Human

m

legal,
no end

m on b

not legel

legal, end

b ← next(b,m)

b to Human

b to Human

b to Human

End

Human

(id,b) to Chinook

Chinook

Chinook

m

b ← next(b,m)

end

no end

b to Human

b to Human

End

Human

Second extension: Limiting players time per turn

Additional to the first extension, the time the player has to move shall be limited to five minutes. If the five minutes run out in the players move, the player should be signalled that a timeout occured Gand the game is ended. Again, we only have to modify the client.

S

b ← init()

b to Human

SET(now + 5 min, T)

Human

Human

m

legal,
no end

m on b

not legel

legal, end

b ← next(b,m)

b to Human

b to Human

End

Human

(id,b) to Chinook

Chinook

T

timeout

End

T

Chinook

m

b ← next(b,m)

end

no end

b to Human

b to Human

End

SET(now + 5 min, T)

Human