Implementation of Communication Protocols

Networks and Embedded Software

Module 5.2.3

by Wolfgang Neff

Implementation (1)

- Design process
  - Analyze the protocol
  - Make finite state machine for the protocol
  - Take care of error handling
  - Implement state machine
  - Test protocol

Implementation (2)

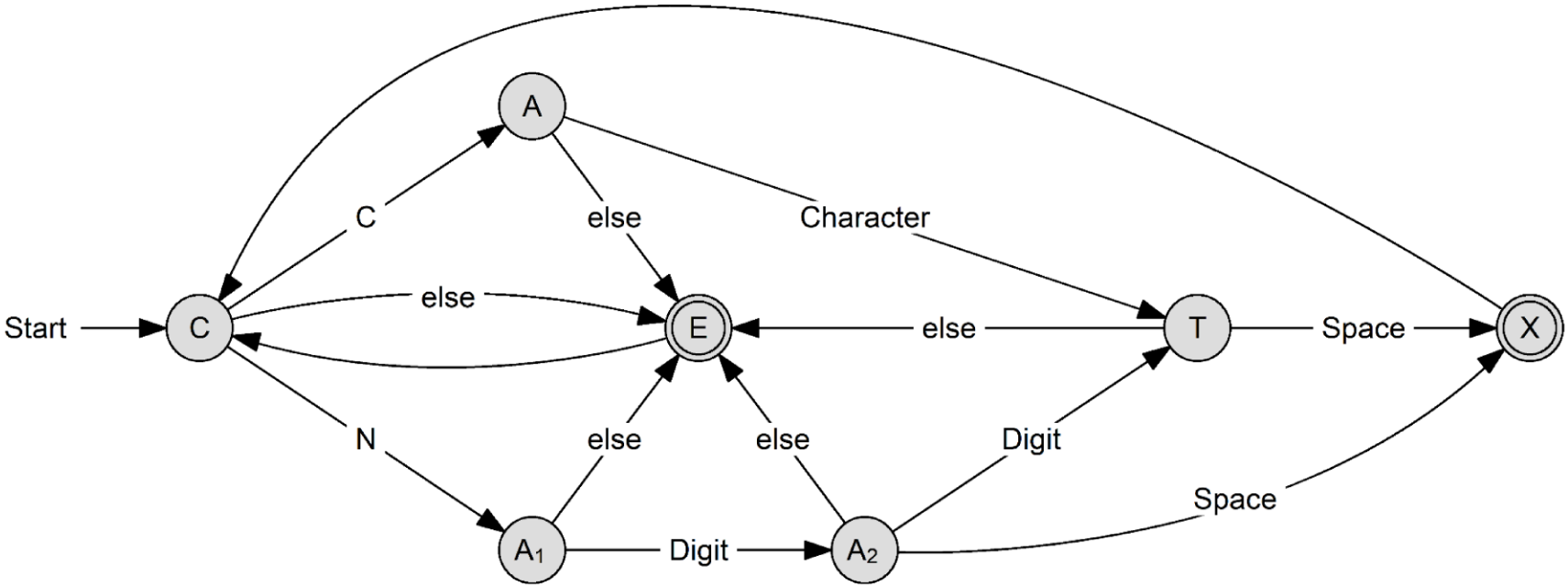
- Example
  - Start Condition
    - Command ('C', 'N')
  - Stop Condition
    - Terminal (*Space*)
  - Message Format
    - «COMMAND»«ARGUMENT»«TERMINAL»
    - «COMMAND» «ARG1»«ARG2»«TERMINAL»

Implementation (3)

- Example (continued)
  - Meaning of Message
    - C *char* : Read a character
    - N *digit* : Read a digit
    - N *digit digit* - Read a two-digit number

Implementation (4)

- Example (continued)
  - Finite State Machine



Implementation (5)

- Example (continued)
  - Enumerations

```
typedef enum {
    CMD_NUMBER,
    CMD_CHARACTER
} command_t;
typedef enum {
    STATE_GET_COMMAND,
    STATE_GET_CHAR_ARG,
    STATE_GET_NUM_ARG1,
    STATE_GET_NUM_ARG2,
    STATE_GET_TERMINAL,
    STATE_EXECUTE,
    STATE_ERROR
} state_t;
```

Implementation (6)

- Example (continued)
  - Implementation of FSM

```
switch (state) {
    case STATE_GET_COMMAND:
        token = get_token();
        state = parse_command(token, &command);
        break;
    case STATE_GET_CHAR_ARG:
        token = get_token();
        state = parse_argument(token, &arg1);
        break;
    case STATE_GET_NUM_ARG1:
        token = get_token();
        state = parse_argument1(token, &arg2);
        break;
```

Implementation (7)

- Example (continued)
  - Implementation of FSM (continued)

```
…
case STATE_GET_NUM_ARG2:
    token = get_token();
    state = parse_argument2(token, &arg2);
    break;
case STATE_GET_TERMINAL:
    token = get_token();
    state = parse_terminal(token);
    break;
case STATE_EXECUTE:
    state = execute_command(command, arg1, arg2);
    break
…
```

Implementation (8)

- Example (continued)
  - Implementation of FSM (finished)

```
        …
        case STATE_ERROR:
            handle_error(state);
            state = STATE_GET_COMMAND;
            break;
    } /* switch */
```

Implementation (9)

- Example (finished)
  - Parsing the Command

```
state_t parse_command(char token, command_t command){
    if (token == 'C') {
        *command = CMD_CHARACTER;
        return STATE_GET_CHAR_ARG;
    } else if (token == 'N') {
        *command = CMD_NUMBER;
        return STATE_GET_NUM_ARG1;
    } else {
        return STATE_ERROR;
    }
}
```