Requirements Definition of Communication Platform

Workpackage: WP5
Task: T5.2
Document type: Deliverable D5.2
Title: Requirements Definition of Communication Platform
Subtitle: CommRob Project
Authors: TUW with support from KTH
Internal Reviewers: 
Version: v1.0, March 6, 2008
Status: Final
Distribution: Public
Summary

Having a communication model is important but certainly not sufficient for enabling a robot to communicate with humans (and other peer robots). In particular, our communication model is mode-independent (which also means that modellers do not have to include information about modalities), but there is something needed to operationalize these models and to enact them as multimodal dialogues.

We provide a high-level specification of the requirements on a communication platform to be developed in software, which is supposed to do exactly this. While this software will be able to handle discourses in general, additional software is needed for the specific “application logic” of this particular robot, so that it can fulfil the tasks given especially via communication at runtime. So, we provide a high-level specification of the requirements on this software as well.
# Table of contents

1. Introduction .................................................................................................................. 5  
   1.1 Purpose of the Document ....................................................................................... 5  
   1.2 Overview of the Document ................................................................................... 5  
2. Requirements on the Communication Component .................................................... 6  
   2.1 Discourse Processing Capabilities ......................................................................... 6  
   2.2 Interleaving of Discourse Execution .................................................................... 7  
   2.3 Multi-modality ....................................................................................................... 8  
      2.3.1 Modalities of the Robot Trolley ..................................................................... 8  
      2.3.2 Scenarios and Priorities of the Modalities ..................................................... 9  
      2.3.3 Applications and Different Modalities ......................................................... 11  
      2.3.4 Communication via Movement ..................................................................... 12  
      2.3.5 Trolley-Trolley Communication .................................................................. 12  
   2.4 Determination of Communicative Act Probability .................................................. 13  
3. Requirements on the Application Logic Component .................................................. 14  
   3.1 Robot Movement Tasks ....................................................................................... 15  
   3.2 Information Processing Tasks ............................................................................. 15  
   3.3 Mode Management ............................................................................................... 15  
4. Prerequisites ................................................................................................................ 17  
5. Conclusion and Outlook ............................................................................................... 18  
References .......................................................................................................................... 19
1. Introduction

1.1 Purpose of the Document

The main purpose of this deliverable document is to provide a high-level specification of the requirements on the communication platform to be developed in software. In fact, it contains even all the requirements on the top layer of the system architecture: communication platform and application logic. So, it defines what is needed from these components to enable the robot trolley to represent and enact communication models according to our communication approach, and to fulfil the tasks given especially via communication at runtime. Still, the high-level specification leaves room for including insights from the interplay with WP7 during the development in Task 5.3.

1.2 Overview of the Document

First, this document defines informally the requirements on the communication component of the top layer of the system architecture. Then it defines also the requirements on the application logic of the robot, which we separate explicitly from the general communication component.
2. Requirements on the Communication Component

The communication model described in Deliverable D5.1 [3] provides a basis for communication between human users and robots in CommRob. To enable the robot trolley to use this theoretical communication approach, the communication layer of the robot trolley (see Deliverable 2.1 [2]) will be designed as a communication platform together with an application logic. This platform processes the high-level communication, creates tasks executed in the lower layers, and handles events from the lower layers triggering further high-level communication. Thus, a number of principles have to be examined which are depicted in the following: The discourse processing capabilities, concurrency of discourse execution, multi-modality and the determination of communicative act probability.

2.1 Discourse Processing Capabilities

To explain the required capabilities it helps to imagine a complete shopping procedure where a user comes to the store, grabs a trolley, and starts processing his/her shopping list. The shopping list may result in a list of tasks that are to be elaborated during the shopping, i.e. the list of shopping items is translated into a list of “Go to” commands – not issued by the user directly, but by the application logic instead (see 3.2 for further explanation). The user will be led subsequently to the items by the trolley if the user does not pose any other task. Once a user has started a task and enacts a discourse, e.g. the user is led to a certain shopping item, the user may decide to pause that discourse for a while, instantiate a new task and enact that discourse, e.g. let the trolley follow the user. After that, the user may choose to resume to a pending task (and discourse) or start another task on the list.

This leads to some requirements on the task execution:

- Instantiation of new tasks,
- Resuming previous tasks, and
- Activation of tasks in a user-defined order must be possible.

If a user resumes a previous task, this means also to resume the associated discourse in the appropriate state.

User actions during a whole shopping procedure are too unpredictable to capture all of them within a single discourse. Modelling just one big shopping discourse would be too time-consuming for a modeller and likely lead to a hard comprehensible discourse. Therefore, we propose two levels of discourse processing:

- A high-level task list where each task triggers a discourse. A task list can be considered as a To-Do list or an agenda of discourses.
- Discourses that can be related to other discourses (see Deliverable D5.1).

Each task triggers the execution of a discourse and will be completed at the end of the discourse’s execution.

Tasks can be managed by a stack or a heap (see Figure 1). For the figure we assumed the following sequence of user commands: “Go to apples”, “Go to bananas”, “Follow me”, and “Meet me at oranges”. In this example, it is important that no task has been finished before a new one is issued. A stack (left part in Figure 1) simply staples one task on the other based on the order in which they
occur. The user influences this hierarchy as he/she operates the trolley – each user action instantiates a new task that is put on top of the stack. The stack is processed in a “last in first out” (LIFO) order – so in this case the last task “Meet me at oranges” is active while the remaining tasks are pending. A heap represents all tasks at the same level (right part in Figure 1). This allows the user to choose a certain task independently from the previously handled task (“Go to bananas” is activated here). Since a heap can be processed like a stack if required, but gives also more flexibility for task processing, we have chosen the heap as more appropriate for the representation of the agenda.

![Figure 1: Two different representations of a task agenda](image)

First, the discourse models need to be operational before they can be triggered. Therefore, the discourse models must be transformed into state machines. The instances of these state machines manage the flow of communication for their respective discourse.

### 2.2 Interleaving of Discourse Execution

The following relations between executed discourses and between their corresponding tasks can be identified:

- **Sequence:** The end of one task can trigger the execution of the next task on the agenda. The user may be asked which task he/she wants to process next. Pre- and post-conditions between corresponding discourses may have to be evaluated first.

- **Embedding:**
  - **Interruption:** The execution of one discourse is paused as long as another one is executed. The two discourses do not necessarily have a contextual relation between each other. Example: While the user is guided to an item, the user tells the trolley to follow him/her. Thus, the trolley stops the guiding and follows the user. After
following the user, the trolley continues to guide the user to the requested item. An interruption occurs at a task switch when the initial task was not yet finished.

- **Clarification**: During the execution of a dialogue, a clarification of a context, meaning, or some detail is required. The temporal dependency between the two dialogues is the same as described for interruption above, apart from that the two discourses are related and this kind of interruption is predictable.

- **Cancelling**: When a task is chosen from the agenda to be started, it may influence the currently running discourse to be interrupted or even cancelled. In the latter case, the corresponding task needs to be deleted from the agenda, too.

### 2.3 Multi-modality

In Deliverable D5.1 [3] a description of the communication model without the different modalities is given. As different modalities are aimed to bring up communication between a user and a robot trolley, this section presents the requirements for so-called multi-modality of the system. Whereas human users and robots communicate with such a multi-modal interface, communication between robots is done via message transfer in a wireless network. User-trolley as well as trolley-trolley communication is based on the same high-level communication model. It is worth mentioning that robot trolleys will not be able to communicate with gestures or speech between them. So, multi-modality is reserved for user-robot communication only. Additionally, the presentation of the different modalities requires their correlation to distinct modes. Thus, every mode has to be taken into consideration. See D2.1 for further details.

#### 2.3.1 Modalities of the Robot Trolley

According to [1], multimodal systems are systems, which offer the user combinations of input/output modalities for (or ways of) exchanging information with computer systems. These modalities may be used complementarily or interchangeably, which allows a more flexible style of interaction. The trolley within this project will support the following modalities for user input:

- **Speech input with a microphone array based on a defined and highly restricted English vocabulary**: It is used for the oral commandment via the microphone of the trolley (combined with gesture input according to the mode) and the menu selection for applications which are also visible on the touch screen.

- **Vision input with a camera via gesture recognition**: It is used for the gestural commandment of the trolley (combined with speech input).

- **A touch screen with a graphical user interface (GUI)**: This modality may be used for input of different applications, the selection of the aided mode, and switching speech input on or off. Additionally, no sophisticated recognition mechanism like signal processing is necessary to interpret the user input.

- **A haptic handle as a walking aid**: This device supports challenged or elderly people and allows them to steer the trolley with a minimum of required force. The support is given by adapting the trolleys’ movement to such a user. It is used by at least one hand and has three zones for pressure detection that allow steering left, steering right or pushing the trolley forward.
The trolley itself has the possibility to create output to the user:

- Speech output via loudspeakers with a limited English vocabulary. The user has always the choice to enable or disable the speech output by pressing a button on the touch screen.
- Interaction through movements of the trolley in 2D space.
- A touch screen provides textual and graphical output.
- *Optional:* Optical output with e.g. LEDs for warning signals

### 2.3.2 Scenarios and Priorities of the Modalities

There exist five different modes for a trolley, and these modes are mapped to scenarios (see their definition in Deliverable D2.1 [2]). Within each scenario, a number of modalities with differing priorities are used. Except the idle mode, all other modes require a so-called session to be opened which is initiated with the user-trolley attachment in the trolley parking zone. During the assignment, the user has to choose if he wants the trolley as a walking aid. This functionality may not be selected again as long as the trolley is not detached from the user and assigned again. The implicit switching between modes is based on the most elaborate state diagram in [2]. Thus, the user may not choose one mode deliberately, but by communicative acts, which are modality independent like “Follow me”. In the following a description with respect to the different modalities is given.

#### Trolley functionality with walking aid

When the user is assigned to the trolley with walking aid, it goes into the aided idle mode. In the aided idle mode there are two available commands: “Aided steering” and “aided guiding”.

The user may press a button on the haptic handle and starts pushing the trolley. With the robot functionality of the trolley, the locomotion itself is easier, as the user must not push the complete weight of the trolley and the goods. The trolley moves forward, with possibilities to turn right or left, according to the pressure the user puts on the position of the haptic handle. The locomotion of the trolley is aided by the hardware (motor). If the trolley is not pushed anymore and the button is released, the aided idle mode is reached again.

Another option to leave the idle state is the command “go to <target>” via speech input. When the haptic handle is grabbed, the button pressed and the speech input recognized by the software, the trolley starts to move and guides the user to the product. After arriving at the product, the trolley gives a speech and touch screen output to inform the user. It is possible to cancel the guided mode with the speech command “cancel” or by toggling the button. Within the aided idle mode, the user may interact with the touch screen to create a shopping list or select a distinct product manually. When this procedure is finished, the user grabs the handle, presses the button and the trolley starts the guiding mode while signalling information on the touch screen. The user may use the speech input for the “go to <target>” while being in the aided steering – with the handle grabbed and button still pressed.

Generally, if the product is recognized via speech input by the trolley, the guided mode is activated. Otherwise, the trolley makes use of the loud speaker and the touch screen to inform the user that the product has not been recognized. The implementation of a clarification dialogue with questions and answers via the touch screen or speech input/output to select the next target is optional. Gesture
recognition is not taken into consideration during the walking aided modes, as the aided mode requires the grabbing of the haptic handle with both hands.

**Trolley functionality without walking aid**

The idle mode is the starting mode of the functionality without walking aid. Within the unaided idle mode there are four available commands: *Pushing* the trolley, “meet me at \(<\text{target}\>\)”, “follow me” and “Go to \(<\text{target}\>\)”.

The user steers and moves the trolley by pushing it in any direction without the use of speech, gesture or touch screen input. By stopping to push the trolley, the idle mode is reached again. The following mode switches are possible during the steering mode with speech, gesture or touch screen input and possible clarification dialogues: “Follow me”, “meet me at \(<\text{target}\>\)” (not with gesture), “go to \(<\text{target}\>\)” (not with gesture). When the command input is successful, the user has to release the handle and the trolley starts moving alone (according to the chosen mode alone or with the user side-by-side).

The “go to \(<\text{target}\>\)” mode requires an according input via the microphone or the touch screen. A clarification dialogue may be necessary if the speech input is not appropriately recognized. When the trolley has processed the command, it starts moving to the next target product in a cautious way, not to surprise the user. During the guiding mode the trolley is driving alone to the next product and the user walks near the trolley without pushing the trolley. The vision system combined with the inner logic of the trolley controls that the user and the trolley are close together. The trolley adapts its speed to the one of the user, if the user stops, the trolley stops too. When the user moves on, the trolley starts its movement to the aimed product as well. During the guiding mode the user may use gestures or speech input to control the trolley like “cancel”, “stop” or “move on”. After the trolley reaches its target, it automatically slows down and informs the user via speech output and a message on the touch screen. If there where no other targets (e.g. shopping list checked off) the trolley switches to idle mode, otherwise it asks the user via speech output and touch screen if it should move on. Now the user may use gestures, touch screen or speech input to respond. In the guiding mode, it is possible to switch to the autonomous mode (not with gesture) or following mode (“Follow me”) with speech and touch screen input and possible clarification dialogues. Pushing the trolley during the guiding mode leads to the steering mode again.

Entering the “follow me” mode requires an according input via the microphone, a gesture or the touch screen. A clarification dialogue may be necessary if speech input is not appropriately recognized. When the trolley has processed the command, it awaits the movement of the user to follow him in a pre-defined distance. When the trolley is near a shopping-list product, it slows down and informs the user via speech and touch output about this fact. If the user moves too fast, and the trolley is not able to follow him, it stops moving and awaits the return of the user. Additionally, it signals the user via speech and touch screen output to stay closer. It is possible to switch to the guiding or autonomous mode (via speech or touch screen) or to give an order to stop (via speech, touch screen or gesture).

Entering the “autonomous” mode requires an according input via the microphone or the touch screen. A clarification dialogue may be necessary. Afterwards, the user and the trolley move independently to the aimed product. When they meet again (recognized via user tracking), the trolley signals this with speech and output on the touch screen. The autonomous mode may be cancelled with gesture, speech or using the touch screen. If the user and the trolley are close to each other, but not at the aimed product, the trolley continues to stay in the autonomous mode as long as the user provides no
additional commands. The user may cancel the autonomous mode with speech, touch screen or gestural input.

A resuming functionality during the “follow me”, “guiding” or “autonomous” mode is necessary. The user may have to steer the trolley via the handle because of a cluttered area. When leaving the handle, the trolley shall return to the previous mode – as long as the user does not cancel this distinct previous mode with an according speech or touch screen input. The touch screen, speech or gesture input may be used for “pausing” and “resuming”.

According to the specification, there are a number of limitations in the different modes regarding the possible usage of the modalities. The user may use all modalities to express communicative acts while using the trolley. However, sometimes the usage of distinct modalities does not have any effects (see below).

**Touch screen:** The touch screen is always usable for input and output of information.

**Speech:** Speech is always usable for input and output of information.

**Gesture:** Gesture input is not usable in Aided Idle, Aided Steering and Aided Guiding Mode.

**Trolley movement:** Communication via Trolley movement (motor) is not used in Aided Idle and Idle Mode.

**Haptic input:** The haptic input is always usable. In Aided Steering and Aided Guiding Mode, it is mandatory.

### 2.3.3 Applications and Different Modalities

The trolley’s system software consists of the following components, which are usable with different modalities.

- **Shopping list:** It allows the administration of a list of products a user wants to buy. It is mainly used with the touch screen, i.e. menu selection and choosing of products to add or delete product items from the list. Furthermore, the speech input may be used for menu selection interchangeably with the touch screen. Product item selection and deletion on the shopping list is done with the touch screen. As an alternative to the shopping list, the user has the choice to directly select a product of the supermarket as the next target of the trolley. Targets where the trolley has to move next are not only buyable products, but the supermarket check-out as well. This is defined within the ontology contained in the systems’ repository.

- **Test application:** It allows the testing of the different modalities to check the trolleys’ multi-modal functionality. It requires the user to interact with the touch screen, the speech system, the gesture recognition and the haptic handle with a well-defined test chain. Additionally, it produces sound to test the loudspeaker and tries to move the trolley in a defined manner.

Of course, the touch screen might as well be used for any kind of game applications (like the well known game “memory”), which helps to make the waiting time at the cash endurable. However, this is not in the focus of the project.
2.3.4 Communication via Movement

A major research topic in the CommRob project is communication via movements. Hence, not only the user is able to make gestures, but the robot trolley may use movement in 2D-space (the floor of the shopping centre) for an intuitive and user-friendly way of communication. In [3], one can see the general discourse model together with a sample description for movement as part of user-robot interaction. “Movement” itself can be understood as the technical locomotion of the trolley, driven by its engine and controlled by the distinct piece of steering software (with the help of the trolley’s sensors). Additionally, the “Movement” gets a meaning in the concept domain of the users and trolleys, based upon the communication model as presented in [3].

Another approach for movement as part of human-robot interaction is depicted in [4]. It describes a scenario where human-robot and robot-robot communication takes place based on writing text and movement of users and robots.

2.3.5 Trolley-Trolley Communication

Multi-modality with speech, touch screen, movement, and gestures is intended for communication between a user and a trolley exclusively. Trolleys, on the other hand, communicate with each other via WLAN (Wireless Local Area Network) with TCP/IP (Transmission Control Protocol / Internet Protocol) as a connection-oriented transport protocol on top of WLAN. Trolley-trolley communication based on this medium, which is solely intended for machine communication, thus can be considered as another modality.

Generally, the communication component (refer to the logical architecture in D2.1) is not restricted to external trolley communication via WLAN and TCP/IP – in fact any medium could be chosen. Though, in this project WLAN and TCP/IP are preferred.

On a higher level, communicative acts as presented in [3] can be formatted and transformed to a e.g. mark-up language application of XML (Extensible Markup Language) and simply sent as text portions between the communicating trolleys. Thus, trolley-trolley communication itself is not multi-modal, but it adds a distinct modality to the domain with two trolleys as communicating parties.

To show a possible communication example between two trolleys the following scenario describes collision avoidance with another trolley:

While two trolleys are moving towards each other they have to decide which one will give way and in which direction. Maybe they also need to resolve a narrow path situation (bottleneck) where one has to go backwards to resolve a blocking situation.

Trolley A requests Trolley B for collision avoidance (compare Figure 2: upper left Request “move aside”) and Trolley B accepts. Trolley A asks the preferred side (ClosedQuestion), Trolley B chooses “right” or “left” and moves aside, informing Trolley A that it has moved to the chosen side (left tree of the Otherwise relation). If they meet at a narrow place where no side-movements would help resolving the conflict, Trolley B has to reject the request to move aside. Then Trolley A requests Trolley B to move backwards (right tree of the Otherwise relation). If Trolley B accepts it informs Trolley A after its movement that is has moved backwards. If e.g. there is no way to move backwards Trolley B rejects the request.

Not shown in Figure 2: Trolley B could then propose to Trolley A to move backwards.
### 2.4 Determination of Communicative Act Probability

The use of different modalities for user-robot interaction requires a mechanism to determine the probability of communicative acts based on the speech and gesture hypotheses (“High-level sensor fusion”). Underlying software functionality delivers the list of possible recognized commands with a probability for each command in different modalities, i.e. vision for gesture and speech. The input via the touch screen can be seen as non-ambiguous and the haptic handling is controlled from the lower software layers – thus it is not taken into consideration within the probability calculation.

After the probability for the gesture or speech input is provided to the communication layer, a decision-making process is triggered. It is intended to use a software solution based on the Bayes’ theorem and Hidden Markov Model which relates conditional probabilities. According to the artificial intelligence community it is used to draw conclusions in domains of uncertain knowledge. Hence, these conclusions are not deductive (thus not always correct), but in real-world examples they have been shown to be quite helpful for building hypotheses.

Generally, when the recognition system of the trolley is not able to recognize the input several times, the trolley does not use this recognition channel any more during the following interactions with the user. A noisy environment or a vague and too quiet articulation of the use may cause this. The trolley signals this behaviour with speech and touch screen output. Nevertheless, the user has the choice to re-enable this input channel via the touch screen.
3. Requirements on the Application Logic Component

The application logic component – together with the communication platform – is needed to fulfill the tasks given via communication at runtime. Generally, it has two functionalities which are independent from each other – called tasks:

- High-level movement coordination of the trolley
- Information processing (like Shopping List processing)
- Mode Management

These make use of different interfaces of the Communication Layer to the repository and the lower layer. The mode management is depicted which guarantees the correct administration of the states and transitions of the (virtual) state machine which is presented in D2.1.

The world model repository has to be accessed by the application logic components in general. It consists of the domain model, the ontology of the products and the ontology of the actions.

![Application Logic Overview](image-url)
Figure 3: Application Logic Overview gives a short overview of the application logic. Actions are used by the Robot Movement Task components and the Information Processing Task components. The third component (Mode Management) is triggered from the Robot Movement Task. These three components are described below.

### 3.1 Robot Movement Tasks

The actual task of the trolley is triggered by the user via communicative acts (taken clarification dialogues under consideration). Such a task leads to a movement of the trolley. Hence, Follow-Me, Go To and Meet Me are possible via different modalities. The robot movement logic uses now the interface to the middle layer (see [2] for further details). So, the movement coordination of the robot is administered in the top layer and the detailed processing happens at lower layers. Moreover, the top layer transforms input from the user in logical addresses, like “apples” to coordinates x and y.

The following robot movement tasks are available (aided or not aided):

- Follow me
- Go To <target>
- Meet me at <target>

These tasks are maintained in a task list (see 2.1 for stack and heap) which states a memory for tasks, too.

### 3.2 Information Processing Tasks

Aside from the trolley movement there are information processing tasks like the shopping list. The shopping list is part of the application logic of CommRob regarding the top layer without triggering robot functionality. The list contains products of the supermarket. It does not make use of the previous mentioned state diagram. Thus, no state transitions happen which are common within the mode management. The interface to the middle layer is not used by the shopping list component.

The interface of the shopping list allows the creation of a new shopping list, the adding of a product, the deletion of a product, showing the actual list and providing information of a distinct product in the shop. Several shopping lists in parallel are not possible – only one list exists at a time. The actual shopping list is a variable in the application logic. Each product in the list variable leads to a movement task (see task list in 3.1), and it is the trolleys’ job to guide the user to the product. Hence, this variable (which may be a distinct data structure) is required by e.g. the calculation of the optimized path, as the next product implicitly defines the upcoming movements as it commands a Go-To to the robot movement task.

Another information processing tasks is the commandment of actions like the modification of the path when the user changes the route. However, these tasks require access to the middle layer.

### 3.3 Mode Management

The modes and their transitions – described in a state chart in deliverable D2.1 – need an operationalisation in terms of a finite state machine that takes care for the management. Therefore, those communicative acts from the user lead to transactions which create movement tasks (see 3.1) that affects the mode management inside of the application logic component. Apart from that also...
events from lower layers of the trolley, like pushing the trolley or completing the setup, have to be reported upwards in order to be able to manage the mode’s state.

An important fact is that no memory for modes will be maintained—the mode management’s only responsibility is to deal with allowed transitions between modes and it does not track inactive modes in the past. Therefore, neither a hierarchical state machine nor a similar approach is required. The only memory that is needed is one for the tasks, since running tasks may be interrupted and retrieved by the user. A task list (see 2.1 and 3.1) already maintains all active or unfinished tasks.

Movement tasks and modes are related to each other as follows: Several certain tasks can be executed in the same mode. E.g.: “Go to apples” and “Go to bananas” are both executed within the guiding mode. However, the same task cannot be executed in different modes. The following example should explain why we need a memory for tasks but none for modes: The user issues the command “Go to apples” which the trolley will execute in the guiding mode. The next user command (while the execution of “Go to apples” is still in progress) is “Go to bananas” (also in guiding mode). Before “Go to bananas” is finished the user says “Follow me” which will be executed in the following mode. After a while the user commands “Stop following me”. If no tasks are pending, the trolley would wait for new commands in the idle mode. But since “Go to apples” and “Go to bananas” still are not finished yet the task execution should ask which of these two tasks the user wants to proceed next. Only resuming to the guiding mode would be ambiguous because that would not select a certain task. The selection of a task from the task list triggers the mode management as if the user would have issued the same command and it switches to the appropriate mode.

Attempted transitions that are not allowed in a certain mode have to trigger dialogs, which inform the user about the reason for not following his/her request.

The mode switching itself happens implicitly, as the mode selection is hidden from the user. He/she just acts with the trolley via the different communicative acts and the according mode is selected. The internal state switching is explicit. Nevertheless, the mode switch itself and the actual mode will be communicated to the user. This matter will be addressed in the next working phase through user studies and interaction design related activities.
4. Prerequisites

The implementation of the trolley’s communication platform and its requirements poses some prerequisites. First, it needs a defined interface to the Operations Layer, responsible for executing the tasks, and second, a WLAN interface for communicating with the communication platform of the other trolleys is needed.

- **Interfaces provided by the Operations Layer:** The functionality of the interface is described in detail in [2]. The interface allows sending commands to the Operations Layer and receiving events. From technical perspective, we will communicate via sockets or equal mechanisms. This allows loose coupling and gives each layer the opportunity to use programming languages and obey other constraints important to the layer.

- **WLAN for Trolley-Trolley Communication:** Trolley-trolley communication will be handled via WiFi networks. Therefore, the communication platform requires an IP network for exchanging communicative acts between the components of the communication platform on each trolley. The communication platform will allow to use different transport protocols on top of the WiFi network infrastructure.
5. Conclusion and Outlook

This report on the requirements definition of the communication platform defines the software part of the robot that will make use of discourse models according to our approach (as given in D5.1) for letting it actually engage in communication. We separate here a general-purpose communication platform component from a specific application logic component.

Since our communication approach as extended for robotics applications involves clarification dialogues, especially for the speech and gesture modalities, key requirements for the platform are to handle discourses in parallel, to allow their interruptions, and to cancel and embed them.

We try to integrate different modalities for communication between the trolley and the user according to our discourse models. This requires the platform to first prioritize modalities according to (the amount of) information to be conveyed and environment conditions (noise), as well as the kind of tasks (e.g., shopping list management versus navigation task). In addition, the platform has to merge utterances in different modalities for better understanding and clarification. Finally, the modalities also depend on the mode that the trolley is currently in.

A key innovation in the CommRob project is communication with the user also via movement of the trolley (in 2D space). Movement is used mainly to emphasize utterances of the robot and to gain attention.

The application logic has to fulfill three major tasks that are initiated through communication and events from the lower layers:

- High-level movement coordination of the trolley,
- Information processing (like Shopping List processing), and
- Mode Management.
References


