Software Top Level Design Document

Experiment Version 2.0

1.0 Introduction

This document presents the top level design for the Taxi Evolution [SRS2000001]. The main purpose of this document is to divide the system into modules which can be individually developed by different development teams.

This document is structured as follows: Section 2.0 gives an overview of the system and its design. A detailed description of the interfaces in the system is presented in 3.0. A description of the interaction design is found in section 4.0, in a MSC notation.

2.0 Design Overview

Figure 1 depicts the three modules in the system.

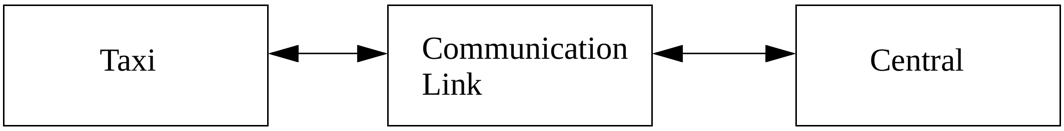


FIGURE 1. System modules

- Taxi module - The Taxi module is the implementation of all the systems present in the taxi. All input to this module from the user comes from the user interface on the terminal in the car. This interface is not part of this design. Instead, only the API is given. In this first implementation, the system is simulated. Due to this, the block Taxi acts as if there are many taxis. That is, even though there is just one block there are still several taxis. The number of taxis are set at start-up, but the active number are controlled by the API.
- Communication Link module - This module contains two main parts: One part for simulating the communication that will take place (mainly the radio communication). One part is the implementation of the communication server. This server is responsible for routing the radio connections correct. As there are more than one radio antenna, some routing is necessary.
- Central module - This module contains the implementation of the systems present for the operators in the central. The user interface is not part of this design. Instead, just the API to the system is presented.

3.0 Interface Description

This section describes the interface to the system. The properties for the communication link is also described, among these the structure of the communication protocol used to send information between the taxis and the central.

The application interface (API) is described as SDL signals. That is, the design assumes that SDL signals are sent to and from the system.

3.1 Global Definitions and Data structures

Order_struct - This data structure describes the information in an order. The fields are:

- Order_number, integer - The identification number of an order.
- Time, integer - The time when the customer is suppose to be picked up.
- Dispatch_time, integer - The time when the order should be dispatched from the central.
- Address, integer - The address where the pick-up is to occur. Note that this is an integer. An integer is used in this implementation as no real map is available.
- Name, charstring - The name of the customer to be picked up.
- Misc, charstring - Miscellaneous information to the driver. For example, special pick-up instructions required for a handicapped person, allergy information, destination etc.

Note that dispatch time is only used on the central. Also note that the address is represented by an integer. This is for implementation reason in this primary version, as there exists no map onto where real addresses can be associated with zones. Hence, the zone translation has to be done by the operator and this information is stored in the address fields. In the final version the address translation into zone will be automatic.

The times are represented by one integer. This integer is the number of seconds that has elapsed form the first of January, 2000.

3.2 Taxi-User API

3.2.1 Signals from the user (taxi driver) to the taxi module

- Ok(integer) - Sent when the user accepts an order. The parameter is the number of the sending taxi.
- Order(Order_struct) - Sent when the driver enters an order into the system.
- Start_Voice(integer,integer) - Sent when the driver initiates voice communication. The parameter is sending taxi number and operator number. If no specific operator is sought, the latter should be set to 0.
- Terminate_Voice(integer) - Sent when the driver terminates the voice communication. Parameter is sending taxi number.

- Send_Voice(integer) - The actual voice message, sent while the send button is pressed on the terminal, i.e. after Start_Voice is given. The parameter is the sending taxi number.
- Stop_Voice(integer) - This signal is sent when the send button is released. That is, when the sound information should not be sent any longer. The parameter is the sending taxi number.
- LoginSignal - Sent when the driver sweeps the driver identity card in the terminal to log on.
- Meter_On(integer) - Sent when the driver starts the meter, i.e. starts driving a customer. The parameter is the sending taxi number.
- Meter_Off(integer) - Sent when the driver stops the meter, i.e. stops driving a customer. The parameter is the sending taxi number.
- Overview(integer, integer, integer) - Sent when the driver requests overview information, i.e. information about cars, zones etc. The parameters are the sending taxi number, the zone in the middle and the number of zones that should be returned.
- Reject_Order(integer) - Sent when the driver rejects an incoming order. The parameter is the sending taxi number.
- Reset_Alarm(integer) - The alarm for the taxi, identified by the parameter in the signal, is reset.

3.2.2 Signals from the taxi module to the user (taxi driver)

- Order(Order_struct) - Received an order, which is sent to the driver.
- AlarmReset(integer) - Received when the alarm is reset. The parameter is the taxi number.
- Confirm_Voice(integer, integer)- Received to confirm that the central is setup for voice communication. The parameters are the taxi number and the operator number.
- Start_Voice(integer, integer) - Received when the central wants to initiate voice communication. The parameters are the operator number and the taxi number.
- Terminate_Voice(integer) - Received when the central wants to terminate voice communication. The parameter is the taxi number.
- LoginOk(integer) - The identity card was accepted and the driver was logged in. The parameter is the taxi number.
- LoginFailed(integer) - The identity card was not accepted. Login was not possible. The parameter is the taxi number.
- ZoneInfo(integer, integer, integer, integer) - Zone information is sent to the display. The parameters are the taxi number, the number of the zone, the number of free cars in the zone, and the number of orders in the zone.
- Text_Message(integer, charstring) - A text message from the central. Parameters are taxi number (if 0 then all cars are receivers) and text message.

3.3 Central-Operator API

3.3.1 Signals from the user (operator) to the central module

- OrderOper(Order_struct, integer) - When the operator has received an order, it is submitted to the system. The parameters are the order_struct, containing the order information and sending operator number.
- Overview(integer, integer, integer) - If the operator wants to see an overview of the amount of traffic in some zones. The parameters describe the operator number, central zone and the last parameter the number of zones that should be returned.
- Allocate_car(integer, integer) - If an operator wants to specifically allocate a specific car. The first parameter corresponds to the number on a operator. The second parameter corresponds to a taxi number.
- AlarmReset(integer) - Sent to reset the alarm in the taxi with the number specified according to the parameter.
- Start_Voice(integer, integer) - Sent from the operator to establish a radio communication to a taxi. The parameters are the sending operator and the receiving taxi number.
- Terminate_Voice(integer) - The operator sends this signal when wanting to terminate the voice communication. The parameter is the operator number.
- Send_Voice(integer) - This signal indicates when the sound transmission should occur, as pressing the send button on a communication radio. The parameter is the operator number.
- Stop_Voice(integer) - This signal indicates to the voice radio link to stop sending sounds across the link. The parameter is the operator number.
- Get_Order(integer, integer) - If the operator wants to view a specific order, this signal is sent to retrieve it. The parameters are operator number and order number.
- Text_Message(integer, integer, charstring) - An operator sends a text message. The parameters are the receiving taxi (if 0 then all taxis are receivers), message number and the message.
- Taxi(integer, integer, integer) - If the operator requests a specific taxi, this is the response. Parameters are operator number, taxi number and state of taxi. The latter is 0, if not logged in, 1 if available, 2 if logged in but not available.

3.3.2 Signals from the central module to the user (operator)

- ZoneInfo(integer, integer, integer) - Information on zone number, number of cars in the zone and number of orders in the zone.
- Start_Voice(integer, integer) - Voice communication initiated by the driver. The parameters are taxi number and operator number.
- Terminate_Voice(integer) - When the other side has terminated the communication, this is sent, independent of which side that terminated first. Parameter is operator number.

- Confirm_Voice(integer, integer) - When the voice channel is established, this signal is sent. The parameters are the operator number, taxi number.
- Order(Order_struct) - An order is sent to the operator, either because an operator requested it, or because the order requires manual dispatching.
- Confirm(integer, integer) - Confirmation signal on an order. The parameters are operator number and order number.

3.4 Taxi-Central API

Note that the signals sent over the communication link are coded as presented in section 3.5 and 3.6. That is, these signals are not actually sent or received by the systems. Instead, Radio_msg is sent and received. The signals in this part are coded according to section 3.5 and 3.6. However, this coding is not considered here, only the semantics of the signals.

3.4.1 Signals from the Central Module to the Taxi module

- Order(Order_struct) - An order is sent from the central system to a specific taxi. The taxi in question is identified in the Order_struct.
- Confirm(integer, integer) - Sent to confirm a submitted order by a taxi. The parameters are taxi number and order number.
- Start_Voice(integer, integer) - Sent when a operator wants to initiate a voice communication. Parameters are taxi number and operator number.
- Terminate_Voice(integer, integer) - This signal is sent when the operator side has terminated the voice communication. This signal is also used to cancel a voice communication before it is initiated. Parameters are taxi number and operator number
- Confirm_Voice(integer, integer) - Confirms the establishing of a voice communication. Parameters are operator number and taxi number.
- Voice_msg(integer, integer, integer) - This signal contains voice data in digital form. Parameters are taxi number, operator number and voice data.
- AlarmReset(integer) - The alarm for the taxi, identified by the parameter in the signal, is reset.
- Text_Message(integer, integer, charstring) - A text message. Parameters are taxi number, message number and message.
- LoginOk(integer) - The system confirms that a login was successful. Parameter is the taxi number.

3.4.2 Signals from the Taxi Module to the Central Module

- Ack(integer, integer) - Sent from a taxi when an order is accepted. The parameters are the taxi number and the order number.
- Order_Reject(integer, integer) - If the driver rejects an order, this signal is sent. Taxi number and order number are the parameters.

- Order_Cancel(integer, integer) - If an order is cancelled (after having being accepted once), this signal indicates this. The parameters are taxi number and order number.
- LoggedIn(integer) - Information on a logged in driver is sent to the central. The Parameter is taxi number.
- LoggedOut(integer, integer) - A driver logs off. Parameters are taxi number and driver number.
- Order(Order_struct) - If a driver has entered an order, the order in question is sent to the central. Parameter is the order in the Order_struct.
- Not_Avail(integer) - When a driver is logged in, but not available for orders, this signal is sent. Parameter is taxi number.
- Start_Voice(integer) - Sent when a driver wants to initiate a voice communication. Parameter is taxi number.
- Terminate_Voice(integer, integer) - This signal is sent when the driver side has terminated the voice communication. This signal is also used to cancel a voice communication before it is initiated. Parameters are operator number and taxi number.
- Confirm_Voice(integer, integer) - Confirms the establishing of a voice communication. Parameters are operator number and taxi number.
- Voice_msg(integer, integer) - This signal contains voice data in digital form. Parameters are taxi number and voice data.
- Position(integer, integer) - This signal is sent from the taxi to the central when the taxi has changed zone. The parameters are taxi number and zone number.

3.5 Communication Link Design

The communication link expects a one-to-many network topology. That is, it is assumed that there are many taxis, but only one central. Because of this, all data packets sent on the communication link contains a identification number of a taxi, but no for the central.

3.6 Communication Protocol Description

The communication protocol is defined as a string of characters with certain fields. The fields are:

1. Taxi Number - The number of the sending or receiving taxi.
2. Message Typd Id - An identification number of the message type.
3. Data1 - Data of type integer.
4. Data2 - Data of type integer.
5. Data3 - Data of type integer.
6. Data4 - Data of type integer.
7. String1 - Data of type charstring.

8. String2 - Data of type charstring.

Table 1 Present all the signals sent over the communication link and their definition in the protocol format. An ‘x’ in one of the parameter fields indicates that the field is used when coding the signal. In general, the parameters in the signals are coded into the fields described above in the order the parameters are defined in the signal. As the taxi number is part of all signals sent, this is not displayed in table 1. The signal is defined as above.

TABLE 1. Message Types

Signal name	Message Type Id	Data1	Data2	Data3	Data4	String1	String2
Ack	1	x					
Confirm_Voice	2	x					
LoggedIn	3						
LoggedOut	4	x					
LoginOk	5						
Not_Avail	6						
Order[a]	7	x	x	x	x	x	x
Order_Reject	8	x					
OverviewSignal	9	x	x	x			
Position	10	x					
AlarmReset	11						
Start_Voice	12	x					
Terminate_Voice	13	x					
Text_Message	14	x				x	
Voice_msg	15	x	x				
ZoneInfo	16	x	x	x			
Order_Cancel	17	x					

[a] The order signal actually contains one more integer in the signal. But since the order number is not used when the taxi sends an order to the central and that the dispatch-time is not interesting to the taxi, one integer field is omitted.

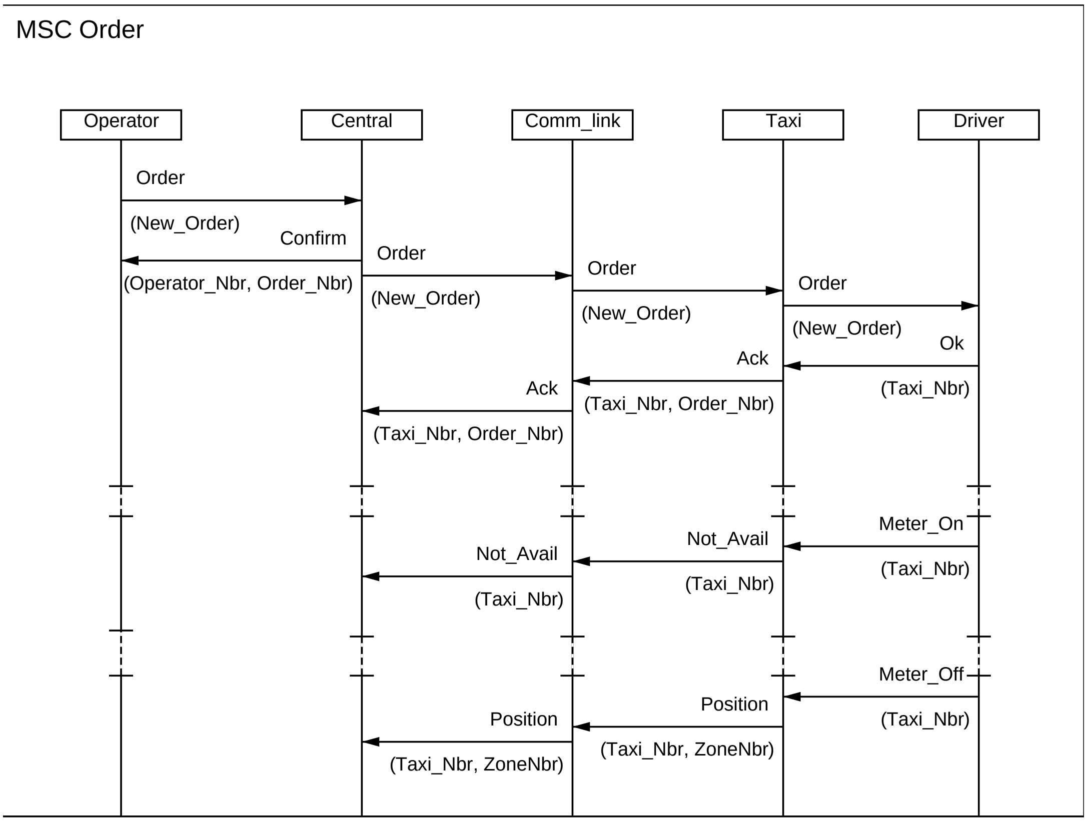
Fault coding is assumed to be handled by underlying protocols. Hence, this is not considered at all in this protocol design.

4.0 Message Sequence Charts

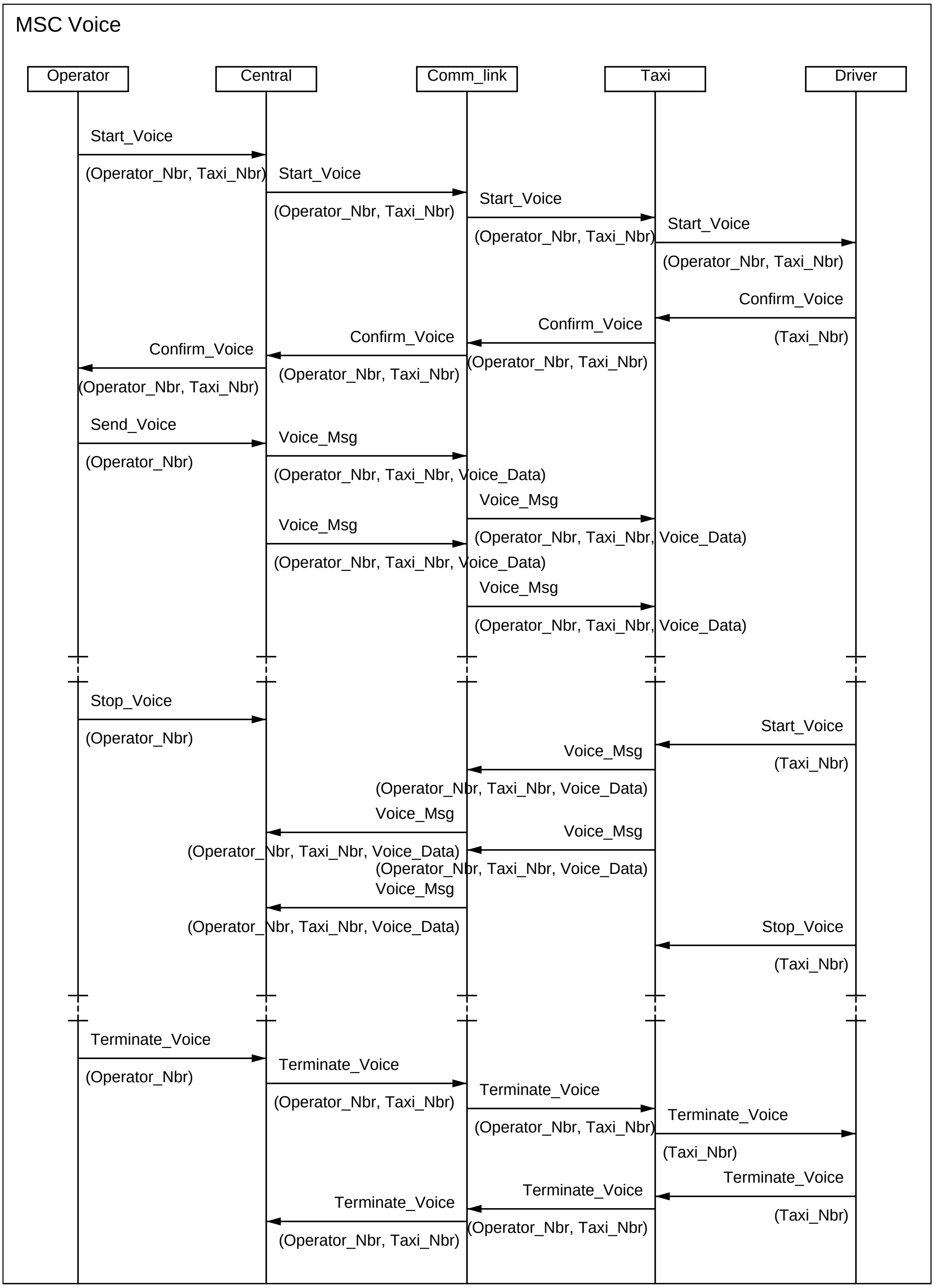
This section contains two MSCs to depict the interaction behavior. The first chart describe an orders transition through the system (section 4.1), the second an example on voice communication (section 4.2). The other interaction behavior of the taxi and the central module is understood from the signal definition.

Note that the SDL signals are used throughout the MSC, without the coding of the signals into text messages as described in section 3.6.

4.1 Order handling, normal case



4.2 Voice communication, central initiated



5.0 References

SRS2000001	Requirement Specification for the Taxi Evolution project.