SCOPE – CARE II Innovative

WP2 – R2 – Interface and architecture specification

Nicolas DEHERLY & Alexandre LEMORT

<table>
<thead>
<tr>
<th>Source</th>
<th>Version</th>
<th>Date</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IntuiLab</td>
<td>2.1</td>
<td>2004-11-17</td>
<td>1/28</td>
</tr>
<tr>
<td>Reference</td>
<td>TRT-Fr/DAS/HIT/OG,04/127</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


This document is published by EUROCONTROL in the interests of the exchange of information. It may be copied in whole or in part, providing that this copyright notice and disclaimer are included. The information contained in this document may not be modified without prior written permission from EUROCONTROL.

EUROCONTROL makes no warranty, either implied or express, for the information contained in this document, neither does it assume any legal liability or responsibility for the accuracy, completeness or usefulness of this information.
## Versions

<table>
<thead>
<tr>
<th>Number</th>
<th>Date</th>
<th>Writing</th>
<th>Review</th>
<th>Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2004-09-29</td>
<td>N. DEHERLY &amp; A. LEMORT (INTUILAB)</td>
<td>M. BROCHARD (EUROCONTROL)</td>
<td>A. LEMORT (INTUILAB)</td>
</tr>
<tr>
<td>2.0</td>
<td>2004-10-22</td>
<td>N. DEHERLY &amp; A. LEMORT (INTUILAB)</td>
<td>M. BROCHARD (EUROCONTROL)</td>
<td>O. GRISVARD (THALES R&amp;T)</td>
</tr>
<tr>
<td>2.1</td>
<td>2004-11-17</td>
<td>N. DEHERLY &amp; A. LEMORT (INTUILAB)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Circulation

<table>
<thead>
<tr>
<th>Addressee</th>
<th>Version</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marc BROCHARD (EUROCONTROL)</td>
<td>1.0</td>
<td>2004-09-30</td>
</tr>
<tr>
<td>Marc BROCHARD (EUROCONTROL)</td>
<td>2.0</td>
<td>2004-10-25</td>
</tr>
<tr>
<td>Marc BROCHARD (EUROCONTROL)</td>
<td>2.1</td>
<td>2004-11-17</td>
</tr>
</tbody>
</table>

## Validation

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Signing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Written by:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N. DEHERLY, &amp; A. LEMORT</td>
<td>2004-11-17</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Signing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Source

<table>
<thead>
<tr>
<th>Source</th>
<th>Version</th>
<th>Date</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IntuiLab</td>
<td>2.1</td>
<td>2004-11-17</td>
<td>2/28</td>
</tr>
</tbody>
</table>

Reference: TRT-Fr/DAS/HIT/OG,04/127


This document is published by EUROCONTROL in the interests of the exchange of information. It may be copied in whole or in part, providing that this copyright notice and disclaimer are included. The information contained in this document may not be modified without prior written permission from EUROCONTROL.

EUROCONTROL makes no warranty, either implied or express, for the information contained in this document, neither does it assume any legal liability or responsibility for the accuracy, completeness or usefulness of this information.
Table of Contents

1. Introduction .......................................................................................................................... 0
   1.1. Purpose of this document .............................................................................................. 0
   1.2. SCOPE objectives .......................................................................................................... 0
   1.3. Document conventions .................................................................................................. 0
2. Operational scenarii ............................................................................................................. 0
   2.1. Watermarking and callsigns recognition ........................................................................ 0
       2.1.1. Principle .................................................................................................................. 0
       2.1.2. Scenarii ...................................................................................................................... 0
   2.2. Data input monitoring .................................................................................................... 0
       2.2.1. Principle ...................................................................................................................... 0
       2.2.2. Scenarii ...................................................................................................................... 0
   2.3. Clearance conflict alert .................................................................................................. 0
       2.3.1. Principle ...................................................................................................................... 0
       2.3.2. Scenarii ...................................................................................................................... 0
3. Architecture overview ......................................................................................................... 0
4. SCOPE agents ..................................................................................................................... 0
   4.1. Controller and pilot A.S.R agents ................................................................................. 0
   4.2. Radar display agent ....................................................................................................... 0
       4.2.1. A.S.R. state plug-in .................................................................................................. 0
       4.2.2. Watermarking plug-in ............................................................................................ 0
       4.2.3. Clearance conflict plug-in ...................................................................................... 0
   4.3. Air Traffic Simulator agent ........................................................................................... 0
5. Communication protocol ..................................................................................................... 0
   5.1. Communication protocol of the Controller ASR agent .................................................. 0
       5.1.1. State of the ASR agent ............................................................................................. 0
       5.1.2. Interpretations .......................................................................................................... 0
       5.1.3. Examples of speech interpretations ......................................................................... 0
   5.2. Communication protocol of the Pilot ASR agent ........................................................... 0
       5.2.1. State of the ASR agent ............................................................................................. 0
       5.2.2. Interpretations .......................................................................................................... 0
       5.2.3. Examples of speech interpretations ......................................................................... 0
   5.3. Communication protocol of the radar display agent ........................................................ 0
       5.3.1. Vectoring instructions .............................................................................................. 0
WP2 - R2 - Interface and architecture specification

SCOPE – CARE II Innovative

5.3.2. Flight level instructions ................................................................................................................. 0
5.3.3. Speed control instructions ........................................................................................................... 0
5.4. Communication protocol of the Air Traffic Simulator agent .......................................................... 0
6. Operational environment ....................................................................................................................... 0
6.1. Operational environment of the Controller ASR agent ..................................................................... 0
6.2. Operational environment of the Pilot ASR agent ............................................................................... 0
6.3. Operational environment of the radar display and ATC simulator agents ........................................ 0
6.4. Other requirements ............................................................................................................................. 0
7. References ............................................................................................................................................... 0
8. Glossary .................................................................................................................................................. 0

Source | Version | Date       | Page |
--------|----------|------------|------|
IntuiLab| 2.1      | 2004-11-17 | 4/28 |
Reference | TRT-Fr/DAS/HIT/OG,04/127 |        |      |


This document is published by EUROCONTROL in the interests of the exchange of information. It may be copied in whole or in part, providing that this copyright notice and disclaimer are included. The information contained in this document may not be modified without prior written permission from EUROCONTROL.

EUROCONTROL makes no warranty, either implied or express, for the information contained in this document, neither does it assume any legal liability or responsibility for the accuracy, completeness or usefulness of this information.
1. Introduction

1.1. Purpose of this document

This document is the second work-package deliverable report of the CARE II innovative project SCOPE (Safety of Controller Pilot dialogue). The work presented has been done by IntuiLab and is based upon the work of Thales Research of Technology (Thales R&T) and IRIT (Institut de Recherche en Informatique de Toulouse).

The SCOPE project will result in a demonstrator (work-package 4) which will present the results of speech tracking in controller-pilot vocal dialog to the controller. This report presents the architecture and the user interfaces specifications proposed by IntuiLab for this demonstrator.

This document provides a high level overview of the SCOPE demonstrator and describes the relationships, interactions, and basic functionality of SCOPE. It only covers the first level of decomposition of SCOPE into its main components. It SHOULD be used as a roadmap to learn: (1) what SCOPE is, (2) what problems SCOPE solves, and (3) core SCOPE functionalities and architecture.

1.2. SCOPE objectives

Communication errors between pilots and Air Traffic Controllers have always been a major safety issue. Very often, those errors result from misunderstandings between the pilot and the Air Traffic Controller during radio conversations. For many, the ability to converse freely with ATC systems represents the ultimate challenge to improve the efficiency and safety of the global man-machine system. Unfortunately, the time and knowledge costs of building even simple speech systems for ATC make it difficult for designers to cover the ATC phraseology.

In order to manage risk and make the best use of available resources, the SCOPE project focuses on a specific subset of en route phraseology such as callsigns, vectoring instructions, flight levels and speed control.

The goal of the SCOPE project is to illustrate how a voice recognition agent introduced as a third-party in the controller-pilot voice communication channel can improve the efficiency and safety of the overall system by capturing parts of the communication, and using the result to display significant/relevant information to the controller.

1.3. Document conventions

The following conventions are used throughout this document:

<table>
<thead>
<tr>
<th>Source</th>
<th>Version</th>
<th>Date</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IntuiLab</td>
<td>2.1</td>
<td>2004-11-17</td>
<td>5/28</td>
</tr>
<tr>
<td>Reference</td>
<td>TRT-Fr/DAS/HIT/OG,04/127</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


This document is published by EUROCONTROL in the interests of the exchange of information. It may be copied in whole or in part, providing that this copyright notice and disclaimer are included. The information contained in this document may not be modified without prior written permission from EUROCONTROL.

EUROCONTROL makes no warranty, either implied or express, for the information contained in this document, neither does it assume any legal liability or responsibility for the accuracy, completeness or usefulness of this information.
2. Operational scenarios

Our approach for the SCOPE project consists in two main tasks for the first year of the project:

- The goals of the first task are to identify and analyze the most appropriate tools for voice recognition in Air Traffic Control (ATC). Results consist in WP1-R1a and WP1-R1b
- The second task consists in identifying a candidate application and prototyping a first demonstrator.

A brainstorming session about the applications of voice recognition by Air Traffic Controllers is organized at IntuiLab in order to identify a candidate application. Participants include IntuiLab, Thales R&T and IRIT. After the brainstorming session, three ideas have been selected and illustrated by small scenarios:

- **idea 1**: watermarking and callsign recognition
- **idea 2**: data input monitoring
- **idea 3**: clearance conflict alert

At the first progress meeting (PM1), two development options were proposed: idea 1 and 2 or idea 1 and 3. M. Brochard, the CARE INO project manager, chose the second option since it is more challenging for the recognition software, which is more in line with the objectives of the SCOPE project.

2.1. Watermarking and callsigns recognition

2.1.1. Principle

Watermarking is a short signature at the beginning of a transmission positively authenticating the speaker. This fully automated identification mechanism can simplify communication in the ATC sector and increase safety. Voice recognition is used to detect conflict between the callsign in the controller's clearance and the callsign sent by the aircraft via watermarking.

This feature is expected to provide benefits on safety.

2.1.2. Scenarii

In the following scenarii, four actors intervene (two human users and two software agents):

- an user playing the role of an air traffic controller,
- an user playing the role of a pilot,
- a controller A.S.R. agent listening to controller's instructions,
- a software agent displaying feedbacks on the controller working position

In these scenarii, the controller asks AFR440 to climb to flight level 280.

21.2.1 Scenario 1: Response of the called pilot

**Controller:** "AFR440, climb to level 280"
(The controller A.S.R. agent understands AFR440)
(A waiting feedback is displayed on the track of AFR440 on the controller working position)

**Pilot:** "Climbing level 280, AFR440"
(The callsign sent via watermarking matches the result of the controller A.S.R. agent)
(A matching feedback is displayed on the track of AFR440 on the controller working position)

21.2.2 Scenario 2: Response of another pilot

**Controller:** "AFR440, climb level 280"
(The controller A.S.R. agent understands AFR440)
(A waiting feedback is displayed on the track of AFR440 on the controller working position)

**Pilot:** "Climbing level 280, AFR450"
(The callsign sent via watermarking does not match the result of the controller A.S.R. agent)
(Two feedbacks are displayed on the controller working position: one on the track of AFR450 and one on the track of AFR440)

21.2.3 Scenario 3: The callsign recognized does not exist

**Controller:** “AFR440, climb level 280”
(The controller A.S.R. agent understands AFR400 which does not exist)
(A feedback is displayed on the controller working position to indicate that the recognized callsign does not exist.)

21.2.4 Scenario 4: Recognition of the callsign fails

**Controller:** "AFR440, climb level 280"
(The controller A.S.R. agent fails to identify a callsign)
(A feedback is displayed on the controller working position to indicate that the controller A.S.R. agent has failed.)

2.2. Data input monitoring

2.2.1 Principle

Voice recognition is used to fill data input form with recognized values before the controller validate or modify them.
2.2.2. Scenarii

In the following scenarii, three actors intervene (one human user and two software agents):
- an user playing the role of an air traffic controller,
- a controller A.S.R. agent listening to controller's instructions,
- a software agent displaying feedbacks on the controller working position

In these scenarii, the controller asks AFR440 to climb to flight level 280 and turn heading 350.

22.2.1 Scenario 1: good recognition of the clearance

Controller: "AFR440, climb level 280 and turn heading 350"
(The controller A.S.R. agent understands Flight level 280 and heading 350)
(The recognized values are displayed on the controller working position)

22.2.2 Scenario 2: wrong recognition of the value

Controller: “AFR440, climb level 280 and turn heading 350”
(The controller A.S.R. agent understands Flight level 260 and heading 350)
(The recognized values are displayed on the controller working position)
(The controller can correct the wrong value by clicking on it)

22.2.3 Scenario 3: wrong recognition of the parameter

Controller: “AFR440, climb level 280 and turn heading 350”
(The controller A.S.R. agent understands Flight level 280 and IAS 350)
(The recognized values are displayed on the controller work position)
(The controller must close the data input form and open a new one)

22.2.4 Scenario 4: Voice recognition fails

Controller: "AFR440, climb level 280 and turn heading 350"
(The controller A.S.R. agent fails to identify an order)
(The system indicates that the ASR has failed)

2.3. Clearance conflict alert

2.3.1. Principle
Voice recognition is used to detect conflict between the controller's clearance and the pilot's acknowledgement.

This feature is expected to provide benefits on safety

### 2.3.2. Scenarios

In the following scenarios, five actors intervene (two human users and three software agents):
- an user playing the role of an air traffic controller,
- an user playing the role of a pilot,
- a controller A.S.R. agent listening to controller's instructions,
- a pilot A.S.R. agent listening to pilot's acknowledgements,
- a software agent displaying feedbacks on the controller working position

In these scenarios, the controller asks AFR440 to climb to flight level 280.

#### 23.2.1 Scenario 1: Acknowledgement matches clearance

**Controller:** "AFR440 climb level 280"
(The controller A.S.R. agent understands Flight Level 280)

**Pilot:** "Climbing level 280, AFR440"
(The pilot A.S.R. agent understands Flight Level 280)
(The acknowledgement matches the clearance)
(A matching feedback is displayed on the track of AFR440 on the controller working position)

#### 23.2.2 Scenario 2: Mismatch on the value

23.2.2.1 Case 1: the pilot makes a mistake

**Controller:** "AFR440 climb level 280"
(The controller A.S.R. agent understands Flight level 280)

**Pilot:** "Climbing level 260, AFR440"
(The pilot A.S.R. agent understands Flight level 260)
(The acknowledgement does not match the clearance.)
(An alert popup is displayed near the track of AFR440 on the controller working position)

[NOTE: This case corresponds to a true error]

23.2.2.2 Case 2: the controller A.S.R. misunderstands the clearance

**Controller:** "AFR440 climb level 280"
(The controller A.S.R. agent understands Flight level 280)

**Pilot:** "Climbing level 280, AFR440"
23.2.2.3 Case 3: the pilot A.S.R. misunderstands the acknowledgement

**Controller**: "AFR440 climb level 280"
(The controller A.S.R. agent understands Flight level 280)
**Pilot**: "Climbing level 280, AFR440"
(The pilot A.S.R. agent understands Flight level 260)
(The acknowledgement does not match the clearance.)
(An alert popup is displayed near the track of AFR440 on the controller working position)

[NOTE: This case illustrates a false error notification]

23.2.2.4 Case 4: Controller and pilot A.S.R. make mistakes

**Controller**: "AFR440 climb level 280"
(The controller A.S.R. agent understands Flight level 260)
**Pilot**: "Climbing level 280, AFR440"
(The pilot A.S.R. agent understands Flight level 240)
(The acknowledgement does not match the clearance.)
(An alert popup is displayed near the track of AFR440 on the controller working position)

[NOTE: This case illustrates a false error notification.]

23.2.3 Scenario 3: Mismatch on the parameter

23.2.3.1 Case 1: The pilot makes a mistake

**Controller**: "AFR440 climb level 280"
(The controller A.S.R. agent understands Flight level 280)
**Pilot**: "Turning heading 280, AFR440"
(The pilot A.S.R. agent understands Flight level 280)
(The acknowledgement does not match the clearance.)
(An alert popup is displayed near the track of AFR440 on the controller working position)

[NOTE: This case corresponds to a true error]
Controller: "AFR440 climb level 280"
(The controller A.S.R. agent understands Speed 280 knots)

Pilot: "Turning heading 280, AFR440"
(The pilot A.S.R. agent understands Flight level 280)
(The acknowledgement does not match the clearance)
(An alert popup is displayed near the track of AFR440 on the controller working position)

[NOTE: This case illustrates a false error notification.]

23.2.3.3 Case 3: The pilot A.S.R. misunderstands the acknowledgement

Controller: "AFR440 climb level 280"
(The controller A.S.R. agent understands Flight level 280)

Pilot: "Turning heading 280, AFR440"
(The pilot A.S.R. agent understands Speed 280 knots)
(The acknowledgement does not match the clearance)
(An alert popup is displayed near the track of AFR440 on the controller working position)

[NOTE: This case illustrates a false error notification.]

23.2.3.4 Case 4: Controller and pilot A.S.R. make mistakes

Controller: "AFR440 climb level 280"
(The controller A.S.R. agent understands Turn left 280)

Pilot: "Turning heading 280, AFR440"
(The pilot A.S.R. agent understands Speed 280 knots)
(The acknowledgement does not match the clearance)
(An alert popup is displayed near the track of AFR440 on the controller working position)

[NOTE: This case illustrates a false error notification.]

23.2.4 Scenario 4: Voice recognition fails

23.2.4.1 Case 1: The controller A.S.R. fails to identify an order

Controller: "AFR440, climb level 280"
(The controller A.S.R. agent fails to identify an order)
(A feedback is displayed on the controller working position to indicate that the controller A.S.R has failed.)

23.2.4.2 Case 2: The pilot A.S.R. fails to identify an order
Controller: "AFR440, climb level 280"
(The controller A.S.R. agent understands Flight level 280)

Pilot: "Turning heading 280, AFR440"
(The pilot A.S.R. agent fails to identify an order)
(A feedback is displayed on the controller working position to indicate that the pilot A.S.R has failed.)

3. Architecture overview

IntuiLab proposes an architecture for the SCOPE demonstrator based upon a multi-agent approach. The whole application is decomposed into several software agents that exchange textual messages on software bus: Ivy.

Some of the advantages of the proposed multi-agent approach are:
- It permits to decompose easily the development work between the different actors of the project by separating the tasks into several software agents easily integrated to the project.
- It is an open software architecture that can be easily extended for others functionalities or modalities by connecting new software agents on the Ivy bus.

**Ivy Bus**

*Figure 1: SCOPE is divided into four software agents connected by an Ivy software bus*
As shown on Figure 1, the SCOPE demonstrator is made of four software agents. The functions of each of these agents are described more precisely in the paragraph SCOPE agents. Each of these agents is integrated to SCOPE using the Ivy software bus. The behaviors of the agents are controlled by the messages received from the Ivy bus as well as the user inputs. The paragraph Communication protocol describes the textual messages exchanged between agents and the corresponding behaviors.

The Ivy software bus supports inter-applications communications across a network as well as on a single computer. It is available in multiple programming languages (Java, C/C++, Perl, VB, etc.) and can run on most of the common systems (hardware as well as operating systems). It is developed and maintained by the CENA (Centre d'Etude de la Navigation Aérienne) as an open source project. Its first goal is to allow and to simplify the quick development of communicating software agents and prototypes in human-computer interaction research. It is used by IntuiLab, IRIT and Thales R&T for multimodal projects since it helps easy integration of agents and extensions.

[NOTE: To learn more about the Ivy software bus see http://ww.tls.cena.fr/products/ivy/]

4. SCOPE agents

As shown in the Architecture overview (section 3), the demonstrator is divided into four communicating agents running on three computers. This section provides a high-level overview of how the functionalities and responsibilities of the system are partitioned and how the individual parts work together.

4.1. Controller and pilot A.S.R agents

The A.S.R. agents used for the controller-pilot dialog recognition are provided by Thales R&T. They are based on Nuance speech recognition engine (see WP1-R1A). The A.S.R. permits to make expression spotting in the dialog between the controller and pilots. The A.S.R. aims at detecting relevant information in the controller or pilot speech based on an air traffic control specialized grammar created by IRIT. The expressions spotted by the A.S.R. are transcribed into textual messages. For making the inter-agents communication easier, the messages are sent on the Ivy bus to other software agents according to a format described in the Communication protocol paragraph.

[NOTE: To learn more about Nuance speech recognition engine see http://www.nuance.com]

4.2. Radar display agent

The radar display agent of SCOPE is based on Twinkle, a radar information visualization application. It displays air traffic information sent on the Ivy software bus by an air traffic simulator. This tool relies on an open architecture which permits to plug new rendering components that are displayed on the radar image. Three rendering components are added to Twinkle for the SCOPE needs:
WP2 – R2 – Interface and architecture specification

- A plug-in which gives a feedback on the state of the controller $A.S.R.$ agent and the pilot $A.S.R.$ agent
- A watermarking plug-in
- A clearance-acknowledgement match plug-in

![Figure 2: Twinkle radar image visualization agent](image)

4.2.1. **A.S.R. state plug-in**

The $A.S.R.$ state plug-in gives a feedback on the actual state of the two $A.S.R.$ agents connected on the $Ivy$ bus. It is composed of two parts (one dedicated to the controller $A.S.R.$ agent and one dedicated to the pilot $A.S.R.$ agent). The possible states of each part are:

- $A.S.R.$ is waiting for speech corresponding to the grammar
- $A.S.R.$ has recognized the beginning of an expression matching the grammar and should return a message for this expression
- $A.S.R.$ recognition has failed because the expression did not really match the grammar or the recognition engine can not interpret the vocal input
- $A.S.R.$ returned a message which is coherent, i.e. it respects the format defined by the communication protocol and the values are plausible
- $A.S.R.$ returned a message which is not coherent.

The three first states correspond to real $A.S.R.$ states, the last two ones are relative to the consistency of the messages sent by the $A.S.R.$. The consistency is checked by the two other plug-ins.
Waiting speech

Waiting message

Is recognition successful?

Yes

Is message consistent?

Yes

Message is coherent

No

Message incoherence

No

Failure

The Figure 3 explains the behaviors of the A.S.R. state plug-in. It shows the linear way the plug-in reacts to the Ivy events and the interpretations made by the two others plug-in. Each of these states (excepting the waiting speech) reacts to a time out which makes them return to the waiting speech state.

4.2.2. Watermarking plug-in

For each clearance/acknowledgement dialog, the watermarking plug-in displays a feedback according to the correspondence between the receiver's callsign of a clearance and the acknowledger's callsign. The receiver's callsign is recognized by the controller A.S.R. agent in the controller speech which sends it on the Ivy bus while the acknowledger's callsign is sent by the pseudo-pilot agent (watermarking of the messages).

This module takes into account the possible vocal recognition errors. In case errors are detected, the match/mismatch feedback is stopped until the next clearance/acknowledgement dialog. Indeed, displaying a feedback knowing that information is erroneous could cause serious misinformation to
the controller while increasing air traffic safety is the main goal of SCOPE project. The inconsistency in the messages sent by the controller A.S.R. agent is displayed as a state of the A.S.R. plug-in.

The watermarking plug-in has twelve possible states; three of them are due to inconsistency in the messages sent by the A.S.R. or appear when the controller A.S.R. agent can not interpret the controller speech. Those states do not have any particular display in the Watermarking plug-in:

- recognition error (no recognition by the A.S.R.)
- message incoherence (recognized callsign does not exist)

The other states depend on the matching between the recognized callsign in the controller speech and the watermarking callsign sent by the pilot. The state of the plug-in changes according to the following parameters:

- the two callsigns are matching
- tracks corresponding to the callsigns are displayed at screen.

When a track involved in a watermarking scenario is not displayed on screen, the radar image displays an attention getter as a feedback. This attention getter permits to inform the controller that he can see the track changing the display area of Twinkle and highlights the direction where the track can be found. The attention getter stays on screen until the track is visible or the end of the callsigns match process.
Figure 4: Watermarking plug-in states and behaviors
4.2.3. Clearance conflict plug-in

The clearance conflict plug-in aims at alerting the controller if there is a mismatch between the orders of a clearance and the acknowledgement of this clearance by the corresponding pilot.

For each clearance, the plug-in displays the orders composing the clearance near the track on the radar image if it is visible or on an attention getter if the plane is not visible. It then waits for the acknowledgement by the pilot. If the foreseen pilot (see watermarking plug-in) acknowledges the clearance, the different parameters of the acknowledgment are compared with the orders of the clearance. If a mismatch occurs, an alert is displayed, permitting the controller to prevent a pilot’s misunderstanding of his orders.

The clearance/acknowledgement comparison is only made if the foreseen pilot acknowledges the clearance. In case the responding pilot is not the target of the clearance, the watermarking plug-in alerts the controller of the error. Adding clearance/acknowledgement conflict alert feedback on the controller interface had not been judged pertinent and could cause a cognitive overload.

In a first time, the displayable states depend on the orders found in the clearance by the controller A.S.R. agent. Then, it is the matching between the parameters acknowledged by the pilot and their corresponding values which is displayed. An alert is shown if parameters and/or values of the acknowledgement do not match with the corresponding clearance.
Figure 5: Clearance/acknowledgement conflicts plug-in behaviors
4.3. Air Traffic Simulator agent

The Air Traffic Simulator used in the SCOPE project, Rejeu, is a software agent created and maintained by the CENA for demonstration purpose. Its main feature is to provide air traffic information needed to visualize radar images on the Ivy software bus. The information sent is based on real air traffic information registered in situ in files respecting the STR and COURAGE formats. Rejeu permits to replay real air traffic situations, to increase and decrease course of time, to place at certain moments in the records. Another function of Rejeu is to take into account changes in some flights information according to messages received via the Ivy bus, those orders can be sent by a pseudo-pilot software agent or a flight simulator connected to Ivy by example.

In the demonstrator, Rejeu will send information to Twinkle, the radar display agent, specialized in radar images visualizations. Indeed, Rejeu and Twinkle are two software agents highly integrated to each other. They are often used together in demonstrators by the CENA and IntuiLab.

5. Communication protocol

This section describes the communication protocol of SCOPE. The protocol is based on simple text messages. Each SCOPE agent receives only messages that it has registered for before (i.e. messages matching a specific template) and ignores any other messages.

5.1. Communication protocol of the Controller ASR agent

5.1.1. State of the ASR agent

The different steps of the analysis of the controller's speech by the A.S.R. cause the sending of the following messages. Those messages are sent on the Ivy bus and are received by the A.S.R. state plug-in of twinkle which waits for those templates messages.

Messages sent by the A.S.R. agent:

- When the beginning of an expression corresponding to the air traffic control grammar is recognized by the A.S.R.: 
  \texttt{controller\ speechAnalysisBeginEvent}

- When the spotted expression's end is detected: 
  \texttt{controller\ speechAnalysisEndEvent}

- When the analysis of the whole spotted expression is finished: 
  \texttt{controller\ speechAnalysisSuccessEvent}
If the A.S.R. failed to analyze the spotted expression:

controller speechAnalysisFailureEvent

5.1.2. Interpretations

The interpretations of the controller speech by the A.S.R. are sent toward the following components connected to the Ivy software bus: watermarking and clearance/acknowledgement conflicts plugins.

The format of these messages is the following:

ccontroller callsign=|…| instruction1=|…| value1=|…| instruction2=|…| value2=|…| responsetime=…

The SCOPE demonstrator is limited to two orders per clearance. This is the reason why two fields "instruction" can be fulfilled in the messages.

The following pairs of instruction/value are possible.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>climb</td>
<td>integer</td>
</tr>
<tr>
<td>descend</td>
<td>integer</td>
</tr>
<tr>
<td>turn-left</td>
<td>integer [0..360]</td>
</tr>
<tr>
<td>turn-right</td>
<td>integer [0..360]</td>
</tr>
<tr>
<td>heading</td>
<td>integer [0..360]</td>
</tr>
<tr>
<td>speed-knots</td>
<td>positive integer [100..390]</td>
</tr>
<tr>
<td>speed-mach</td>
<td>positive floating [0.70..0.90]</td>
</tr>
</tbody>
</table>

When the interpretation can be ambiguous, the Nuance recognition engine returns the N-best possible interpretations (N-bests) to the A.S.R. component. Those N-bests can be expressed in the Ivy message separating the possible instructions or values with the character `|`.

For example, if the A.S.R. is hesitating between 240 and 214, it will send |240|214|.

The "responsetime" field is an integer representing the A.S.R. response time expressed in milliseconds.

5.1.3. Examples of speech interpretations

51.3.1 Only one order in the clearance:

Controller: "AFR440, good morning climb level 280"
(The A.S.R. hesitate on the interpretation, Nuance returned N-bests.)
The following messages are sent on the *Ivy* bus:

controller speechAnalysisBeginEvent

ccontroller speechAnalysisEndEvent

ccontroller speechAnalysisSuccessEvent

controller callsign=&AFR440 instruction1=climb value1=[280|270] instruction2= value2= responsetime=37

[NOTE: When there is only one order in the clearance, the instructions2 and value2 fields are empty]

51.3.2 Two orders:

Controller: "AFR440 climb level 280 and turn heading 350"
(The *A.S.R.* hesitate on the interpretation, Nuance returned N-bests.)

The following messages are sent on the *Ivy* bus:

ccontroller speechAnalysisBeginEvent

ccontroller speechAnalysisEndEvent

ccontroller speechAnalysisSuccessEvent

controller callsign=&AFR440 instruction1=heading value1=[350] instruction2=level value2=[350] responsetime=29

51.3.3 Recognition failure:

An interpretation is launched on an expression but the *A.S.R.* can't interpret the speech with the provided grammar.

The following messages are sent on the *Ivy* bus:

ccontroller speechAnalysisBeginEvent

ccontroller speechAnalysisEndEvent

ccontroller speechAnalysisFailureEvent

5.2. Communication protocol of the Pilot ASR agent
5.2.1. State of the ASR agent

The different steps of the analysis of the pilot's speech by the *A.S.R.* cause the sending of the following messages. Those messages are sent on the *Ivy* bus and are received by the *A.S.R.* state plug-in of *twinkle* which waits for those templates messages.

Messages sent by the *A.S.R.* agent:
- When the beginning of an expression corresponding to the air traffic control grammar is recognized by the *A.S.R.*:
  \[\text{pilot speechAnalysisBeginEvent}\]
- When the spotted expression's end is detected:
  \[\text{pilot speechAnalysisEndEvent}\]
- When the analysis of the whole spotted expression is finished:
  \[\text{pilot speechAnalysisSuccessEvent}\]
- If the *A.S.R.* failed to analyze the spotted expression:
  \[\text{pilot speechAnalysisFailureEvent}\]

5.2.2. Interpretations

The interpretations of the pilot speech by the *A.S.R.* are sent toward the following components connected to the *Ivy* software bus: watermarking and clearance/acknowledgement conflicts plug-ins. Furthermore, those messages permit to send the flight changes information which composes the acknowledgement to the ATC simulator.

The format of these messages is the following:

\[\text{pilot callsign=}[...]\text{ instruction1=}[...]\text{ value1=}[...]\text{ instruction2=}[...]\text{ value2=}[...]\text{ responsetime=}[...]\]

The SCOPE demonstrator is limited to two actions per acknowledgement. This is the reason why two fields "instruction" can be fulfilled in the messages.

The following pairs of instruction/value are possible.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>climb</td>
<td>integer</td>
</tr>
<tr>
<td>descend</td>
<td>integer</td>
</tr>
<tr>
<td>turn-left</td>
<td>integer [0..360]</td>
</tr>
<tr>
<td>turn-right</td>
<td>integer [0..360]</td>
</tr>
<tr>
<td>heading</td>
<td>integer [0..360]</td>
</tr>
<tr>
<td>speed-knots</td>
<td>positive integer [100..390]</td>
</tr>
<tr>
<td>speed-mach</td>
<td>positive floating [0.70..0.90]</td>
</tr>
</tbody>
</table>


This document is published by EUROCONTROL in the interests of the exchange of information. It may be copied in whole or in part, providing that this copyright notice and disclaimer are included. The information contained in this document may not be modified without prior written permission from EUROCONTROL.

EUROCONTROL makes no warranty, either implied or express, for the information contained in this document, neither does it assume any legal liability or responsibility for the accuracy, completeness or usefulness of this information.
When the interpretation can be ambiguous, the Nuance recognition engine returns the N-best possible interpretations (N-bests) to the A.S.R. component. Those N-bests can be expressed in the Ivy message separating the possible instructions or values with the character ‘|’.

For example, if the A.S.R. is hesitating between 240 and 214, it will send |240|214|.

The "responsetime" field is an integer representing the A.S.R. response time expressed in milliseconds.

5.2.3. Examples of speech interpretations

52.3.1 Only one action in the acknowledgement:

Pilot: "AFR440, climbing level 280"
(The A.S.R. hesitate on the interpretation, Nuance returned N-bests.)

The following messages are sent on the Ivy bus:

pilot speechAnalysisBeginEvent

pilot speechAnalysisEndEvent

pilot speechAnalysisSuccessEvent

pilot callsign=|AFR440| instruction1=|climb| value1=|280|270| instruction2= value2=
responsetime=35

[NOTE: When there is only one action in the acknowledgement, the instructions2 and value2 fields are empty]

52.3.2 Two actions:

Pilot: "AFR440 climbing level 280 and turning heading 350"
(The A.S.R. hesitate on the interpretation, Nuance returned N-bests.)

The following messages are sent on the Ivy bus:

pilot speechAnalysisBeginEvent

pilot speechAnalysisEndEvent

pilot speechAnalysisSuccessEvent
52.3.3 Recognition failure:

An interpretation is launched on an expression but the A.S.R. can't interpret the speech with the provided grammar.

The following messages are sent on the Ivy bus:

- pilot speechAnalysisBeginEvent
- pilot speechAnalysisEndEvent
- pilot speechAnalysisFailureEvent

5.3. Communication protocol of the radar display agent

The communication between the air traffic simulator, Rejeu, and the radar image visualization agent, Twinkle, has already been identified and will not be modified for SCOPE. This report only documents a small set of the full protocol.

The three plug-ins added to Twinkle only receive messages from other agents connected to the Ivy bus.

[NOTE: In the communication protocol between Rejeu and Twinkle, the tracks are designed by a flight number instead of callsigns. The communications toward Rejeu keep this designation format.]

The pilot actions on the en route flight information must be displayed in the image radar visualization tool and taken into account by the air traffic simulator, Rejeu. The following orders are sent to Rejeu when the pilot acknowledges a clearance. Rejeu inserts the new data in its tables and notifies the visualization agent.

5.3.1. Vectoring instructions

Two kinds of vectoring instructions are supported:

- absolute heading:
  
  \[
  \text{AircraftHeading Flight}=77 \text{ To}=180 \\
  \text{Where To is a angle expressed in degrees (between 0 and 360)}
  \]

- relative heading:
  
  \[
  \text{AircraftTurn Flight}=434 \text{ Angle}=-15 \\
  \text{Where Angle is an angle expressed in degrees. If positive, it's a turn on the right else it's a turn on the left}
  \]
5.3.2. Flight level instructions

The following message is sent on the *Ivy* bus:

```
AircraftLevel Flight=76 Fl=330
```

Where *Fl* is the new level expressed in feet.

5.3.3. Speed control instructions

Changes of the speed are expressed by the *Ivy* message:

```
AircraftSpeed Flight=56 Speed=250
```

Where *Speed* is expressed in knots.

5.4. Communication protocol of the Air Traffic Simulator agent

Communication between the air traffic simulator, *Rejeu*, and the radar image visualization agent, *Twinkle*, already exists and are not be modified for SCOPE. The protocol is not documented in this report.

6. Operational environment

This section describes the target hardware and software environment, i.e. the minimal configurations for the use of the deliverables.

6.1. Operational environment of the Controller ASR agent

The ASR engine used in the SCOPE projects has been integrated and tested under Windows 2000.

The minimal hardware configuration consists of the following items:

- Intel Pentium 4 or Athlon XP processor, 2 GHz or above;
- 512 Mb RAM memory;
- Monitor: 1024x768 resolution or higher monitor;
- CD-ROM;
- SoundBlaster compatible sound card;
- 10/100 Mb Network interface card with RJ45 connectivity;
- A good quality microphone.

The software environment consists of the following:

- Windows 2000

The recommended network configuration is the following:

- IP address: 10.0.0.1
6.2. Operational environment of the Pilot ASR agent

The ASR engine used in the SCOPE projects has been integrated and tested under Windows 2000.

The minimal hardware configuration consists of the following items:
- Intel Pentium 4 or Athlon XP processor, 2 GHz or above;
- 512 Mb RAM memory;
- Monitor: 1024x768 resolution or higher monitor;
- CD-ROM;
- SoundBlaster compatible sound card;
- 10/100 Mb Network interface card with RJ45 connectivity;
- A good quality microphone.

The software environment consists of the following:
- Windows 2000

The recommended network configuration is the following:
- IP address: 10.0.0.2
- Subnet Mask: 255.255.255.0

6.3. Operational environment of the radar display and ATC simulator agents

The radar display (Twinkle) and the ATC simulator (Rejeu) have been tested under Linux Mandrake 9.2.

The minimal hardware configuration consists of the following items
- Intel Pentium 4 or Athlon XP processor, 2 GHz and above
- 256 Mb RAM memory
- Monitor:
- A NVIDIA graphic card: GeForce 4 (not MX) and above
- CD-ROM
- 10/100 Mb Network interface card with RJ45 connectivity

The software environment consists of the followings:
- Mandrake 9.2
- FVWM Window manager
- perl (5.8 or above) and perl/Tk (800 or above)
The recommended network configuration is the following:
- IP address: 10.0.0.3
- Subnet Mask: 255.255.255.0

[NOTE: We suggest using the same computer for the radar display HMI and the Air Traffic Simulator because the Air Traffic Simulator doesn’t have a GUI]

6.4. Other requirements

In order to make SCOPE agents work together, the followings items are needed:
- a 4 ports RJ45 Hub
- 3 RJ45 cables to connect computers to the network Hub

7. References

Work packages of the SCOPE project:
- WP1-R1a - ASR software evaluation, Thibaut Ehrette et Olivier Grisvard, Thales R&T
- WP1-R1b - ASR software recommendations, Thibaut Ehrette et Olivier Grisvard, Thales R&T

Software components used:
- Ivy software bus: www.tls.cena.fr/products/ivy/

8. Glossary

A.S.R.: Automatic speech recognizer
Ivy: Software bus developed by the CENA
Rejeu: Air traffic simulator developed by the CENA (www.cena.fr)
Twinkle: Radar image visualization component developed by the CENA (www.cena.fr)