EUROPEAN ORGANISATION
FOR THE SAFETY OF AIR NAVIGATION

EUROCONTROL

This document is issued as an EATM Guideline. The contents are not mandatory. They provide information and explanations, or may indicate best practice.

CoRe Project

Baseline Exemplary Style Guide

Edition Number	:	1.0
Edition Date	:	16.08.2004
Status	:	Released Issue
Intended for	:	EATM Stakeholders

EUROPEAN AIR TRAFFIC MANAGEMENT

DOCUMENT CHARACTERISTICS

TITLE		
CoRe Project Baseline Exemplary Style Guide		
	EATM Infocentre Reference:	040201-10
Document Identifier	**Edition Number:**	1.0
HRS/HSP-006-GUI-01	**Edition Date:**	16.08.2004

Abstract

This document has been developed as part of the CoRe Project conducted within the Human Factors Sub-Programme (HSP) of the EATM* Human Resources Programme (HRS) by the EUROCONTROL Human Factors Management Business Division (DAS/HUM)** to improve the process and outcomes of the activity of defining and developing effective Controller Working Positions (CWPs) for Air Traffic Management (ATM). Work was carried out within the ATM Systems and Concepts Business Unit at the EUROCONTROL Experimental Centre (EEC), Brétigny, France.

Specifically this document addresses the nature and value of 'an Interaction Style Guide (ISG)' as a key element in the development and implementation of the direct manipulation graphical interfaces, which are starting to be introduced into operational centres. It identifies the structure and contents appropriate to such a guide, and demonstrates these with the guidance material corresponding to the 'Example' Human-Machine Interface produced to test and illustrate the CoRe development processes. The document contains a large number of hyperlinks and is best used in an electronic form.

* In 1999 the 'European Air Traffic Control Harmonisation and Integration Programme (EATCHIP)' was renamed 'European Air Traffic Management Programme (EATMP)'. Since May 2003 it is known simply as 'European Air Traffic Management (EATM)'.

** Formerly known as the 'Human Factors and Manpower Unit (DIS/HUM)'

Keywords

Interaction Style Guides	Human-Machine Interaction/Interface	Controller Working Position (CWP)	Visual presentation guidelines
Fonts and text presentation	Use of colour	Design principles	Consistency
HMI requirements	Input management	Usability	Re-usability

Contact Person	Tel	Division
Alistair JACKSON, CoRe Project Leader	+33 1 69 88 75 44	Safety Analysis and Scientific Centre of Expertise (CoE SAS), EEC Brétigny, France

Authors

Alistair Jackson, Alexandra Dorbes (STERIA) and Michel Geissel

STATUS, AUDIENCE AND ACCESSIBILITY					
Status		**Intended for**		**Accessible via**	
Working Draft	☐	General Public	☐	Intranet	☐
Draft	☐	EATM Stakeholders	☑	Extranet	☐
Proposed Issue	☐	Restricted Audience	☐	Internet (www.eurocontrol.int)	☑
Released Issue	☑	*Printed & electronic copies of the document can be obtained from the EATM Infocentre (see page iii)*			

ELECTRONIC SOURCE		
Path:	G:\Deliverables\HUM Deliverable pdf Library\	
Host System	**Software**	**Size**
Windows_NT	Microsoft Word 8.0b	

EATM Infocentre
EUROCONTROL Headquarters
96 Rue de la Fusée
B-1130 BRUSSELS

Tel: +32 (0)2 729 51 51
Fax: +32 (0)2 729 99 84
E-mail: eatm.infocentre@eurocontrol.int

Open on 08:00 - 15:00 UTC from Monday to Thursday, incl.

DOCUMENT APPROVAL

The following table identifies all management authorities who have successively approved the present issue of this document.

AUTHORITY	NAME AND SIGNATURE	DATE
CoRe Project Leader	Alistair Jackson A. JACKSON	15.7.04
Chairman HRT Human Factors Focus Group (HFFG)	V.S.M. WOLDRING	15.07.04
Manager EATM Human Resources Programme (HRS-PM)	M. BARBARINO	16.7.04
Chairman EATM Human Resources Team / Programme Steering Group (HRT/PSG)	A. SKONIEZKI	19.7.04
Director ATM Programmes (DAP)	G. PAULSON	27.7.04

DOCUMENT CHANGE RECORD

The following table records the complete history of the successive editions of the present document.

EDITION NUMBER	EDITION DATE	INFOCENTRE REFERENCE	REASON FOR CHANGE	PAGES AFFECTED
0.1	15.12.2002		Working Draft (first full version for review and comment)	All
0.2	20.12.2002		Draft 1 (restructuring by AD, reorganisation of bookmarks)	All
0.3	31.12.2002		Draft 2 (incorporating comments from CGP Review and including Part 3)	All
0.4	18.03.2003		Draft 3 for HFFG (small changes)	p49
0.4a	03.04.2003		Draft 4 (small corrections, intermediate version awaiting HFFG feedback)	§5.6, §6.8
0.5	02.01.2004		Proposed Issue for HRS-PSG Meeting in January 2004 (document configuration and editorial changes)	All
1.0	16.08.2004	040201-10	Released Issue (agreed on 28-29.01.2004) (final document configuration and editorial changes)	All

CONTENTS

Page intentionally left blank

EXECUTIVE SUMMARY

This document addresses the role of Interaction Style Guides in support of the development of Graphical Human-Machine Interfaces for Air Traffic Management applications.

It discusses the potential usefulness of such guides. It makes suggestions as to what they should contain and how they could be structured. It then illustrates these suggestions through the example of a style guide developed for the baseline, en-route air traffic controller working position used to demonstrate the development framework produced by the Core Requirements for ATM Working Positions Project (CoRe: HRS/HSP-006).

The document is divided into six parts:

- Part 1 consists of two sections. Section 1 is the introduction. Section 2 discusses and explains the role of a style guide within a requirements-based development framework such as that proposed by the CoRe Project.
- Part 2 explains the importance of design philosophy and Section 3 deals with the expression of the design approach, automation philosophy and user characteristics.
- Part 3 addresses general design principles and ‘transversal’ human-machine interaction requirements which are potentially applicable across all the functionality of a particular operational interface. Following an introduction in Section 4:
    - Section 5 deals with principles of interaction,
    - Section 6 deals with principles of input management,
    - Section 7 deals with principles of visual presentation.
- Part 4 considers basic interface elements and illustrates these in Section 8 with the specification employed for the basic display and management of the CoRe Baseline HMI.
- Part 5 provides a bibliography and reference list; explains abbreviations and acronyms, and acknowledges the many contributors to this deliverable.
- Part 6 consists of technical annexes to the document.

In order to better realise the ‘exemplary function’, Parts 2 and 3 are presented in the form of two parallel tracks: the first track provides explanation and guidance material, while the second illustrates how these might be practically realised with the CoRe-based example.

Page intentionally left blank

PART 1: INTRODUCTION

Page intentionally left blank

1. DOCUMENT OVERVIEW

1.1 Background

This document is one part of a body of material developed by the CoRe Project to improve the process and the outcomes of the activity of defining and developing effective Controller Working Positions (CWPs) for ATM.

The CoRe Project was conducted as part of the Human Factors Sub-Programme (HSP) of the EATM[1] Human Resources Programme (HRS) by the EUROCONTROL Human Factors and Manpower Unit (DIS/HUM), today known as the Human Factors Management Business Division (DAS/HUM) [20]. It was a three-year project completed at the end of 2002, and the work was carried out within the ATM Systems and Concepts Business Unit at the EUROCONTROL Experimental Centre (EEC), Brétigny, France.

The approach taken by CoRe is described in detail in [21], [5] and [6]. In summary, rather than taking a formal system development approach, CoRe begins by trying to understand current processes and practices and to improve them. Methodologically, CoRe views the development of working positions for ATM applications as a socio-technical process, one involving a variety of human actors and technology with a collective objective. Within this perspective, it employs a bottom-up approach to the understanding of the problems involved in integrating a variety of expertise to arrive at a successful outcome ([21] §3). Many of these problems are directly related to communication difficulties between actors and between processes. A fundamental objective of CoRe has been to improve communication, not only within development activities but also between them.

The project has also identified a need for a more structured approach to evaluation and testing. Related to this is the need for a shift to a more requirement-oriented approach, not only for evaluation but also to better support harmonisation and re-use through transfer from one project to another.

To realise both process and 'cultural' improvements the project exploits software engineering technology (the Unified Modelling Language [UML]), case tools, etc.) and methods to develop a requirement-oriented process and a specially designed support framework. The framework helps to reconcile the functional and component views appropriate to different stakeholders, and to different stages and activities in development. It also supports consistency, traceability and document management.

[1] In 1999 the 'European Air Traffic Control Harmonisation and Integration Programme (EATCHIP)' was renamed 'European Air Traffic Management Programme (EATMP)'. Since May 2003 it is known simply as 'European Air Traffic Management (EATM)'.

To illustrate the approach and the improved process, CoRe has three main areas of deliverable as shown in Figure 1:

- a qualitative, organisational model of the activities involved in developing adequate CWPs, supported by additional explanatory documentation;
- the UML/XML[2]-based supportive development framework;
- a baseline CWP - a worked example of a high-quality (but basic functionality) en-route CWP, completely described using the framework (CoRe Baseline Human-Machine Interface [HMI]).

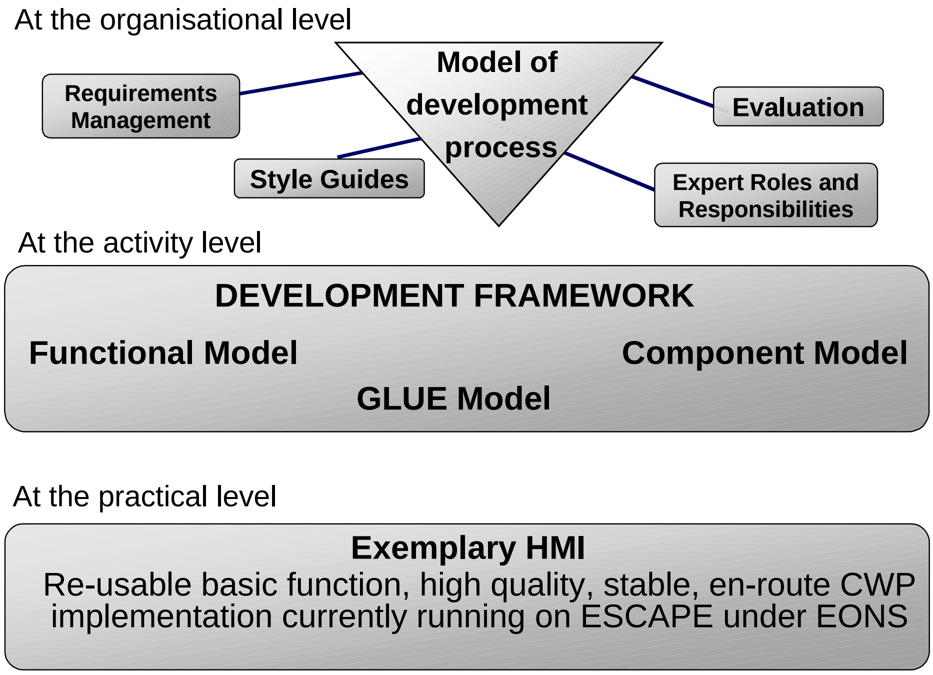


Figure 1: CoRe Deliverables

The additional documentation referred to in the first bullet point includes a general introduction to requirements capture and management [3], recommendations on HMI evaluation techniques [4], a general review on the subject of style guides for ATM applications [2], and an overview of the CoRe Project [21].

In this context, the current document forms a bridge between the first and third deliverables. As detailed below, it serves a double purpose: it explains the role and function of a style guide for a graphical user interface and, in parallel, illustrates both of these using the example of the CoRe Baseline CWP.

[2] Extensible Markup Language

1.2 Scope and Purpose

In more detail this document addresses the nature and value of 'an Interaction Style Guide' (ISG) for the development and implementation of the direct manipulation graphical interfaces, which are starting to be introduced into operational centres.

Drawing on work already commissioned by CoRe [2], it:

- explains the CoRe perspective on the function of a style guide;
- describes, in detail, the different areas that a style guide should cover; and
- provides examples for each of these areas as applied to a baseline en route HMI.

While the example is clearly focussed on ATC issues and interfaces, the basic process of defining and employing a style guide should be applicable to most interfaces based on graphical interaction.

By the end of the document the reader should have:

- a clear understanding of the importance of the style guide in ensuring the consistency and coherency of the interface;
- a basic comprehension of how to define the main elements of such a guide; and
- an example that can serve as a starting point for the production of style guides for similar systems.

1.3 Structure

Following this overview of the document, which makes up Section 1, Section 2 discusses the role of a style guide and puts it into the requirement-oriented context which characterises the CoRe Project. These two introductory sections form the first part of the document. The second part deals with the design philosophy. The third part discusses global issues and principles that apply to all aspects of the interface. The fourth part, extracted from the functional model of the HMI, describes the individual elements of the interface infrastructure. The fifth part contains References, Abbreviations and Acronyms, and a list of those who contributed to this document, and finally the sixth part contains the technical annexes to the document.

In Part 2, 'Design Philosophy',

- Section 3 deals with the expression of the design approach, automation philosophy and user characteristics.

In Part 3, ‘General Design Principles and Transversal Requirements’,

- Section 4 provides an ‘Introduction’,
- Section 5 describes the ‘Principles of Interaction’,
- Section 6 addresses the ‘Principles for Input Management’, while
- Section 7 is about the ‘Principles for Visual Presentation’.

In Part 4, ‘Basic Interface Elements’,

- Section 8 covers the ‘Display and Management of the Baseline HMI’.

In Part 5, ‘Bibliography, Terminology and Contributors’,

- details are provided on authors referred to in the document,
- Abbreviations and Acronyms used, and their full designations, are listed, and
- Contributors to this guidance deliverable are thanked.

Finally, Part 6 consists of five ‘Technical Annexes’ to the document.

Parts 2 and 3 are each presented in the form of two parallel tracks, one track being guidance material, the other providing an example.

The ‘guidance track’ can be identified by the use of brown text, in a Verdana font. The guidance material describes the purpose, scope and principles of each sub-section of a style guide suitable for CWP HMI.

The ‘example’ illustrates these by applying the principles to the CoRe Baseline HMI. By the end of the document, this exemplary track constitutes a complete style guide to the CoRe Baseline. The exemplary material is presented in a Verdana font on a textured background as in this paragraph.

Within the example text there will be elements of boxed text, containing ‘bookmarked’ statements in the form of assumptions, guidelines or requirements.

IDENTIFIER (this example is a transversal HF guideline)

SUBJECT

TYPE assumption, guideline or requirement

[X.HF.G1] (Support for generalisation/guideline): Similar user action applied to similar objects in similar contexts should result in similar behaviour.

CONTENT

1.4 A Word on References and Resources

Throughout this document, the reader will find conventional reference citations supporting the text. If the document is being viewed in an electronic form these citations will link to the main 'References' Section of the document. However there are many Web sites and resources available on the Internet relating to the subject of 'Interface Style Guides' and the related subjects of 'Interface Design' and 'Usability'. Where possible, links to relevant Web sites are also included in the 'References' Section.

Additionally, at the end of several sections of the document, the reader will find a short bibliography of background material relevant to those sections.

Page intentionally left blank

2. INTERACTION STYLE GUIDES FOR ATM: AN OVERVIEW

2.1 Definition

A more extended discussion of several of the issues addressed below can be found in [2], which forms part of the background documentation furnished with the CoRe Organisational Model.

In the CoRe Project we use the term ‘Interaction Style Guide (ISG)’ rather than ‘style guide’ to avoid any confusion with the ‘style guides’ developed by some large companies to help manage the ‘company image’.

The use of the term ‘style guide’ in relation to human computer interaction grew with the development of Graphical User Interfaces (GUIs). These interfaces, which first achieved maturity with the XEROX STAR and ALTOS [56], came to wider public attention with the advent of the Apple Macintosh. They revolutionised interaction with computers, replacing the typing of ‘commands’ by the ‘direct manipulation’ of objects and tokens, presented graphically on the computer screen, by means of a pointing device. These WIMP (Window, Icon, Menu, Pointer) interfaces allowed users to interact using a simplified subset of the same ‘hand-eye’ skills they employed to interact with objects in the real world.

However, these interfaces had a cost. They were more demanding in terms of code and were more complex in design. Their conception implied an understanding of basic human performance characteristics and graphical design as well as software development[3]. In particular, early production machines were faced with the issue of how to communicate the necessary requirements and principles to third-party developers who had not been involved with the development of the system. The first published ISG was produced by Bruce Tognazzini [48] as part of the launch of the original Apple Macintosh (Classic). It described the principles of interaction, the use of fonts, layout, sizing, spacing, menu design, etc., explaining to designers how to create an application consistent with those applications already provided by Apple[4]. Today, with the increasing complexity of operating systems and interface software, many companies and organisations, e.g. Open Software Foundation (OSF) Motif, IBM, Microsoft, etc., make use of ISGs as a mixture of standards and guidelines for developers. In some cases, e.g. Apple, these are intended to reinforce a particular ‘look’ (visual style) and ‘feel’ (interactive

[3] The fact that they also made computers accessible to non-computer specialists was one of the main drivers in the development of User-centred System Design (UCSD), which is difficult to separate from the emergence of the GUI.

[4] In the case of the early Macintosh computers, this was strongly reinforced by the necessity of using the ‘Apple Toolbox’ software library to build GUIs within the Macintosh Operating System (OS) which was not openly published.

style); in others (e.g. OSF Motif [7]) they provide developers with guidance in developing their own consistent look and feel.

2.2 Value for ATM

In discussing the nature of ISGs in the context of CoRe, Cooke and Marti [2] make a distinction between:

- ‘commercial guides’, such as those described in the preceding section, which are intended for third-party developers and are generally open to the public, and
- ‘corporate guides’ that are intended for use within a single organisation and often remain ‘private’.

The function of these corporate guides is somewhat different in that they are designed to *“ensure and enhance internal software consistency where the primary concern of the organisations involved is not the development of the system itself but the use of that system as a tool in providing another service, such as ATM”. [2]*.

There are already examples of such style guides employed in the development of individual ATM systems, e.g. for Sweden 2000 and for ODS France. These may form part of the requirements documentation provided to industrial suppliers but they are also used internally to perform at least three main roles:

- to ensure that interface and system behaviour are internally consistent;
- to ensure that any additions or modifications subsequently made to the software also remain coherent and consistent;
- to consolidate and share design decisions made in the course of the project, serving as a communication aid to different stakeholders and as a shared document to help build the sense of community within a project[5].

One additional and potentially very important role for ATM is that ISGs, and the evaluation of a system’s conformity with them, have the potential to become part of processes employed in the qualification by regulatory authorities of new ATM systems or changes to ATM systems.

[5] Several authors (e.g. [12], [2]) emphasise the importance of beginning the development of the style guide at an early stage and employing it as a focal activity for an integrated, multi-disciplinary design team.

2.3 Background

In fact, as Cooke and Marti indicate ([2], §2), the requirements of the CoRe Project contain elements of both the ‘commercial’ and ‘corporate’ models.

Corporate in the sense that, within an organisation developing an ATM system employing CoRe, succeeding generations of developers will need to be able to modify and extend elements in a consistent and coherent way. *Commercial*, in the sense that other organisations (third parties) may be contracted to develop systems to meet requirements expressed in an ISG or to add new functionalities to a system already defined by an ISG. Their analysis runs as follows:

> *The intention behind CoRe is the development of re-usable requirements, including those governing style, that can capitalise on advances made to date and act as a bridge between previous and future advances. The style guide therefore plays an important mediating role in various contexts. The style guide must be able to effectively communicate:*
>
> 1. *The style philosophy, which provides the general conceptual direction from which style details emerge.*
>
> 2. *The general human requirements arising from this philosophy and which will remain constant across the design and implementation.*
>
> *In so doing the style guide must, where appropriate, take account of local definition and tailoring of style in particular implementation contexts whilst providing sufficient information (based on 1 and 2) to ensure that these local style definitions are consistent (functionally) with the original HMI concept.*
>
> (Cooke & Marti, 2002)

In fact the requirement for CoRe includes one further element. The HMI and associated ISG provided in CoRe are both intended to serve as examples (albeit re-usable ones) for CWP development teams. CoRe emphasises the re-use of requirements above solutions. It is perfectly reasonable to assume that there may be several different HMI solutions that meet similar requirements (perhaps with different trade-off priorities). The owners of these different solutions would probably need to produce ISGs to support their development. CoRe has the additional objective of helping them to develop an ISG appropriate to the needs of their stakeholders and their development. Thus we can add a further communication objective to the two provided above.

> 3. *Provide guidance material which allows users of the CoRe approach to produce suitable ISGs, or adapt existing ISGs, during the course of their development.*

The example provided for the CoRe Baseline is intended to meet objectives 1 and 2. Reference [2] and the introductory and guidance material in this document are intended to meet objective 3.

2.4 Development

In order to achieve these objectives then, an ISG as used within the framework of CoRe is considered as providing at least four kinds of information:

1. The design philosophy.
2. The general design principles.
3. Transversal Human Factors (HF) and HMI requirements[6]
4. A specification of the basic interaction infrastructure.

Each of these is explained briefly below before being expanded upon in the following sections.

2.4.1 Design Philosophy

The design philosophy explains the general approach to the development of the system and the main policies and assumptions which underlie it.

Of particular interest for the ATM community, it may address the ‘automation philosophy’ which gives rise to the allocation of roles and responsibilities between human and hardware or software components.

The commonest design philosophy currently employed for interactive systems is generally termed ‘human-centred’, or ‘user-centred’ design and is addressed by an ISO standard [13]. The CoRe example in §3 onwards, adopts a human-centred approach.

The design philosophy impacts on the organisation of the process of designing, realising and introducing an HMI into service.

2.4.2 General Design Principles

These are general design principles, high-level human-machine interaction principles, characteristics of operator roles, which arise from the application of the design philosophy. They are often expressed as design objectives, which are desirable at all levels and in all aspects of the interface. Commonly cited examples are *consistency, flexibility, tolerance, robustness.*

Design principles provide the design teams with the basis for making choices about the types of strategies and mechanisms which are appropriate to the

[6] For convenience CoRe makes a distinction between ‘Functional Requirements’ which relate directly to the function and objectives of the socio-technical system and Human Factors (HF) requirements which arise in order to optimise the contribution of the human participants in the system.

design. If the design principles are clearly expressed, understanding of the reasoning behind design choices can be shared not only by the team, but also by other designers who may have to become involved with the maintenance or modification of the interface at a later date. Note, however, that design principles may come into conflict and have to be prioritised.

Like design philosophy, the establishment of design principles has to be based on a sound knowledge of human characteristics, perception and performance.

2.4.3 Transversal HF Requirements

Transversal HF requirements (XHF) are requirements or constraints that potentially apply to all aspects of the interface. They will generally be derived directly from the design principles. For example the ‘consistency’ principle will lead to the choice of a particular selection model or logic for the use of colour coding throughout the interface. XHF relate to the general approach that the interface has adopted to interaction and are thus potentially applicable to any new function that is being added. They can be contrasted with HF requirements that derive directly from the functionality being introduced. If, for example, a Short-term Conflict Alert (STCA) was being integrated, there could be an HF Requirement to *‘provide a visual alert x minutes before loss of separation’*. This requirement derives directly from the functionality of the STCA. However the detailed choice of presentation should take into account the general approach to alerts defined transversally across the whole HMI, so that the presentation mechanism eventually chosen will be coherent with other functions and have an appropriate priority in relation to any other alerts deriving from other functions.

In the CoRe approach it is intended that HF requirements which derive directly from functions should be noted and described in the specifications/functional model along with the function itself. On the other hand, for reasons of economy, XHF should be expressed in the ISG and cross-referenced in the functional model wherever they are applicable.

These transversal requirements are often integrated and expressed in terms of **policies** or **conventions**, e.g. the policy for use of colour, for management of errors, for highlighting, for selection, etc.

2.4.4 Specification of the Basic Interaction Infrastructure

Strictly, an ISG is NOT a specification. Nevertheless, in most graphical interfaces there are classes of objects and sets of mechanisms that provide the basic infrastructure within which sets of applications or functions operate. We have already referred to the main classes of entities in the most common class of graphical interfaces, namely, windows, icons, menus and pointers. There are other general presentation objects (e.g. error messages) and input mechanisms (e.g. sliders and other tools). All of these have visual conventions associated with their presentation and characteristic behaviours. In order to implement them consistently, it makes sense to describe these centrally.

It may be convenient to describe them in the ISG, or it may be that they are expressed directly in the Functional/Specification Model, or both.

2.5 Structure and Content

ISGs vary considerably in terms of the level of detail of their contents and the way in which they are structured. Commercial guides, such as those for the Apple Mac [9] and IBM [10], can be very large (when printed, the current Mac guide runs to almost 400 pages) and very complete especially in terms of the provision of underlying philosophy, and explanatory material on relevant aspects of human performance and behaviour. These guides have been carefully designed to take advantage of both Web and printed formats and to allow navigation by a wide variety of potential users.

Those ISGs which currently exist for operational HMI's are generally much shorter (in the range of twelve-forty pages) and concentrate on design principles and the expression of HMI requirements. Typical topics are:

- input model,
- default rules,
- menus,
- icons,
- window management,
- error handling,
- navigation,
- response times,
- fonts and colours.

Their structure is more organic, reflecting the fact that they are living documents, evolving and expanding to keep a record of issues, requirements and principles identified as the HMI development process proceeds.

Given the emphasis within CoRe on the re-use of information and the importance of information on design rationale, it is to be expected that a CoRe type ISG will be somewhat more detailed than the current operational guides. Like them, however, it should be a dynamic document, reflecting the development and evolution of the project in the hands of the design team. One way to reduce the detail is by the identification of external referents or standards to provide context or default choices for the interface. Given the high quality of the commercial guides and their availability on the Web, they can be a good and widely accessible source of such reference material.

The example provided in the following sections is structured as follows:

- PART 2 addressing the 'Design Philosophy' as described in §2.4.1.
- PART 3 addressing both 'General Design Principles and Transversal HF Requirements' as described in §2.4.2 & §2.4.3. This part is structured in terms of interface functionality (rather than being layered into two sections) in order to make clearer the derivation of requirements from principles. The main three areas of functionality are:
  - the principles of interaction,
  - the principles for input management,
  - the principles for visual presentation.
- PART 4 defining the 'Basic Interface Elements'.

2.6 Bibliography

For the reader who wishes to consult background material, the following papers are recommended:

CENA PII Division Projects http://www.tls.cena.fr/divisions/PII/.

Cooke, M. and Marti, P. (2002). *CoRe Project Document: Style Guides for ATM Development,* also available as EEC Note..

Gale, S. (1996). A Collaborative approach to Developing Style Guides in: Proceedings of ACM CHI Conference on Human Factors in Computing Systems, Vancouver, Canada, April 13-18 (pp. 362-367) ACM Press.

Wilson, C.E. (2001). Guidance on Style Guides: Lessons Learned, Usability *Interface, Vol.* 7, No 4..
Available at: http://www.stcsig.org/usability/newsletter/0104-style.html.

For examples of style guide material, see the:

Macintosh Human Interface Guidelines (2002)
http://developer.apple.com/techpubs/mac/HIGuidelines/HIGuidelines-2.html.

Descendent of the original, this is perhaps the most complete example but also provides both background material and excellent reading lists on related topics. Highly recommended and beautifully presented.

Open Software Foundation (1991). OSF/Motif Style Guide, Release 1.1, Prentice-Hall: Open Software Foundation.
A more up-to-date electronic version of the style guide can be accessed at http://w3.pppl.gov/misc/motif/MotifStyleGuide/en_US/TOC.html.

This is also a classic style guide for an environment within which many ATM systems are currently developed.

Page intentionally left blank

PART 2: DESIGN PHILOSOPHY

Page intentionally left blank

3. WHAT IS ADDRESSED WITHIN A DESIGN PHILOSOPHY?

3.1 The Chosen Approach to Organising the Development Process

The design philosophy impacts on the organisation of the process of designing realising and introducing an HMI into service. The general approach to the design of highly interactive systems today is called User-centred System Design (UCSD) [15]. UCSD has been formally recognised and practised in some parts of the ATM community since the late eighties and in the cockpit only slightly later [14]. User-centred design processes are justified on the basis of their ability to produce more usable systems, which better meet both user and organisational needs. The basic framework for user-centred design processes is concisely and clearly described in ISO 13407 [13]. Figure 2 below is drawn from that standard.

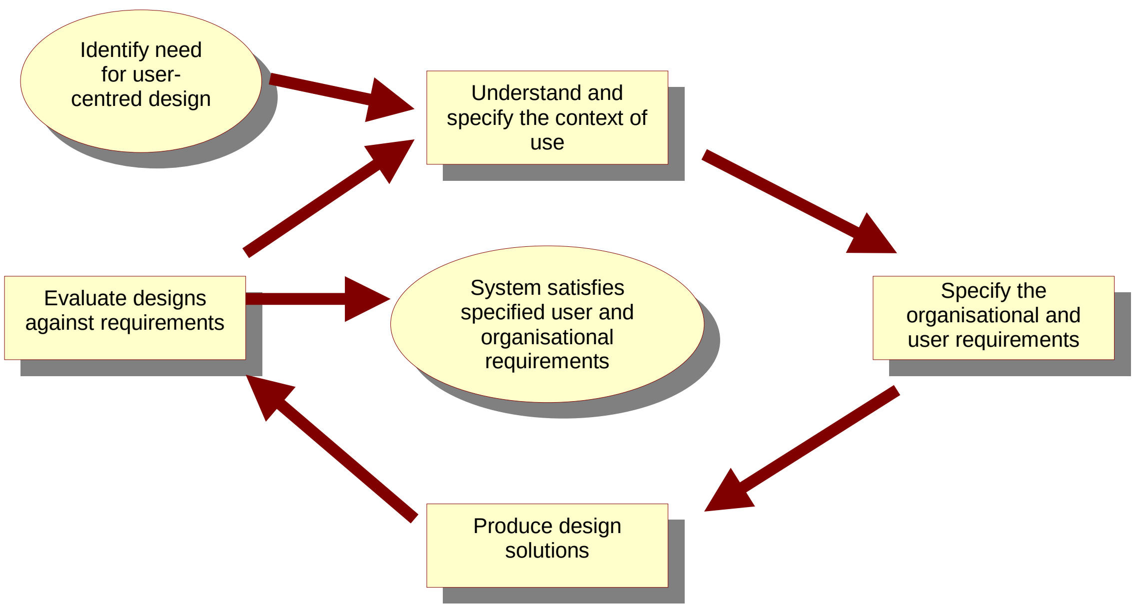


Figure 2: The interdependence of human-centred design activities (after ISO 13407)

EXAMPLE: THE CORE BASELINE

Introductory note

The HMI which is used for the CoRe Baseline has a history. It is an improved version of an HMI derived from ODID IV [18], simplified to form the baseline for the PHARE PD1 study [25]. The HMI was then documented in detail and made available as the EEC REFGHMI in 1995 [24]. The REFGHMI was the first EUROCONTROL HMI Specification which attempted to include information on Style, visual presentation and design rationale.

That documentation became the basis for the initial EATCHIP III baseline [17] and in that form was a major input to a number of other HMI's, most notably the Denmark Sweden Interface (DSI) [42] and the Italian Interface project (ITI) [43].

Much of the text in the following sections has been reproduced or updated from these original documents in order to preserve a record of the original design rationale.

Organisation of the development process

Within CoRe, the basic approach is one of User-centred Design as described in the standard references [13], [15] and [27]. However the approach is applied within a more general framework developed by CoRe, which considers the organisation of CWP development within the ATM community [21].

The baseline is designed to support different levels of study, with the details of the organisation of the process depending on the level of study selected.

The process followed when defining many of the requirements expressed in the section dealing with the basic window elements has some particularities. Although an iterative development was followed, and these elements have been refined over the years, requirements were not always expressed formally. Most of the requirements expressed in that section have been 'reverse engineered' by the CoRe Project.

3.2 Target and Scope of the Development Processes

This section should also define the scope of the interface being developed and the general nature of the type of solution required. For example: Are there pre-existing assumptions and constraints? Is there a reason why this must be a graphical user interface? Or a menu driven system? Are roles and responsibilities already largely defined or is there freedom to define them? Are there organisational constraints that may limit the effectiveness of a User-centred System Design approach?

For example, in ATM we do not generally need to define criteria for managing several applications at once. In most cases we are dealing with a single application, but one which may be extended by the addition of supplementary functions at a later date.

Any standards or regulations that should be applied to the development process or its product should also be identified here.

EXAMPLE: THE CORE BASELINE

Domain of application

The baseline HMI itself has been developed to provide common support for activities at different levels:

- early R&D concept studies (small scale),
- integrative studies and studies directed towards prototyping an operational HMI,
- collaborative studies with different partners.

The precise organisation of the process will depend on the level targeted.

Scope of the HMI

The scope of the interface specification is the single en-route controller working position. However, for operational analysis, the sector is the logical unit. Flexibility is assumed in the different working positions, so the same basic HMI can be re-parameterised to support different controller roles, e.g. Planning Controller/ Organique, Tactical Controller/Radariste.

The initial *a priori* objective was a graphical user interface with a pointing device and (if possible while maintaining good usability) no requirement for conventional alpha-numeric keyboard for the primary functions.[7]

Relevant standards

A number of standards were applicable to the initial REFGHMI and remain current for the CoRe Baseline. These are:

EATCHIP (1993). ODID (Operational Display and Input Development) Report and Guidelines. EUROCONTROL.

[7] Unlike in the US most operational staff do not have basic touch-typing skill and at the time there was considerable resistance to the idea of a keyboard amongst the operational population of certain States.

Reynolds, L. and Metcalfe, C. (1992). The Interim NATS Standard for the Use of Colour on Air Traffic Control Displays. CS Report 9213. Issue 1.2.

EUROCONTROL (1991a). Common Operational Performance Specifications (COPS) for the Controller Working Position, Version 6-91/1, Brussels: EUROCONTROL.

3.3 Relevant Human Performance Characteristics (Human Performance Model)

Most of the HF requirements which will be expressed within the ISG will be based on knowledge about human characteristics and behaviour drawn from research and experience in Human Factors, Experimental and Social Psychology, Cognitive Science & Engineering, Anthropology and in more recent years 'user engineering' [10]. This literature is not known to everyone who might be a user of the ISG, so it is common both to explain some of the most important ideas and to provide references to some of the more synthetic, relevant literature. The current Macintosh User Interface Design Guidelines are an excellent example of this practice, with chapters on 'Human Interface Principles' and 'General Design Considerations' supplemented by structured reading lists.

Whilst this is undoubtedly too extensive an exercise for most ATM activities, it can be very helpful to identify one or two basic references for the design team. The need for detail at this stage may be reducing in any case as the developers become increasingly educated in basic human factors. This will usually be less true for the users involved although, both in ATM and in the cockpit, we are fortunate that here too some basic HF training is becoming standard.

EXAMPLE: THE CORE BASELINE

The psychological framework employed for the original REFGHMI was described in [24], (Ch 6) and is included for reference as ANNEX 1. It is basically Norman's 7 Stage Model of Action [35], extended to distinguish between phases of 'monitoring' and integration' and 'active control' for ATC operators. This is important for the design of the HMI in that **it assumes that an important element of the controller's functioning is management of their own and the system's resources**. This must be systematically supported by the interface.

[X.HF.G1] (Manage resource/guideline): The system interface must support the user, as far as possible, in the task of managing resources (both their own, internal and external, and other system resources under their control/ responsibility).

ANNEX 1 also lists a number of important assumptions about controller scanning and intervention.

In addition to the above, Norman's related notion of **cognitive distance** [35], and the need to minimise it in order to minimise cognitive demand during evaluation and action, was a key concept in considering how to design for ease of use.

Subsequent revision for inclusion as the CoRe Baseline has been influenced by the work of Neisser [34] and Hutchins [29] in extending Ecological Psychology to the consideration of human-computer interaction.

ADDITIONAL NOTE: Since the original baseline was established, a more elaborate form of the model has been developed within the IMPACT Project. This model, ACOMOD (ATM COgnitive MODel), was based on discussion with a number of Human Factors practitioners who have had extended experience of HMI design in the ATM context. It was developed as part of a toolset for assessing the impact of change, induced by new ATM operational concepts, on the controller's way of working and thinking. Like Norman's original, ACOMOD is not a performance level model aimed at qualitative prediction. Instead, the model's role is to provide a conceptual framework for considering different aspects of human cognition and how they might react to change. The interest of the ACOMOD model is that it expands the framework of 'the action model' to associate two other aspects which are not normally linked within a single conceptual framework: 'regulation of resources' and 'management of internal factors'. ACOMOD also developed a glossary of terminology specifically to support discussion between the different stakeholders. Details can be found in [32].

3.4 Identification of the HMI Stakeholders and especially the Users

This is a key element of User-centred System Design (UCSD). Customers of a system are stakeholders but need not be end users of the system. It is important to identify all the stakeholders because all their requirements must be satisfied if the end system is to be implemented and accepted. However, it is the potential activity of the end users which is the basis for the design of an effective and efficient system. These users should be directly involved in the processes of establishing both functional and HF requirements and in the evaluation of design solutions.

Another issue that should be addressed at this level is the 'automation philosophy' [16] covering the overall approach and the relationship between the human and computational elements of the system, in terms of general roles and the allocation of initiatives and responsibilities. This is a subject of considerable interest and sensitivity within the ATM community. A clear policy and traceable implementation of that policy can be important in ensuring that automation-related issues are addressed.

The UCSD approach itself is intended to mitigate some of the difficulties that can arise if automation issues are not considered adequately.

EXAMPLE: THE CORE BASELINE

Stakeholders

The identity of the stakeholders will depend on the scope of the development activity selected (see §3.2 for the range of activities). At the minimum, they will involve the following roles:

a) The sponsors of the activity.

b) The domain experts for the activity.

c) The potential end users of any HMI developed if they are different from b) above. Even if they are from the same group, it may be necessary to have different individuals.

d) Designers of the HMI.

e) Developers (implementers) of the HMI.

f) Those responsible for evaluating the outcomes.

g) The managers of the process.

For the puposes of CoRe in its present form, it is assumed that participants from b) & c) will be operational experts with a specialisation in en-route air traffic control.

Participants from c) will normally hold a current licence.

Automation philosophy

Because of its evolution from ODID IV and the First PHARE Demonstrator (PD1) Baseline HMI, the automation philosophy underlying the REFGHMI was heavily influenced by the work of the PHARE 'Role of Man Group' as presented in general terms in [28]. In particular, [28] §2.3.1.2 describes a number of constraints which were accepted into those earlier systems and remain current in the CoRe Baseline. These are listed in ANNEX 2.

To summarise, the system is seen as user(controller)-centred evolution of present day systems with an emphasis on keeping the 'controller in the loop'.

The human and responsible agents remain the key agents in the activity supported by information processing tools.

The PHARE system assumed the introduction of some very advanced forms of controller support tools. This led to a number of important reflections on the need to be sure that the controller 'remained in the loop' ([24], §6.2.2.3) as the key responsible agent in case of unanticipated situations (see also §5.5, 'Designing for Interruptability'). The key elements of this discussion, which address issues of **understanding, responsibility** (see also §3.5 below), **liability** and the **possibilities for action**, are rehearsed in ANNEX 2.

3.5 Special Characteristics of the User Population

If any of the users have special characteristics likely to lead to additional requirements or constraints on the design then these should be identified at this stage so that they can be considered throughout the design process. Such characteristics may be physical (e.g. limited mobility), psychological (e.g. level of motivation, expertise, etc.) or a compound of both (e.g. age distribution).

EXAMPLE: THE CORE BASELINE

Expertise

The target user population is made up of air traffic controllers holding a current rating at the time of use.

The users are EXPERT users in terms of both:

- domain knowledge and
- their ability to employ their current operational interfaces.

The objective is to create interfaces on which these controllers (or their successors) will be able to perform appropriate ATC functions and follow evolved procedures, having achieved a suitable level of mastery of the interface within a short learning time.

Responsibility

In most European States the operational controllers carry legal responsibility for the safety of aircraft under their control. Quite correctly, this makes them 'powerful' users in terms of system acceptance. It also raises the design issue of how to ensure that this responsibility and the associated need and power to act are not diluted or reduced by the system design.

These issues are closely related to the discussion on 'keeping the controller in the loop'. (See §3.4, 'Automation Philosophy', and ANNEX 2).

Intention

Both the above factors imply a significant assumption underlying the design of the HMI in the baseline, which must be made explicit because it has very important consequences, not only for the flexibility but also for the 'vulnerability' of the HMI.

Essentially it is that the user is intelligent, responsible and well intentioned and will not deliberately seek to push the system beyond its limits, except in cases of emergency.

[X.HF.A1] (user skills/assumption): The HMI is designed on the assumption that the user is well intentioned and responsible, and has been trained to appropriate skill levels[8] to employ the interface appropriately under normal operating conditions.

[8] As we shall see later, this should not be read as implying that the interface is either fragile or difficult to use.

It also recognises that an **error** is not defined simply by the nature of an action but by the nature of the **context** in which that action occurs and the **appropriateness** of the action to the context.

It is recognised that the well-intentioned expert, operating real-time in the operational context, is almost certainly better qualified than the designer to identify the most operationally appropriate action. Hence neither the interface presentation nor the processes of interaction should dictate the choice of this action - although they can and should guide it (see also §5.5).

No special accessibility provision: absence of visual or sensory motor disability

It is assumed that the user population is selected to eliminate the need for special accessibility provisions. Both input device characteristics and the properties of visual presentation (size, contrast, brightness, colour vision, etc.) are based on values for the 'healthy normal' population.

To make the significance of this clearer: the design of such an HMI could be contrasted with that of a public cash dispensing machine. The cash dispenser has a limited functionality but must be robust against almost any imaginable user behaviour. This robustness has to be traded against other design criteria, such as speed and flexibility of operation.

3.6 Contextual Factors

No two classes of system development will be the same. There may well be contextual factors that imply modification of a development process, or that special attention be dedicated to certain aspects of system performance (safety criticality might be one example). While these could be considered and expressed as requirements at different stages of development, it may be helpful to highlight their significance from the initial stages.

EXAMPLE: THE CORE BASELINE

Contextual factors

The CoRe Baseline is somewhat unique in that it is designed for re-use in a variety of different contexts – as a generic HMI. In every context in which it is applied it will be necessary to identify the special contextual requirements which dictate a need for adaptation (e.g. if the HMI is to be used in a daylight tower environment).

Page intentionally left blank

PART 3: GENERAL DESIGN PRINCIPLES AND TRANSVERSAL REQUIREMENTS

Page intentionally left blank

4. INTRODUCTION

4.1 Contents and Scope

This part should cover the General Design Principles addressed in §2.4.2 and the **transversal** (or globally applicable) HMI requirements addressed in §2.4.3, that is:

a) Those general design principles, high-level human-machine interaction principles, characteristics of operator roles, which arise from the application of the design philosophy. Where appropriate explanation of the human perceptual or performance characteristics underlying the principles may be indicated.

b) Those transversal HF requirements (XHF) or constraints that potentially apply to all aspects of the interface.

4.2 Choosing a Structure

Rather than describing these in two separate layers, this ISG has made the decision to group the general design principles, and the requirements that derive from them, together so as to make the derivation clearer. In order to provide structure, these are grouped in terms of the HMI functionality to which they relate. The resulting structure is similar to that found in other ISGs of both the commercial and corporate types.

The chosen structure is grouped under three broad headings:

- Section 5: Principles of Interaction,
- Section 6: Principles for Input Management,
- Section 7: Principles for Visual Presentation.

The following paragraphs provide a little more detail on the content and coverage of each.

Page intentionally left blank

5. PRINCIPLES OF INTERACTION

5.1 Interaction Model and Metaphor

This introduces the ideas of the interaction model and of metaphor for providing the user with a frame of reference for generating expectations about how the system will behave. Basic vocabulary and terminology should be introduced.

EXAMPLE: THE CORE BASELINE

The following principles are identified as applicable to the CoRe Baseline. A number of these were identified in the course of definition of the REFGHMI [24] and the EATCHIP III HMI [17] prior to their application in CoRe.

Interaction model and metaphor

[X.HF.R1] (Direct manipulation/requirement): The Graphical user interface of the CoRe Baseline shall follow the principles of Direct Manipulation.

Explanation: In a true Direct Manipulation (DM) Interface, user inputs are made by manipulation of previously made system outputs [9] Ch1, Page 5. In the case of graphical DM, this process is conducted using an input device that performs operations on system outputs in the form of graphical objects presented on the display surface.

When the user interacts with the object the process of interaction is immediately apparent because of changes in the visual appearance or position of the object.

In the case of CoRe, the grammar of the interaction is a simple **Object-Action** Grammar involving selection of a graphical object and the subsequent association of an action with the object to create an input (from the user to the system).

The intention behind DM is to allow users to have a more **natural** interaction based on an exploitation of the same sensory-motor control processes as we employ in interaction with real objects in the world. (Contrast this with the abstract symbol manipulation processing associated with the use of language or text.) Good accounts of the assumptions and processes underlying DM are to be found in [38] and [39].

5.2 Behavioural Consistency

In order for the Direct Manipulation (DM) metaphor to be effective, the virtual world has to behave like the real world. The system behaviour therefore has to be predictable, which in turn implies that its behaviour should be consistent, i.e. the same event, applied to the same object in the same context, provokes the same response. This is the simplest form of a consistency rule. Consistency is however, a complex issue. Wilson [40] identifies at least ten different types of consistency, which may be relevant to HMI development.

The paragraph below explains how this is simplified and implemented in the CoRe Baseline.

EXAMPLE: THE CORE BASELINE

In order for the DM metaphor to be effective, the virtual world has to behave like the real world. The system behaviour therefore has to be predictable, which in turn implies that its behaviour should be consistent, i.e. the same event, applied to the same object in the same context, provokes the same response.

For CoRe, it is possible to simplify and interpret Consistency in terms of two requirements and two guidelines related to consistency:

[X.HF.R2] (Basic Consistency/requirement): The same user action applied to the same object in the same context shall result in the same behaviour.

[X.HF.R3] (Consistency for the user/requirement): The same user action applied to objects in the same class in contexts that are perceived to be similar[9] shall result in similar behaviour.

[X.HF.G2] (Support for generalisation/guideline): Similar user action applied to similar objects in similar contexts should result in similar behaviour.

As a related corollary, it can be very tempting to identify ingenious mechanisms specific to the manipulation of particular parameters, thus emphasising local optimisation over global consistency. It is the opinion of the authors that this is a frequent and serious error leading to interface complexity. Exceptions may be appropriate in the case of very frequently repeated operations by expert users ('shortcuts') but, even then, the (lengthier) mechanism predicted by the global rules should also be available to the user. Hence,

[X.HF.G3] (Global optimisation/guideline): Unless there is an overriding operational justification, global consistency should take priority over local optimisation

The two guidelines relate to both usability and the 'learnability' of the interface.

[9] "Similar" relates to the user's perception that sufficient properties and behaviour are shared. It relates to the associated user expectations. The presence or absence of these perceptions and expectations can only be established through user involvement.

5.3 Policy on Modes and Modality

'Modal error' is commonly cited as a critical factor in certain risk or even 'incident' situations, but modality is a difficult concept to understand. Modality should firstly be introduced and explained before considering how it will be addressed within the HMI.

EXAMPLE: THE CORE BASELINE

In interface terminology, an interface is **modal**[10] when the same action results in different effects.

The argument for modes is that they allow a much greater range of functionality to be made available in response to any given range of input possibilities.

Modes are controversial because they are recognised as a potential source of error, arising when there is confusion as to the current mode status. Users perform an action, misunderstanding the mode, and the resulting behaviour does not match with their expectation or intention. This is known as **modal error** and potentially has very serious consequences.

In order to protect against modal error some HMI guidelines suggest that modes should be completely avoided. Others argue that the problem lies with the confusion between modes and not in the existence of modes themselves. If there is no ambiguity about the presence of a mode, the risk of modal error can be reduced or removed. CoRe follows this position. If we return to the definitions of consistency provided in § 5.2 above, the issue of 'similarity of context' is directly relevant. The problem becomes one of making context clearly explicit and is addressed by the following CoRe guideline:

[X.HF.G4] (mode policy/guideline): Modes are permitted provided that there is no ambiguity concerning the existence of a mode and the outcomes of otherwise potentially ambiguous actions. Criteria: no modal error is detected in trained subjects.

As safeguard, CoRe (following the REFGHMI and EATCHIP) adopts two additional guidelines:

[X.HF.G4-1] (mode policy/guideline): The duration of modes should be kept as short as possible.

[X.HF.G4-2] (mode policy/guideline): It should be possible to escape from a special mode and return to the default mode as quickly as possible.

---

[10] As an example of a **mode** consider a Word processing/text editing application. Most such applications support two editing modes: **insert** mode and **overwrite** mode (in MS Word™ the modes can be exchanged by pressing the INSERT key once). In **insert** mode, striking any alphanumeric key on the keyboard, when the text cursor is inserted in the middle of a word, inserts the corresponding character at the cursor position and moves the following letters one position to the right. The cursor also displaces to the right. This is the commonest mode in word processing. In **overwrite** mode, thesame keystroke results in the character to the right of the cursor being replaced by the character corresponding to the keystroke, with the cursor also displacing one place to the right.

5.4 Level of User Awareness

This introduces the notion of focus and explains the importance of allowing the user to focus on the operational task activity, not on managing the HMI.

EXAMPLE: THE CORE BASELINE

A direct manipulation graphical user interface is intended to furnish a 'natural' interaction; one that taps into the same mechanisms used for performing tasks in the real world.

In the handling of everyday objects, much of this manipulation is carried out automatically, i.e. with little conscious attention being allocated to the control of the manipulation itself[11]. Instead, conscious attention is focussed on the objectives and outcomes of the process. When writing or typing, for example, attention is usually focussed on the tasks of composing and expressing content, not on the mechanics of manipulating the pen.

The target with the DM interface is similar. The user should only be 'thinking' in terms of the task activities and never in terms of the mechanics of the interface. Intentions (as characterised in the model in ANNEX 1) should be task-related intentions. The user should be thinking in terms of operational actions and consequences. This is critically dependent on the intuitiveness and ease of use of the interface.

[X.HF.R4] (Level of intention/requirement): The interface should allow the user to normally maintain a focus at the task (operational) level and not the interface level. Criteria: based on probing user's intention verbal protocols, etc., level of descriptions obtained.

[11] Under normal operating conditions. When difficulties are encountered with the control processes, attention is switched to them.

5.5 Explicit Design to Support Resource Management and Monitoring

This introduces one of the specific themes of the CoRe (and REFGHMI) interface family: namely, that the skilled operator is also managing his own and the resources of other elements of the system. The interface must support this.

EXAMPLE: THE CORE BASELINE

Paragraph 3.3 identified that an important role of the HMI is to support operational users in managing the resources and the task, both for themselves and for other system elements. This was expressed in a CoRe guideline [X.HF.G1]. The following general design principles and practices are intended to improve an interface's ability to provide that support.

Role of procedures and interface mechanisms

Both procedures and interface mechanisms are part of the interaction which must be designed. In CoRe's opinion, when working with expert professionals, the rules and limitations should be built into the procedures rather being 'hard-wired' into the interface code. There should be some separation of operational procedures (which, in any case, cannot often be well defined at the initial system specification stage) and the mechanisms of the interface. Thus, while the interface is designed to support the procedures, it should not be over-optimised to accommodate the finer details of the procedure. If the operational procedure has to be revised, the interface should not necessarily need to be changed.

[X.HF.G5] (procedures/guideline): Rules and constraints should be primarily implemented in the procedural part of the interaction rather than hard wired into the interface

This has special implications for the error management style adopted by the interface. Robustness and error tolerance should come from (be designed into) the interaction of the different agents and procedures. Highly specific error monitoring and prevention based on procedures should be supported by, but not normally be 'hard-wired' into, the code.

NOTE: This is certainly true of interfaces designed for R&D purposes and simulation studies, but there may be cases in an operational application where a specific 'safety requirement' means that some actions should be impossible in some circumstances. Such cases should be clearly identified as safety requirements for the design and their consequences fully understood.

Otherwise the designer has to assume that the user is intelligent, responsible and well intentioned and will not deliberately seek to push the system beyond its limits except in cases of emergency (see assumption [X.HF.A1]). The system does not try to forbid potentially destructive actions. Specific inputs are not made impossible since, with a change in operational procedure or task partitioning, they may become useful. The design therefore has to become error tolerant rather than error free. This is not incompatible with the tactic expressed in §6.8, 'Strategies for Design', b), concerning robustness against certain classes of error.

Initiative for action stays with the controller – designing for interruptability

As far as possible, the controller should not be forced to be reactive to the interface but should be presented with the information which permits ATM to be adaptive to the operational context, i.e. which allows the controller to decide how to allocate resources and priorities to tasks generated by the traffic.

The REFGHMI baseline [24] §6.2.2.3.1 identified a number of additional guidelines for keeping the user in control of priority and resource management. These relate to the view that skilled ATM is not well characterised as a series of tasks carried out in sequence but ***is better described as a number of threads of activity, being carried our in parallel with the controller switching between them expertly*** (**ANNEX 1**). Supporting this means that the controller must be able to halt one activity to divert to another and then revert to the initial one at some later stage without paying a high overhead. Reference [24] describes this as **designing for interruptability** and suggests the following guidelines and requirements to ensure that the interface is open and flexible:

[X.HF.G6] (Interruptability/guidelines): The interface should not require long related input sequences that have to be followed through without interruption (see ANNEX 1, c)).

[X.HF.G7] (Retain work/guideline): When switching tasks the system should help the user by retaining results of partially completed tasks so that they can subsequently be recovered[12].

To support low cost task switching:

[X.HF.R5] (Prioritisation information/requirement): The interface should provide the user with the information necessary to decide when to change tasks and priorities.

[12] Guideline [X.HF.G7] is closely related to the guidelines for limiting the consequences of modes (§5.3). If there are any closed modes, then this guideline may be hard to achieve.

5.6 Flexibility in the Use of the Interface: Multiple Paths and Expert Use

Another CoRe specific feature, this relates to the fact that the same function may have to be called in different contexts and that there may have to be a number of logically coherent ways of accessing that same function. This sub-section describes the rationale and the approach.

EXAMPLE: THE CORE BASELINE

A further guideline derives from both the preceding arguments and from [X.HF.A1], which expresses recognition of the expertise of the users. This deals with the fact that, in complex interfaces (for complex application domains), it is not possible to assume that the designer has specifically identified every possible way in which the interface will be used and every possible state that it may achieve.

The design must be flexible and tolerant because, at the detailed level, we do not know all the ways in which the expert may make use of it.

Two guidelines are related to this requirement for flexibility:

[X.HF.G8] (Availability/guideline): Many basic functions will need to be easily accessible in several different contexts. There may need to be several ways to address the same function in which case they must be consistent[13].

[X.HF.G9] (Exploratory behaviour/guideline): The interface should not prevent the controller from seeking innovative, but viable solutions to unusual problems.

This reinforces the previous assumption of a need for a tolerant approach to error management. The system must assume that the controller is an intelligent and responsible agent, and allow exploratory behaviour.

[13] A good example of this is that the Callsign Menu is accessible with a single click of the action button, anywhere the callsign is available. Thus the controller can access the same callsign menu from the radar label or any list or other display where the callsign is present.

5.7 Economy

Economy is addressed within the context that there is a cost (in terms of workload) associated with interaction and that this should be minimised. Background material relating to the notion of 'distance' in the interface, and to cognitive and physical economy, can be found in [35] and [38].

EXAMPLE: THE CORE BASELINE

Within the constraints of the guidelines already described in the preceding paragraphs:

[X.HF.G10] (Economy/guideline): The interface design should keep required input actions to a minimum both in terms of the cognitive effort associated with them (cost) and their number.

6. PRINCIPLES FOR INPUT MANAGEMENT

6.1 Input Basics

This sub-section describes the basic selection model, the general nature of the input devices and the 'grammar' of interaction. In direct manipulation interfaces this is usually an object-action grammar.

It explains how objects are selected and the use of feedback (in this case, highlighting). Highlighting principles are described.

EXAMPLE: THE CORE BASELINE

We have previously noted (§5) that the user input will be based on an object-action grammar, i.e. the user selects an object and applies an action to that object. This sub-section describes how this is done in the CoRe Baseline interface. Almost all of the material is reproduced nearly unchanged from [17] and [24]. It has also been reproduced and used in the Denmark Sweden Interface (DSI) [42] and the Italian Interface (ITI) [43] projects based at EEC. Its status can be considered as relatively mature.

Selection model and highlighting

Selection of an object is usually by means of a **pointing device**. The OSF/Motif Style Guide [7] describes a pointing device as:

- letting the user move a pointer (cursor) around on the screen,
- allowing the user a means of activating the object under the pointer.

An object which can be manipulated (i.e. through which an input is possible) is known as a **selectable** object.

An INPUT EVENT is defined as an event that the system is capable of distinguishing. Input Events correspond to the primitives, which build the **action** element of the **object-action** model (e.g. the depression of a mouse button).

An ACTION is defined as one or more **input events** in a sequence, which correspond to User Identifiable Actions. These are normally the **actions** of the **object-action** Model (e.g. opening a menu by a mouse button click).

There may be need to 'highlight' an object in order to provide identification feedback that it will be the object on which any subsequent selection input acts[14]. This implies the existence of an implicit input event, which triggers the appropriate highlighting function when the cursor crosses the **acquisition** boundary of the selectable object in question. This boundary crossing input event will also be required when leaving a selectable object to remove the highlighting and de-select the object.

[14] Objects like buttons are always selectable and, provided these have a visual design which makes this clear, there may be no need for another feedback mechanism to emphasise their selection.

[X.HF.R6] (Highlighting principle/requirement): By default any field or object that is selectable will provide a feedback (highlight) to show selection acquisition and an 'inverse' feedback to show de-selection.

[X.HF.R6-1] (Highlighting exception/requirement): For cases where highlighting is not available, the justification should be provided.

[X.HF.R6-2] (Highlighting response time/requirement): For the feedback to be effective an appropriate response time should be defined for each selectable object. In absence of definition ≤125 milliseconds is assumed (see §6.3 and also [X.HF.R6-5]).

NOTE: An important distinction is made between the different states that follow in composing an input. Firstly the cursor is moved onto a **selectable** object. This object is then described as selected. When the user chooses which action to apply to the object and applies it, the object is said to be **activated**. These two stages directly reflect the object-action grammar.

Focus and selection model

It is a design assumption of the interface that the **focus of attention** of the operator within the interface is normally[15] represented by the cursor on the display surface. That is, that the **focus of attention** and the **focus of interaction** should coincide.

Consequently, the **focus model** employed will be implicit, i.e. all user inputs will go to the locus of the mouse cursor. (This is in keeping with the recommendation of the Motif Style Guide [7].)

In support of focus, **cursor warping** will be avoided as far as possible; the cursor will be displaced solely in consequence of user intent to do so (for example, by launching a process that the user already knows will displace the cursor to an already identifiable location). This is also in keeping with the Motif Style Guide.

Where objects are **selectable** (see below), the **selection** model will be **explicit**, i.e. a specific **input action** (for example a mouse button click) will be required to complete the **activation**.

Highlighting principles and mechanisms

Highlighting is treated as an attribute of selectable objects. Not all selectable objects will have the attribute enabled. When the attribute is enabled, an appropriate mechanism must be defined.

[15] Of course there will be times when users make eye movements away from the cursor but, if they have undisputed cursor control, they would normally be able to return to the origin of their excursion and resume 'normal navigation' through the task and the interface.

The following principles govern the processes of highlighting employed in the HMI:

[X.HF.R6-3] (highlighting activation/requirement): The object shall become selected only when the **acquisition boundary** of (an instance[16] of) the object has been crossed by the cursor. (The area within the acquisition boundary is called the **acquisition area**.)

[X.HF.R6-4] (highlighting de-activation/requirement): The object shall cease to be highlighted only when the cursor crosses the **retention boundary**, or when the presentation of the object changes as the result of an input action. (The area within the retention boundary can be called the **retention area**.)

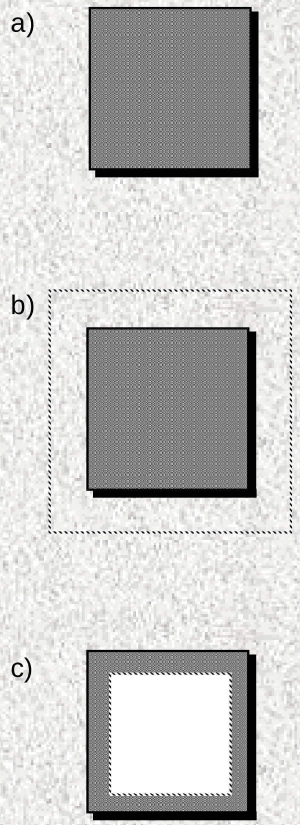
[X.HF.R6-5] (highlighting latency/requirement): There may be a latency parameter associated with highlighting such that the object highlights only when the cursor lies within the two boundaries for more than a certain time, i.e. the value of the latency parameter.

The acquisition and retention boundaries need not be identical. Further, they need not correspond to the visible boundaries of the displayed object.

This is illustrated for the case of the acquisition area in Figure 3. The Figure illustrates three interesting sets of conditions:

- In the first, (Figure 3a) the acquisition boundary and area correspond to the visible boundary and area of the object. The object highlights as the cursor enters the object. This is the most natural and intuitive structure.
- In the second (Figure 3b) the acquisition area is significantly larger than the visual area of the target object. This is particularly useful for the acquisition of small objects which are reasonably well separated from others of their layer. A good example is provided by Beacons or Way Points in a Radar Plan View Display. This function is comparable with the 'Snap-to' function of some drawing programmes.
- The third example (Figure 3c) is the case where the acquisition area is smaller than the visible area of the target. The target only highlights when the cursor is well within the visible boundary. This is effective where a number of medium sized targets are close together or share boundaries, and reduces mis-selection. An example could be the data entry fields within the radar label of an aircraft on a Plan View Display.

[16] There are cases in the HMI where selection of one instance of an object results in the cross highlighting of all other instances (e.g. selecting an aircraft in a list highlights the label).



Application: Default selection of objects. Best hand/eye feedback

Application: Small objects, far apart, to reduce aquisition time.

Application: Good sized objects, grouped very close together. To reduce mis-selection and overshoot errors

In all the above examples, acquisition area and retention area are assumed to be the same.

Figure 3: Possible relationships between the visible area of an object and the acquisition area (shown by the dotted line)

- Figure 4(a) shows a menu for the selection of flight levels. Classically the user moves the cursor over the menu and the entry that would be selected if the action button were to be clicked, highlights in a different shade.
- Figure 4(b) shows what would happen if the cursor moves out of the menu in the 'normal' case (i.e. where Visual area = acquisition area = retention area). In this case, the potential selection is lost (indicated by the loss of highlighting).
- The case of a 'sticky' menu is shown in Figure 4(c). In this case the retention area is > the visual area and within the limits of the retention area the ability to make selections continues. (To close the menu without making a selection, the cursor is simply moved away until no element is highlighted before clicking a button.)

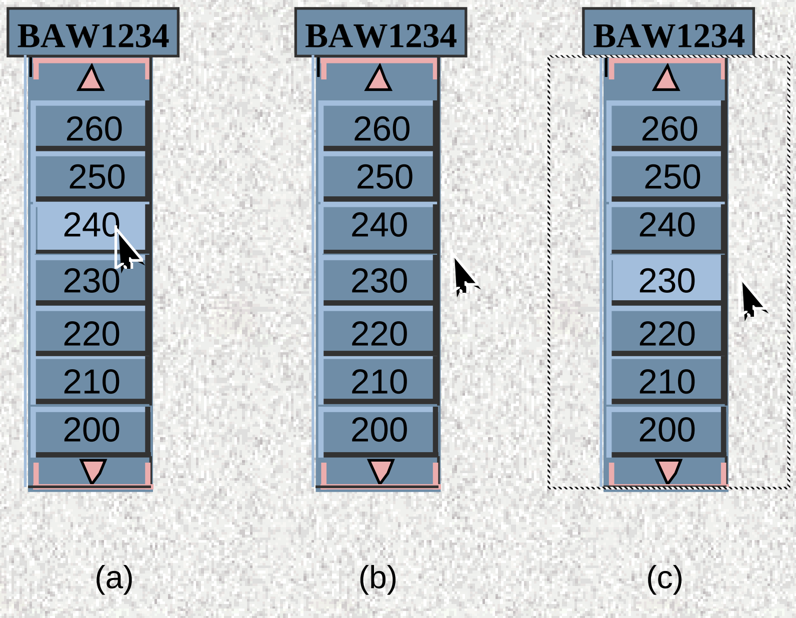


Figure 4: The effect of a retention area which is greater than the visual area (and also the acquisition area of a screen object, see text)

[X.HF.R7] (boundary definition/requirement): Acquisition and retention boundaries must be defined as attributes, for each selectable object. (The boundaries are defined in millimetres[17] relative to the visible boundaries of the object.)

[X.HF.R7-1] (boundary default/requirement): If unspecified, the default assumption is that all boundaries correspond to the visible boundary.

---

[17] Of course it is the visual angle presented which is important.

Discussion of highlighting mechanisms

Highlighting of an object is usually shown by a change of colour of the object and, if the object supports text, a change of text colour which preserves the legibility of the text.

For some classes of objects, specific highlighting colours are identified along with the original colour definition, e.g. Window Blue[18] exists as Window and as Window Highlight.

For other types of objects, particularly those for which colour is a variable, there may need to be a rule which defines the highlighting mechanism. For example, for radar labels there is the 'selected label' form which represents the highlighting of a label. However, there may also be a need for a secondary highlighting mechanism which deals with highlighting of the individual fields within the label and this should be defined along with the description of interaction with the parent object (in this case the radar label itself).

Finally, there may be a need for a general rule for defining highlighting principles, which can be applied to any object. This is not easy. The only suggestion which is made at the moment (**but which must be subjected to evaluation in each potential application context**) is that ALL the elements of the selected object turn to their colour complement, i.e. that the RGB percentage values (x, y, z) are replaced by (100-x, 100-y, 100-z). This mechanism should have the advantage of preserving the internal contrast and structure of the object and the relative contrast of any text.

[18] See ANNEX 4.

6.2 The Accessibility of Selectable Objects

In order for objects to be selectable, it must be possible to see the objects and reliably position the pointing device on them. This sub-section describes the criteria for achieving this and derives the minimum size of selectable objects in terms of Fitts' Law.

EXAMPLE: THE CORE BASELINE

Accessibility

Direct manipulation graphical interaction, based on the location of a single pointing device, has a number of characteristics.

a) The number of alternative inputs is largely governed by the available selectable screen objects [19].

[X.HF.R8] (Target access/requirement): To be selectable a screen object must be visible to the user (i.e. not completely overlapped by other screen objects).

[X.HF.R8-1] (Target availability/requirement): If a screen object is visible its availability for input shall be clearly indicated to the user either through a convention or a specific encoding.

b) Input to the system is completely serial. The device can only point to one object at a time.[20] That object is the focus of the current activity.

Fitts' Law

The ability to point to an object is governed by Fitts' Law, which relates the movement time (MT) taken to acquire a target to the width of the target along the direction of approach (W) and the distance (D) which the pointer must be moved.

$$\mathbf{MT = a + b \log_2(2D/W)}$$

where **a** and **b** are empirically derived constants.[[49], §9.201]

For the mouse this can be approximately[21] re-expressed as:

$$\mathbf{MT = 100 \log_2 (D/W + 0.5)}$$

[19] It is also dependent on the range of compositional rules available.

[20] At the time of writing, there is no intention to implement a 'multiple selection' function of the type that can be found on personal computers. This function permits several objects to be selected simultaneously, either by the use of a 'rubber band' mechanism or by holding down the shift key while clicking to make selections without de-selecting previous choices.

[21] This alternative form of Fitts' Law was derived by Welford [50]. The value of 100 for the slope constant is extrapolated from Card, Moran and Newell [51] as representing a very conservative value.

A worst-case value for D on today's display technology is 685.8 mm (27" diagonal) although a more typical value would be around 30.0cm for the Radar PVD and tools. Under these conditions a minimum target width of 4mm[22] along the direction of movement would result in reasonable MT values of between 620 and 750msec.

Note: This size also corresponds to the minimum recommended character height .

[X.HF.R8-2] (Target Size/requirement): Selectable objects shall be at least 4.00mm across the shortest axis along which they will be acquired. Applies to both text and non-text selectable objects

[22] Since D/W is a ratio, it does not matter whether the values are expressed as visual angles or screen sizes. For reference, the 27" diagonal of a Sony or Intergraph display presents a visual angle of 57° 7" at the estimated mean viewing distance of 0.63metres. It is also important to note that this is the size of the acquisition area associated with a target. If suitable feedback as to target acquisition is provided, this can be bigger or smaller than the extent of the visible target.

6.3 Response Times

The responsiveness of the system is a key element in establishing and maintaining a satisfactory quality of interaction. This sub-section describes issues underlying response time requirements and establishes the guidelines that should be applied to the CoRe Baseline.

EXAMPLE: THE CORE BASELINE

Priorities in response

With minor qualifications, the practices recommended in this sub-section follow those already recommended in the EEC REFGHMI [24].

From an engineering perspective, the priorities in a graphical interface may seem counter-intuitive. They can be expressed quite simply.

[X.HF.G11] (priority response/guideline[23]): An **initial** response to user input, indicating recognition that an input has been made, should take priority over any other activity at the interface.

The 'other activity' includes even such apparently essential priorities as updating of the radar display. The logic is simple. If the user's focus of interaction is located at the input cursor, then that is where the processing priority should be. Users will notice a lag in response to their input, but will NOT notice a lag in the update of the radar display because they will not currently be looking at it. At times when they *are* monitoring the radar display they will not be making inputs, so there will be no conflict in updating priority. It is for this reason that most Macintosh or Windows programmes begin with a mouse monitoring event loop. The behaviour of an ATM interface should appear to be the same.

Although these priorities may seem somewhat strange to an external observer watching controller activity at the interface, it is the responsiveness to the user performing the task which matters.

This requirement does not mean that all system activity has to stop while any sequence of user inputs is followed to completion. It does mean that initial feedback of the fact that an input has been made and parsed should appear to occur immediately, e.g. in less than 125 milliseconds. This may be little more than a change in state of the cursor to show that input is being processed.

[23] This has the status of a guideline simply because it is not enforceable on some of the system architectures currently used by ATC, where events are handled on a 'first come-first served' queuing principle. In this case care must be taken to avoid, or to clearly indicate the existence of, 'blocking events' which will delay response to the input. The use of a 'busy' cursor may be an appropriate solution. The best solution is for such tasks to be asynchronous and run in the background so that they do not block further actions.

NOTE: This is a very demanding requirement. Slower response times MAY be acceptable to users if they are aware that a complex process is being launched within the system. However, even in this case, good design practice implies that some feedback acknowledging the input should be provided as quickly as possible.

Some **implicit inputs**, such as highlighting of objects to provide feedback on cursor location, may have to be processed even more quickly (within a few screen refresh cycles) to be useful. Figure 5 illustrates the sequence of times involved for a very simple input, such as the appearance of a 'pop-up' menu when a mouse button is depressed.

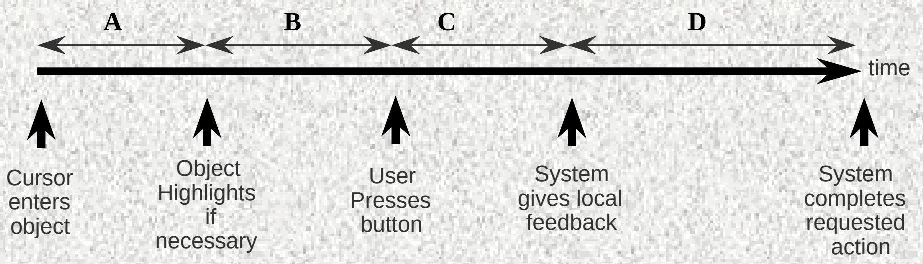


Figure 5: The time intervals associated with a simple input

Figure 5 illustrates the following sequence:

- Interval A: The cursor enters a selectable target. In some, but not all cases (e.g. a window slider), this requires immediate (explicit) visual feedback. This type of feedback (interval A) must occur within a few screen refresh cycles or there is a risk that the cursor will have moved on into another screen object. This corresponds to the Screen Update Response Time (SURT) described in [[22], p2.29].
- Interval B: This is the user response time required to recognise successful acquisition. In (implicit) cases where no explicit feedback indicating acquisition is provided, A+B becomes a single interval but, since the user has to make a more complex judgement, A+B will be longer. The sum of intervals A and B (for both explicit and implicit conditions) is often referred to as the 'target acquisition time'.
- Interval C: When the user makes an active input (by changing the state of a mouse button), some form of 'immediate' acknowledgement of the input must be provided within around 125 milliseconds. In cases where the input can be fully processed within 125 milliseconds (i.e. interval C+D < 125 milliseconds[24]) the successful outcome of the resulting process will suffice - provided that there is a consequential change in the visual environment. In cases where the system requires longer, visual feedback acknowledging the input and informing the user of the delay should be provided by a change in cursor format to a 'busy' cursor within the 125 milliseconds time frame. The cursor will change back to an appropriate form on completion of the outcome.

[24] This is a conservative value. The justification for this figure is that it should result in the response appearing 'immediate' to the user. 125-150 milliseconds is conventionally the shortest time in which a user can make a voluntary eye movement or 'see and react' in a basic motor control loop. As a consequence, it is also a good basis for the feedback loop controlling drawing actions, e.g. 'elastic vector' or 'trajectory editing'. See also 'IMPORTANT NOTES ON RESPONSE TIMES'.

[X.HF.G11-1] (Feedback response time/guideline): For good interface engagement a minimum feedback response to an explicit user input should be provided in ≤ 125 milliseconds (perception of immediate response). See 'IMPORTANT NOTES ON RESPONSE TIMES' below.

- Interval D: Once 'immediate' input feedback has been provided, interval D represents the balance of the time until the system completes the processing initiated by the input. The sum of intervals C and D may be considered as the system response time to the user's input. It corresponds to the Internal Processing Response Time (IPRT) described in [22], p.2.29.

The sequence in Figure 5 is extended for other types of input such as PRESS-DRAG-RELEASE but, in reality, this involves little more than a cycling around the 'C' component while the cursor is repositioned and then the mouse button is released.

Consistency of response time

[X.HF.G11-2] (feedback response consistency/guideline): Consistency of response time is more important than the absolute time of response.

This is particularly true for the immediate feedback to inputs and the process of continuous control. If there is variability in this response, it generally leads to high levels of frustration and dissatisfaction. It can also lead to incorrect input sequences when users repeat inputs that have not been acknowledged. The resulting queued events make the interface perform in apparently unpredictable ways, destroying the illusion of direct manipulation.

IMPORTANT NOTES ON RESPONSE TIMES

1. The response time of 125 milliseconds is a conservative value. It is chosen as a reasonable answer to the question, "At what speed of response will we never have any problems?". Slower responses may be acceptable in many circumstances. Indeed, universal application of this criterion may result in serious over-specification. Nevertheless, where other values are proposed, care must be taken to verify their acceptability both in usability and operational terms.
2. In some system architectures, measurement and verification of response times is difficult to achieve. Care must be taken in any contractual arrangements to ensure that all parties are in agreement as to both the appropriateness and achievability of the agreed response times, and the means of establishing compliance.

6.4 Interaction Intent: Action versus Information

The controller may wish to have access to the same information for different reasons. The ODID studies identified two different styles of data access: for ACTION and for INFORMATION. This sub-section explains the distinction and how these two styles are supported by the baseline interface through specific requirements.

EXAMPLE: THE CORE BASELINE

For the purposes of the CoRe Baseline two basic classes of interaction are presumed and implemented. The distinction is supported by the use of separate input elements (in this case, mouse buttons).

The distinction is derived from the later ODID studies [19], [18] and [23], and has been a feature of most subsequent advanced ATM simulations.

The first class is based on the idea of an intent to action. In the current interface this class shall be realised through mouse button B1. This is known as the ACTION Button (AB).

The second class is based on the idea of interactions with an intent to organise (access or remove) information. This shall always be realised through mouse button B2 or B3, depending on the mouse employed. This is known as the INFORMATION Button (IB).

Within this second class of information access/organisation, two different **interaction styles** are required:

[X.HF.R9] (Access Quick Look/requirement): The interface shall provide the capability to display objects or information temporarily, removing it after the controller has obtained the desired information, with no further intentional action on the controller's part. (ODID Quick Look)

[X.HF.R9-1] (Access Sustained/requirement): The interface shall provide the capability to display objects or information which remains displayed on screen for successive use by the controller, and is removed only as a consequence of a subsequent, specific controller action.

6.5 Policy on Default Selections

Default selection is a difficult issue and has been the source of much discussion and revision within ODID type interfaces.

The issue deals with the extent to which 'shortcuts' can be provided for routine patterns of operational interaction.

It relates mainly to the use of menus to select parameter values or choose actions (e.g. select an exit flight level, or 'assume' an aircraft). Expert users request that, in contexts where they are most likely to make certain inputs, the menu should open in such a way that the cursor is already positioned on that input ready for their selection.

While there are a number of resolvable secondary issues related to techniques for implementing this (e.g. cursor warping, consistency of menu structure) the main human factors concerns relating to defaults are:

- that the default helps in routine situations, but is actually an obstacle when a non-routine choice has to be made; the interface is 'pushing' towards a certain choice when another has to be chosen, and this may make the appropriate selection more difficult;
- the risk that these local optimisations reduce the overall consistency of the interface selection mechanisms.

This is an area of delicate trade-offs. The emphasis is on ensuring that any negative consequences of the final design choices are both understood and minimised.

EXAMPLE: THE CORE BASELINE

The main requirements related to menus are included in §8.3.2. In this sub-section we address only those requirements relating specifically to the management of defaults.

The baseline accepts that the apparent operational penalty of excluding the use of defaults in certain conditions can be sufficiently high that it may lead to difficulties with user acceptance as well as contravening other requirements relating to the efficiency of the interface.

The baseline policy is expressed in the following guidelines:

[X.HF.G10] (menu defaults/guideline): Defaults should be avoided if possible but, where they are deemed necessary, care should be taken to (a) limit their tendency to impede non-standard inputs (they should never block them) and (b) ensure that consistency of interaction is maintained.

[X.HF.G10-1] (menu defaults/guideline): Where a default rule is applied, its existence and nature should be clearly understood by the users.

A family of requirements is developed to support these guidelines.

[X.HF.R20] (menu defaults/requirement): The use of menu defaults shall not lead to inconsistency in menu presentation or in the rules for user interaction with menus.

This requirement is realised through additional, more specific, requirements:

[X.HF.R20-1] (menu defaults/requirement): The order of menu items[25] shall not be modified in menus for the input of operational parameters or actions.

and

[X.HF.R20-2] (menu defaults/requirement): If the menu should open on a specific selection, the opening menu shall be displaced vertically to put the appropriate value under the cursor[26] which shall not move from the point of last user action.[27]

Additionally there is a particular problem associated with the use of defaults in the input of parameter values, such as 'cleared flight level' (CFL). When the menu is opened from the existing parameter field, it obscures the current value of the parameter. If there is no default rule in place then the menu will open with the current value under the cursor. If there is a default rule such as 'default to exit flight level' then the current value is retained only in the memory of the user. This prompts one further guideline.

[X.HF.G10-2] (menu defaults/guideline): In cases where a parameter selection menu defaults to a value, there should be a method or coding which allows recognition of the currently assigned parameter value.

[25] Generally the contents of menus are fixed. Items not available in a certain context will be present but 'greyed-out'.

[26] Care must be taken to confirm that there is no negative interaction with mechanisms for managing menu presentation at the vertical limits of the display surface or window.

[27] Avoiding 'cursor warping'.

6.6 Input Devices and Input Events

This sub-section looks more closely at the requirements for interaction devices (in this case a mouse, pointing device). It provides requirements for the assignment of mouse buttons to different classes of dialogue function.

EXAMPLE: THE CORE BASELINE

[X.HF.R10] (pointing device/requirement): It shall be possible to dialogue with the system and access all functions initiated through designated screen objects representing data using a suitable (mouse) pointing device.

Dialogue may differ on a particular object when the state of that object has changed e.g. XFL status during coordination will be different to the normal status.

[X.HF.R10-1] (input device/requirement): Controller input to the system shall use a three-button (B1, B2, B3) or two-button (B1, B3) mouse.

Buttons are numbered left to right, B1, B2, B3.

[X.HF.R10-2] (button classes/requirement): Each numbered button shall have a specific associated input action type.

[X.HF.R10-3] (buttons hand/requirement): The B1 functions shall be interchangeable with B3, as an option for left-handed subjects.

The CoRe Baseline Interface system uses a pointing device, a mouse, as the primary input device. One of the evaluation issues will be the relative merits of three-button versus two-button mice. In principle, the mouse could be replaced by any continuous pointing device, which also supports the required number of input actions.

[X.HF.R10-4] (button assignment / requirement)

Button	Name	Usage
B1	Action Button (AB)	All normal inputs To initiate a system dialogue and to input a new value into the system (modifications to flight levels, to routes etc., restrictions, such as direct route, heading, climb/descent rate, etc.). All window management actions e.g. re-size, move, scroll, etc.
B3	Information Button (IB)	Quick Look mode or Fixed (permanent) display of information
B2	Special Function Button (SB)	Reserved for special use, described when relevant.

6.7 User Actions

This sub-section shows every combination of input primitive and (mouse) button allocation. It thus describes the complete set of input actions (the input vocabulary) supported by the CoRe Baseline system.

EXAMPLE: THE CORE BASELINE

The possible input actions are presented in the following table:

[X.HF.R19] (button action definitions/requirement)			
Action	**Mouse button**	**Result**	**Description**
Point	Mouse pointer	Makes an implicit selection.	The mouse pointer is moved until it is located within the Acquisition Boundary of a target screen object.
Single Click (SC)	**B1 (AB)**	Performs an action.	The mouse button is pressed and released in rapid sequence, i.e. the Button DOWN and Button UP events occur within the Click Interval. The Click Interval is a time parameter that should be accessible in the user preferences set-up. The default setting of this parameter shall be 300 milliseconds[28]. The SC action makes an explicit SELECTION.
	B3 (IB)	Displays or removes information in Fixed mode.	
	B2 (SB)	Performs the “special use” action.	
Press and Hold (PH)	**B1 (AB)**	Performs some of the window management actions (scroll).	The mouse button is depressed for a time greater than the Click interval (see above).
	B3 (IB)	Displays information in Quick Look mode.	
Press and Drag (PD)	**B1 (AB)**	Performs some of the window management actions (move, re-size).	The mouse button is depressed while the pointing cursor is positioned on a movable (or re-sizeable) screen object. The pointing device is then moved to a new position dragging the object, or a ‘wireframe’ representation of the object, with it. When the button is released, the screen object is re-drawn at the new location.
Release	Any button	Represents the end of either of the above actions (PH or PD).	The mouse button is released.

NOTE: Double clicking is not a basic input action of the CoRe Baseline (§6.8, ‘Strategies for Design’, e)).

[28] This is the parameter currently used on EONSCLICK to separate presses from clicks, based on controllers in simulation.

6.8 Input Error Management Policy

This is an important sub-section. It describes the overall policy for dealing with errors of input and interaction. The approach chosen is based on the recognition of controllers as experts and an emphasis on responsible users who must be allowed to exercise their responsibility.

EXAMPLE: THE CORE BASELINE

The responsible interface: rationale

In §3.5 we recognised expertise and responsibility as characteristics of the users of the ATM en-route interface. In §5.5 we began to address some of the consequences for the way in which users can remain in control of resources in order to responsibly exercise their functions. These functions include the management the monitoring and regulation of their own actions and the detection and management of errors. How do we make an interface to do this?

The key to the present approach is the recognition that the understanding of errors depends not so much on the nature of an **action** but more on the appropriateness of an action within a **context**, in terms of the nature of the **consequences,** which arise from it.

In this sub-section we present an approach to the management of input errors which reflects these observations.

The idea of a **responsible interface style** began to emerge during development of the PHARE PD1 interface and has been carried further through the REFGHMI [24]. The following discussion re-expresses and extends the arguments produced there.

Characteristics of responsible interfaces

The use of the term **responsible interface style** implies:

- an interface that encourages the user to give consideration to inputs which might have an operational significance *before* they are made;
- at the same time the interface must be made as easy to use and as 'direct' as possible.

Initially, it might seem strange to suggest that anyone should create an interface with a **non-responsible style**. However, upon reflection, it is reasonable to expect that, in circumstances where:

a) the consequences of input errors are severely limited, and
b) the cost of correction is very low, and
c) minimising cognitive effort from the user is given a high weighting,

there might be cases where such an interface is suitable.

Counter examples

A good example is to be found with many word processors where it is accepted that typing errors and other action slips will occur with a comparatively high frequency. Nothing other than the document being prepared is normally directly affected by the input slip. In such cases there are many lines of defence. There can be on-line and 'at-save-time' spelling and grammar checkers. There can be simple last action and multiple action '*undo*' functions. Additionally, there are the normal outflow monitoring skills of the proficient user. The few actions that have a danger element, such as saving, deleting and overwriting files, can be subject to confirmation and system backups of actions.

In ATC, however, where the proportion of significant actions is much higher, where the skills of the expert user militate against a restricting interface, and where checking/confirmation loops would be too onerous, another approach must be found. In particular, we should note that as support for communication processes increases, through tools like System Supported Coordination (SYSCO, [36]) and Controller Pilot DataLink Communications (CPDLC, [53]), the effect of inputs is no longer local. An action by a controller will have an impact on the information displayed to another controller or a pilot, in a remote location. Under these circumstances the notion of an UNDO becomes potentially very complex.

We would argue that where an interface action leads to changes in the information made to another agent, an UNDO function is not an appropriate solution, (see 'Consequences: strategies for design' below, d)).

A single 'responsible' style

Further, we suggest that it is very difficult to build an interface that uses a few simple input rules; is highly consistent (thus supporting 'automatic' behaviour); but that also allows separation of areas of interaction so that some areas require 'responsible' interaction and others allow more 'relaxed' procedures to apply.

We would argue that where lax procedures, demanding little cognitive effort, are established, they would tend to generalise or spread to other areas of the interaction. We must build a system where the Controller's style and attitude to errors is also consistent and is essentially defensive. **The user must take responsibility for actions and the interface must support the user in doing this**.

This is coherent with the policy on responsibility expressed in §3.5.

Consequences: strategies for design

The problem for the ATC interface then becomes: how to ensure that the input mechanisms demand little cognitive effort but at the same time ensure that sufficient thought is given to the **operational consequences** of "hard to recover" input actions (see §5.4). The following strategies were employed in the REFGHMI and are extended in the CoRe Baseline:

a) Following ODID the principal operational decisions were placed in a single location (the callsign menu). This principle is extended in the CoRe Baseline to establish task-related divisions for information grouping.

[X.HF.G12] (functional grouping/guideline): Assignment of operational functions to menus should be based on the natural logic of the operational task as identified in consultation with the operational experts.

b) Steps were taken to limit the ease with which 'action slips' could be made. These particularly related to the possibility of action slips [52] and errors relating to poor response times (and the queuing of inputs) and to target sizes which were too small (for both operational and interface management functions, e.g. selecting position symbols, re-sizing windows).

[X.HF.R11] (error grouping/requirement): Input mechanisms shall be designed to minimise the probability of undetected action slips through, appropriate selection area sizes, acquisition response times and feedback to the user.

c) Further, the 'cursor warping' (system displacement of the cursor) to different callsign menu items, available in ODID, was removed. Currently there is no cursor warping in the interface.

[X.HF.R11-1] (error grouping/requirement): The cursor shall be displaced only under the control (or at the instigation) of the user.

d) No 'UNDO' function is provided[29]. Some items can be REDONE, inflicting a slight but identifiable workload penalty for error.

[X.HF.R12] (error recovery procedure/requirement): Recovery from user identified errors shall be by means of a REDO (i.e. the user shall repeat the same input function with the correct procedure).

[X.HF.R12-1] (error recovery procedure/requirement): In cases where a REDO is not operationally possible (e.g. when authority is no longer retained) an alternative procedure must be identified and supported by the interface.

e) No double or triple clicking (although the advantages of double clicking can be maintained and an implicit double click for ASSUME and SKIP can be found in the operationally relevant contexts).

[X.HF.R13] (error double click/requirement): Double clicking shall not be a basic input requirement on the baseline interface.

Additional measures, such as finding a direct means of indicating/coding operationally critical decisions can be considered as functionality is added.

[29] Although a case can be made for a single UNDO ASSUME where, depending on the implementation, an action slip has consequences only for the controller who made the error.

Page intentionally left blank

7. PRINCIPLES FOR VISUAL PRESENTATION

This sub-section addresses the general visual characteristics of the interface.

7.1 Visual Style – The Look

This sub-section describes the importance of having a consistent general visual look for the interface in terms of improving the quality of the interaction.

EXAMPLE: THE CORE BASELINE

The visual style of an interface is very important. A good, consistent, well-designed look can improve the interface in a number of important ways:

- by presenting information clearly and legibly,
- by reinforcing the metaphor guiding the interactions more precisely,
- by cueing and inviting actions, aiding anticipation and improving predictability,
- by simplifying the presentation and minimising visual complexity,
- by supporting a sense of quality, uniqueness and ownership.

A good visual style should combine:

- understanding of the necessary interface functionality,
- knowledge of human perceptual characteristics,
- a sense of aesthetics and an understanding of graphic design principles.

The participation of a graphic designer can improve the quality and performance of an interface significantly. The work of the PII Division of CENA provides some good examples of this [1].

7.2 Fonts, Sizes and Viewing Distances

This sub-section addresses the physical presentation of alphanumeric information. It describes the basis for choosing suitable typefaces and fonts for the presentation of such information. Font sizes are described in terms of visual angle, the calculation of which is explained in terms of viewing distance. Legibility and figure ground contrast are identified as issues and some references to relevant literature are provided.

EXAMPLE: THE CORE BASELINE[30]

[X.HF.R14] (Legibility/requirement): All alphanumeric information presented on the controller interface shall be readily legible by the normal population with corrected vision.

In order that this should be achievable, all alphanumerics should have adequate size, be correctly designed and not liable to confusion, and be presented with sufficient contrast.

The current requirements for the primary HMI display capacity are approximately 4 megapixels of display per controller working position. This may be augmented by secondary displays.

Typically this takes the form of a square display, approx. 500 mm x 500 mm (71.1 cm diagonal = 28 inch), of 2000 x 2000 pixels, giving a pixel centre-to-centre separation of approx. 0.25mm.

Operational information is both graphic and text based. The physical size of a font employed for the display of alphanumeric information is a function of the viewing distance, which can vary in turn as a function of physical workstation configuration, lighting, display sharing and display technology[31].

By defining character size in terms of the visual angle it subtends at the eye, viewing distance can be incorporated into the font specification

EQ1 **$\tan\theta = s/d$** where **θ** is the visual angle in degrees, min. and sec.,
s is the character height in mm,
d is the viewing distance normal to the screen.

[X.HF.R14-1] (Font size/requirement): The visual angle (θ) for the smallest alphanumeric characters on the display shall be between 20 and 22 minutes of arc.

[30] Please note that there may be locally applicable health and safety standards which have to be applied for font legibility, size and contrast.

[31] Emissive Cathode Ray Tube technology has recommended viewing distances based on health and safety considerations. Consult locally applicable health and safety standards.

This gives a character height (s) of between 3.5 and 4.0 mm at a viewing distance (d) of 630 mm. This corresponds to character about 13 – 16 pixels high[32] on a typical Raster Scan or Liquid Crystal Display.

Larger fonts are almost invariably appropriate, especially under conditions where a text string may be the target of a pointing device as part of a direct manipulation input sequence[33].

Ideally a range of highly legible fonts will have to be available in a variety of screen sizes.[34].

[X.HF.G13] (font choice/guideline): CoRe currently employs a 15pt ORLY[35] font for aircraft radar labels and list text

Fuller details on typeface design and terminology and the factors to be considered in the choice of screen fonts for ATM is provided in another CoRe deliverable [41].

With monochrome cursive displays, ICAO Guidelines suggested that the contrast between character stroke and background should be 10:1 for text that has to be read [26]. A ratio of 3:1 could be acceptable for other map information.

With the introduction of colour raster scan displays, colour contrast and brightness contrast are combined. However **absolute** contrast can still be determined in terms of energy.

NOTE: There are no universally accepted standards for the level of contrast on large raster displays. However, [41] suggests that a figure:ground ratio of 3:1 is an absolute minimum. In any case, like Cardosi and Hannon, we would recommend that *'a thorough test of legibility and operational suitability in all anticipated lighting conditions is imperative'.* ([45], p8). Consequently:

[X.HF.R15] (Contrast/requirement): For Text to be read, the absolute minimum figure to ground contrast shall be at least 3:1. Target levels should be closer to 7:1, BUT in any case legibility must be confirmed in the context of application.

32 Readability is best when character heights presenting visual angles of between 20 and 22 arc minutes are employed. This also assumes a height to width ratio of between 1: 0.7 and 1: 0.9.

33 The admonitions against using text greater in size than 24 arc minutes are based on the need for more eye movements per unit of text when READING larger characters. If, however, the messages are short enough to be read in single fixations, the legibility advantage of larger text, and more especially the more rapid acquisition of a larger target during visual search, will more than offset any such effects.

34 To avoid confusion between characters, it is recommended that, when assessing the fonts for use in the interface, particular attention should be paid to the confusability of uppercase 'B', '8' and 'P' and to the confusability of 'S' and '5'. Further, in choosing sans-serif fonts, care should be taken with the readability of '1'(one), 'l' (el) 'I' (aye) and 'L', especially when they are placed close to other characters such as 'm', 'n', 'u' 'w' and 'h' with multiple parallel vertical segments. In examining proportional fonts, care should be taken with how both upper and lower case 'C' interact with a following vertical character to give the impression of 'O' or 'd'.

35 The ORLY font is one of the few which has been specially developed for ATM applications and for use on emissive displays. It has been designed by Jean-Luc Vinot, a graphics designer working on interface development at CENA near Paris. In addition to alpha-numerics, it includes an extensive set of ATM symbols. Details may be found on the CENA Web site [1].

NOTE: A fixed space font can be useful in selectable alphanumeric fields in aircraft labels, where the area of the selectable field is a function of the string size. It is also beneficial in the structuring of any list material.

7.3 General Presentation of Infrastructure Objects

This sub-section describes the conventions adopted for the presentation of the basic window elements of the CoRe Baseline.

EXAMPLE: THE CORE BASELINE

For convenience, we refer to the body of the support objects, windows, menus, dialog panels etc as 'furniture'. The look and feel of the support furniture is intended to separate it naturally from the information more directly related to the application.

One mechanism for achieving this is through the use of colour. A particular sub-palette is employed for the furniture elements, window frame, sliders, buttons, etc. The Core baseline follows the PHARE HMIs in using a Blue and Fawn look for furniture and buttons.

More specifically, the colours employed for the furniture are based on the NATS/RCA Palette for Low Foreground Items L3 [33]. (NOTE: The RCA Low Foreground Blue (RGB% 44, 55, 66), perceived by the users as 'too blue' for this particular purpose, has been changed to a more greyish colour named Window colour (RGB% 44, 51, 57). A depth impression has been created as described in §7.4 below. An additional shade (RGB% 64, 71, 77) has been added to provide highlight edges and dark edges, any other visual structure being provided by the existing RCA Grey Lines colour named Shadow colour (RGB% 20, 20, 20). (See ANNEX 4.)

7.4 Lighting Models

A lighting model is used to provide an impression of relief and texture on the graphical interface through a process of 'highlighting' and 'shadowing' the edges of the objects in a systematic way. The visual conventions used for the CoRe Baseline are described.

EXAMPLE: THE CORE BASELINE

To further differentiate them from operational data, the elements making up the frames and mechanisms of the windows (furniture) are given a slight impression of depth (broadly in keeping with the Motif guidelines [7]). This is achieved by creating an impression of illumination from a source at an infinite distance beyond the top left hand corner of the displays. The impression will be produced by giving all rectangular surfaces, which are to be perceived as 'proud', brighter left and top edges and darker right and bottom edges. Objects which are to be seen as 'indented' will be infilled with a darker colour with similar chromatic characteristics to the proud levels and the rules for edges will be reversed, i.e. top and left edges will be darker and bottom and right will be brighter.

Precise details of colours are provided in ANNEX 4.

7.5 Principles for the Presentation of Data

The largest sub-section deals with the principles and general rules for the presentation of alphanumeric information. It interprets the ODID minimum information principle in terms of requirements and stresses the importance of providing the controller with tools for managing the very large amount of information potentially available in this type of interface. It provides guidelines and requirements for the use of colour in the interface and introduces the potential use of transparency.

EXAMPLE: THE CORE BASELINE

Minimum information, maximum access

In the course of the ODID studies, the ODID group established a number of display principles [19]. Several of these have proved durable, the most important being the ODID Minimum Information principle. Simply this states that:

[X.HF.R16] (minimum information/requirement): The interface shall not directly display information which is considered by the controller to be no longer necessary.

However,

[X.HF.R17] (information access/requirement): The interface shall allow the controller to access all relevant information with an amount of effort inversely related to its immediate operational significance (see also § 6.4).

Access effort is used in visual search and processing, as well as in interaction with objects and working down through levels of menu. In this context

[X.HF.R18] (information flashing/requirement): Flashing can be used to draw the eye to a region of important information but shall not be an attribute of text which has to be processed.

applies because of the difficulty in reading flashing text.

The application of these rules is general but is most evident in the management of the radar label and its various forms. This is described more fully in the CORE Baseline HMI Functional Model itself.

Basic information management

We have already made reference to the fact that in complex operational systems it is not possible for the designer to anticipate all the ways in which the expert user will need to organise and manage the information for the interface. Nevertheless, one of the objectives of our interfaces is to minimise the amount of effort that the user has to spend in performing routine tasks of information organisation and management.

[X.HF.G14] (housekeeping/guideline): The interface design should seek to minimise the amount of effort the user has to expend on routine tasks of accessing, structuring and removing information.

Currently a great deal of the effort in the design of controller tools has concentrated on the development of tools which will support particular operational tasks, such as conflict detection, deviation from flight plan, etc. The reasoning in this sub-section, when coupled with the requirements in §5.5, gives rise to the notion of developing specific support for the management of the information on the interface; empowering the controller by providing simple general purpose mechanisms[36].

Colour and its use

The use of colour in the CoRe Baseline is the result of evolution over a large number of studies most notably, ODID, PHARE and EATCHIP. It also draws on a number of sets of guidelines (most notably ODID [19] and NATS [33]). A detailed description of how these two approaches were integrated, particularly in terms of the management of radar labels, is included as ANNEX 3. A number of other relevant standards exist, e.g. the PATRICIA Palette developed for the French PHIDIAS system upgrade [55], and the work of Cardosi and Hannon for the FAA [45], but generally there is reasonable consensus on the main principles.

These key guidelines can be summarised as follows (all have been considered in the palettes incorporated for the current baseline interface which can be treated collectively as a guideline):

1. Colour can be used for [19]:
   - indications of particular significance for the controller,
   - separation of information from the overall background,
   - to associate information which is spatially separated.
2. Large blocks of saturated colours are to be avoided.
3. The opposite ends of the spectrum should not be used in proximity in a saturated form. (i.e. red and blue).
4. Provided that contrast and legibility are respected, the use of grey backgrounds between 30 and 40% saturation is useful in:
   - reducing the effect of reflections;
   - allowing both light and dark colours to be used for foreground information without causing significant flicker (which increases with the brightness of the display for any given refresh rate on raster scan devices).
5. Colours of operational significance should be identifiable in isolation (i.e. should not depend on comparison – especially relevant for yellows and greens, or blues and greens).
6. There is tolerance in the final selection of these colours provided they meet all the other requirements, especially those on the adequacy of contrast and legibility on the range of backgrounds.

[36] Some such work has already been undertaken in CoRe in addressing radar label anti-overlap [54] and in providing mechanisms for individual radar label configuration.

7. Colour should be used consistently.
8. Some codings may have to be reviewed in the light of cultural norms.

Use of transparency

Limitation on current use

One set of mechanisms for organising information which has not yet been adequately exploited relates to the use of 'transparency'. To date transparency has not been applied in many ATM interfaces, although it is addressed thoroughly in the NATS Colour Standard [33] and was employed in PHARE PD1.

The current equipment configuration at EEC does not support the use of transparency, but it is considered as an important extension.

It is the intention to incorporate transparency in the baseline as soon as it becomes technically feasible for the provision of both static and dynamic data.

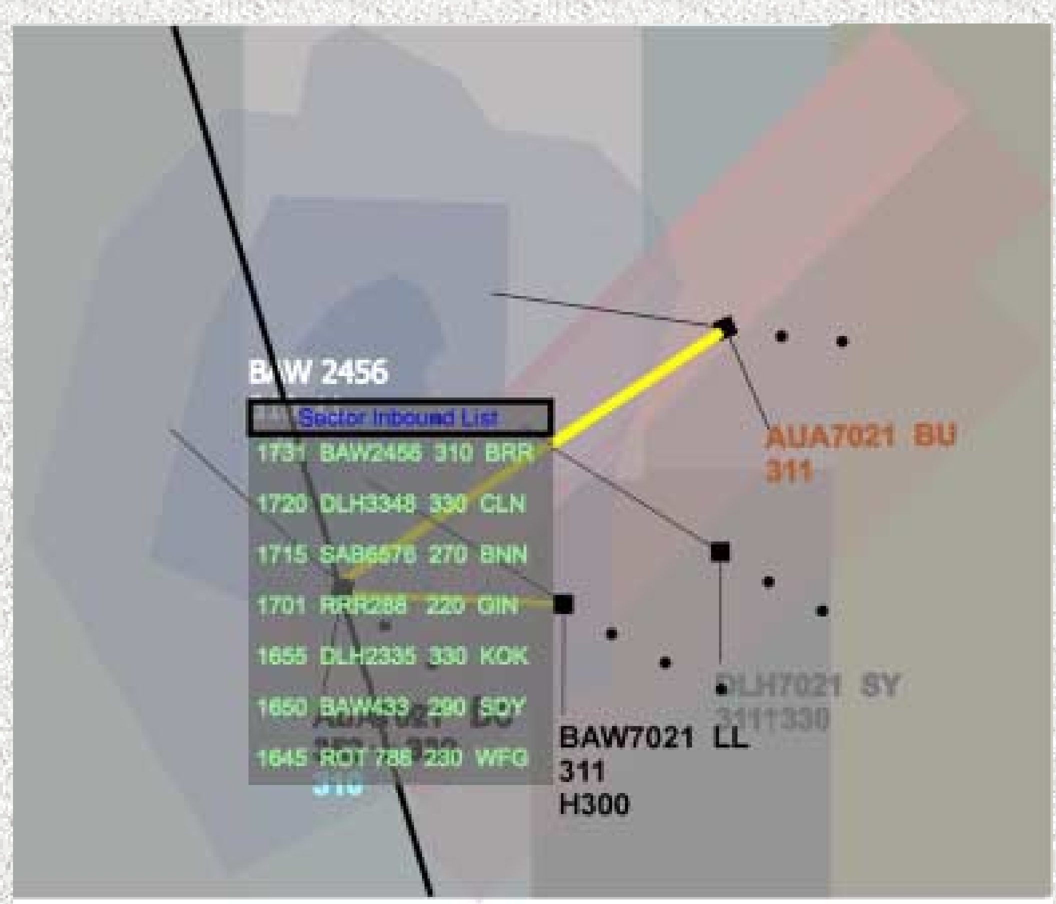


Figure 6: An example of the use of transparency on a plan view display

(Courtesy of D. Spragg)

The value of transparency

Why do we want to use transparency on displays? The answer lies in our belief that it provides a mechanism to help organise displayed information in a way that allows users more control of how they extract the information. We believe it does this by allowing users to employ perceptual control mechanisms which have developed to cope with the real visual world (especially in dealing with depth perception) to control extraction of information from the display. The fact that these perceptual mechanisms are already highly 'automatised' means that they can be employed with little apparent effort and cost. The analogy is with the looking and pointing we employ for graphical interfaces which is based on a similar exploitation of, highly tuned, 'natural' mechanisms. In the case of graphical interaction, the mechanisms in question relate to natural hand-eye coordination. In the case of transparency the mechanisms appear to relate to depth perception, colour constancy and our visual mechanisms for segregating scenes – for 'disembedding figures' in the psychological vocabulary.

When applying transparency, our assumption is that judicious use allows us to present information, which the user can selectively ignore, accessing it only when it becomes interesting; see Figure 6 for an example drawn from the work of D. Spragg.

Guidance material on the use of transparency shall be included in a subsequent version of this document.

PART 4: BASIC INTERFACE ELEMENTS

Page intentionally left blank

8. DISPLAY AND MANAGEMENT OF THE BASELINE HMI

8.1 Introduction

The available physical and electronic display area on the controller working position is constrained by:

- the physical limitations of the controller working position,
- the choice of information presentation technology,
- human characteristics, e.g.:
  - the characteristics of the human sensory motor system,
  - memory and attention,
  - awareness of context,
  - management of resources,
  - navigation within the task and interface,
  - interpersonal communications, team-working, etc.,
  - personal state, emotional arousal, fatigue, environmental conditions, etc.

These limitations lead to the set of requirements presented hereafter.

On the basis of these requirements recommendations are made:

- to employ a **graphical user interface** of the Windows, Icons, Menus, Pointers (WIMP) type;
- for technical reasons, to base the WIMP on industrial standards such as the X-Windows environment (with the reservation that special functions may be required which need additional development).

NOTE: The remainder of this section describes the CoRe baseline as it was in early 2003. Both the HMI described and the notation will continue to evolve and be documented as part of the CoRe process' application in other projects. More up-to-date versions may be available at the time of reading.

8.2 Requirements for the Display and Management of Basic Elements

Identification	Requirements description	P	Reference
[H.BL.HMI.1]	The limits of the space of interaction with the system shall be clearly defined.		[Worktop]
[H.BL.HMI.2]	The background colour and luminance of the worktop should be such as to: • provide good, homogenous, contrast for all task-related information; • minimise flicker* effects at the available screen refresh rate. *Ref. literature on flicker.		[Worktop]
[H.BL.HMI.3]	There is a need to access more information than can instantaneously be presented on the surface of the display area, which can be made 'available' to a single controller.		**[Window]**
[H.BL.HMI.4]	The general set of information that can potentially be presented mixes graphical and textual sources and does not conform to a unique hierarchy, hence: The user shall be able to navigate freely amongst the available information sources, which implies that the user shall be able to: • visualise the sources of information available (overview requirement); • select/deselect and prioritise the information sources which are available; • flexibly and easily re-organise information within the display, including the possibility of a temporary undisplay.		**[Window]** [Open Window] [Close Window] [Iconify Window] [De-Iconify Window] [Temporary Iconify Window] [End Temporary Iconify Window] [Move Window] [Come on Top Window]
[H.BL.HMI.5]	A window size need not be identical to the field of information, which can be viewed through it. This generates the need to change the viewpoint to access all the information in the field.		**[Window]** [Re-Scale Window] [Re-Size Window] [Slider-Scroll Window] [Page-Scroll Window] [Step-Scroll Window]
[H.BL.HMI.6]	The visual appearance of a window's structure should provide the controller with cues as to its behaviour and available properties.		**[Window]**
[H.BL.HMI.7]	The operational significance of the window content should always be clearly indicated.		**[Window]**
[H.BL.HMI.8]	The background colour of the window interior should be such as to provide good contrast for the displayed information.		**[Window]**

Identification	Requirements description	P	Reference
[H.BL.HMI.9]	At any moment, the locus of potential user actions shall be clearly indicated.		[Cursor]
[H.BL.HMI.10]	The type of system activity and the entry into a mode shall be clearly indicated.		[Cursor]
[H.BL.HMI.11]	At any moment, possible actions on the interface shall be clearly evident.		[Cursor]
[H.BL.HMI.12]	The occurrence of key events, which may lie outside of the user's current focus of attention, shall be clearly indicated.		[Cursor]
[H.BL.HMI.13]	The cursor shall always be visible and easily located.		[Cursor]
[H.BL.HMI.14]	The cursor movement should be smooth and track input device (e.g. mouse) movements without perceptible lags. No system activity should take priority over the cursor tracking relationship.		[Cursor]
[H.BL.HMI.15]	An easy and immediate access shall be provided at the HMI level to all the functions available to the controller, allowing him to perform actions and to access tools and objects containing information.		[Switches Buttons]
[H.BL.HMI.16]	Input mechanisms shall provide feedback specific to the action taken.		[Switches Buttons]
[H.BL.HMI.17]	The physical appearance of the input mechanisms shall provide the controller with information on their functions and characteristics.		[Switches Buttons]
[H.BL.HMI.18]	On request, the controller shall be provided with a set of selectable items amongst which he can make a selection to perform an action. Any one of the selectable items shall be easily and quickly accessed.		[Menu] [Open Menu] [Select Menu Item] [Scroll Menu]
[H.BL.HMI.19]	The availability of selections and the operational status of the presented menu items shall be clearly indicated.		[Menu]
[H.BL.HMI.20]	The currently selected menu item shall always be clearly indicated.		[Menu] General feedback principles
[H.BL.HMI.21]	The menu item that is the most probable selection, depending on the context, shall be very easily and quickly accessed.		[Menu]
[H.BL.HMI.22]	The identity of the object that initiated the display of a menu shall be clearly indicated. Depending on the context, it could be the aircraft, the particular field indication, etc.		[Menu]
[H.BL.HMI.23]	The process of making a selection shall be completed by the selection action (i.e. no supplementary action shall be necessary to close a menu).		[Close Menu]
[H.BL.HMI.24]	It shall be possible to easily and quickly abandon any initiated menu dialogue without having made a selection.		[Menu] [Cancel Menu]
[H.BL.HMI.25]	The system status shall always be clearly stated and indicated to the controller.		[System Message Window]

8.3 'WIMP' Graphical User Interface Management

8.3.1 Window Management

In this sub-section all the possible window management actions are presented. For a given window, the possibility of executing a window management action is indicated to the user by the presence of the appropriate display attribute of the window (e.g. re-sizing possible for windows with the re-size button, iconify possible for windows with the iconify button).

<table>
<tr><th colspan="3">[Open Window]: Open (display) a window</th></tr>
<tr><td>Objective</td><td colspan="2">To display a window.</td></tr>
<tr><td>Requirement Ref.</td><td colspan="2">[H.BL.HMI.4]
Rationale: Information required.</td></tr>
<tr><td>Actors</td><td colspan="2">System. Tactical Controller. Planning Controller.</td></tr>
<tr><td>Trigger</td><td colspan="2">A window can be opened:
- Either by an internal System event after the System start-up (e.g. System message display…). In this case a window is opened automatically.
- Or on Controller decision. In this case, a window is opened manually by the Controller from an invoking field (or button).</td></tr>
<tr><td>Pre-condition</td><td colspan="2">The window is not displayed.
Manual opening: the object (button) permitting invocation of the window is available.</td></tr>
<tr><td rowspan="2">Dialogue</td><td>Action</td><td>Result</td></tr>
<tr><td>1a - Occurrence of an internal System event
OR:
1b- Single Click AB on the window invoking field (button)</td><td>1a or 1b - The window is opened at its default position, or at a location defined by the currently stored preferences list. A window having been displaced during the session opens at its last position.
NB: for windows (tools) represented by a toggle button in a “parking rank” area of a Toolbox Window, Ref. the [Access Tool] procedure, Baseline HMI Functions section.</td></tr>
<tr><td>Used HMI Objects</td><td colspan="2">[Window]
[Switches Buttons]</td></tr>
<tr><td>Associated Procedures</td><td colspan="2">P: --
F: [Close Window]</td></tr>
<tr><td>Particularities</td><td colspan="2">--</td></tr>
<tr><td>Evolution</td><td colspan="2">--</td></tr>
</table>

<table>
<tr><th colspan="3">[Close Window]: Close a window</th></tr>
<tr><td>Objective</td><td colspan="2">To close a displayed window.</td></tr>
<tr><td>Requirement Ref.</td><td colspan="2">[H.BL.HMI.4]
Rationale: Information no longer required.</td></tr>
<tr><td>Actors</td><td colspan="2">System. Tactical Controller. Planning Controller.</td></tr>
<tr><td>Trigger</td><td colspan="2">A window can be closed:
- Either by an internal System event (e.g. System message delete…). In this case a window is closed automatically.
- Or by Controller decision.</td></tr>
<tr><td>Pre-condition</td><td colspan="2">The window is displayed.
The Close button of the window is available.</td></tr>
<tr><td rowspan="2">Dialogue</td><td>Action</td><td>Result</td></tr>
<tr><td>1a - Occurrence of an internal System event
OR:
1b- Single Click AB on the Close button of the window</td><td>1a or 1b - The window is closed.

NB: for windows (tools) represented by a toggle button in a “parking rank” area of a Toolbox Window, Ref. the [Replace Tool] procedure, Baseline HMI Functions section.</td></tr>
<tr><td>Used HMI Objects</td><td colspan="2">[Window]
[Switches Buttons]</td></tr>
<tr><td>Associated Procedures</td><td colspan="2">P: [Open Window]
F: --</td></tr>
<tr><td>Particularities</td><td colspan="2">--</td></tr>
<tr><td>Evolution</td><td colspan="2">--</td></tr>
</table>

[Iconify Window]: Iconify a window		
Objective	To modify the displayed aspect of a window from a full display to a small pictorial representation (icon).	
Requirement Ref.	[H.BL.HMI.4] **Rationale**: To manage information load, to save space but allowing restoration of the information at a later time (medium term).	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	The Iconify button of the window is available.	
Dialogue	**Action**	**Result**
	1 - Single Click AB on the Iconify button of the window	1 - The window is iconified. The icon is placed at its previous position or, if the window is being iconified for the first time, so that the position of the top left-hand corner of the icon coincides with the top left-hand corner of the window at the time it was iconified.
Used HMI Objects	**[Window]**	
Associated Procedures	P: [Open Window] F: [De-Iconify Window]	
Particularities	The icon of a window retains the same screen priority as the window that it represents.	
Evolution	--	

[De-Iconify Window]: De-Iconify a window		
Objective	To restore the original full display of the iconified window.	
Requirement Ref.	[H.BL.HMI.4] **Rationale**: To restore temporarily concealed information.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	The icon of the window is available.	
Dialogue	**Action**	**Result**
	1 – Single Click AB on the icon of the window	1 - The window is displayed with the same attributes (size, position…) as when it was iconified.
Used HMI Objects	**[Window]**	
Associated Procedures	P: [Iconify Window] F: --	
Particularities	--	
Evolution	--	

[Temporary Iconify Window]: Temporarily iconify a window		
Objective	To iconify a window during a period when it has no content.	
Requirement Ref.	[H.BL.HMI.4] **Rationale**: specific delegation of management of display space.	
Actors	System. Tactical Controller. Planning Controller.	
Trigger	Automatic, when the window is empty.	
Pre-condition	For the activation of the function: The Iconify button of the window is available. For the iconification: the controller has activated the temporary iconification and the window has no content.	
Dialogue	**Action**	**Result**
	1 – Single Click IB on the Iconify button of the window	1 - The Iconify button is displayed in *Button2Depressed* colour. - As soon as the window is empty: The window is iconified. The icon (in *Button2Depressed* colour) is placed at its previous position or, if the window is being iconified for the first time, so that the position of the top left-hand corner of the icon coincides with the top left-hand corner of the window at the time it was iconified.
Used HMI Objects	**[Window]**	
Associated Procedures	P: [Open Window] F: [End Temporary Iconify Window]	
Particularities	- A “temporarily” iconified window re-opens automatically with its original attributes on display of incoming information. - A “temporarily” iconified window iconifies automatically as soon as it becomes empty. - The icon of a window retains the same screen priority as the window that it represents.	
Evolution	--	

[End Temporary Iconify Window]: End the temporary iconification of a window		
Objective	To end the temporary iconification of a window.	
Requirement Ref.	[H.BL.HMI.4] **Rationale**: specific delegation of management of display space is no longer needed by the controller, who re-asserts his control.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	The controller has previously activated the temporary iconification.	
Dialogue	**Action**	**Result**
	1a - Single Click IB on the Iconify button of the window **OR** 1b - Single Click IB on the icon of the window	1 - The window is displayed at the same position as when it was iconified. - The Iconify button is displayed in its normal display colour.
Used HMI Objects	**[Window]**	
Associated Procedures	P: [Temporary Iconify Window] F: --	
Particularities	--	
Evolution	--	

<table>
<tr><th colspan="3">[Move Window]: Move a window</th></tr>
<tr><td>Objective</td><td colspan="2">To move a window.</td></tr>
<tr><td>Requirement Ref.</td><td colspan="2">[H.BL.HMI.4]
Rationale: to support controller management of display space.
The task shall require a minimum of attention, allowing controller to remain focused at Task level. Probability of error should be very low because errors will shift attention to the HMI level.</td></tr>
<tr><td>Actors</td><td colspan="2">Tactical Controller. Planning Controller.</td></tr>
<tr><td>Trigger</td><td colspan="2">Controller decision.</td></tr>
<tr><td>Pre-condition</td><td colspan="2">The window is displayed. Some part of the name bar of the window is available on the display surface.</td></tr>
<tr><td rowspan="2">Dialogue</td><td>Action</td><td>Result</td></tr>
<tr><td>1 - Press and Drag AB in the name bar part of the window

2 - Release AB</td><td>1 - The window moves in a fixed relation to the cursor to a new position on the screen (the window should be re-drawn with a speed that allows it to track the cursor position moving at the same speed as is required in normal cursor movement).
2 - The window stays at its last display position.</td></tr>
<tr><td>Used HMI Objects</td><td colspan="2">[Window]</td></tr>
<tr><td>Associated Procedures</td><td colspan="2">P: [Open Window]
F: --</td></tr>
<tr><td>Particularities</td><td colspan="2">The process of moving a window by means of the name bar moves it automatically to the top of its stack. Single Click AB on the name bar has the same effect.</td></tr>
<tr><td>Evolution</td><td colspan="2">--</td></tr>
</table>

<table>
<tr><th colspan="3">[Re-Size Window]: Re-size a window</th></tr>
<tr><td>Objective</td><td colspan="2">To change the size of a window. The scale of the window contents is not changed but the aperture of the window is changed to show an increased or decreased field of view*.

* The appropriate model is of a window looking through onto an image plane, not of a flat surface containing a fixed image that is re-sized. This model also applies to scrolling and panning where the viewpoint follows the movement of the slider.</td></tr>
<tr><td>Requirement Ref.</td><td colspan="2">[H.BL.HMI.5]
Rationale: to support controller management of display space.
The task shall require a minimum of attention, allowing controller to remain focused at Task level. Probability of error should be very low because errors will shift attention to the HMI level.</td></tr>
<tr><td>Actors</td><td colspan="2">System. Tactical Controller. Planning Controller.</td></tr>
<tr><td>Trigger</td><td colspan="2">The re-sizing of a window can be:
- Either automatic (by the system): the system dynamically re-sizes the window to allow just adequate display of varying amounts of information (e.g. the Sector Inbound List).
- Or on Controller decision. In this case, the window is re-sized manually by the Controller.</td></tr>
<tr><td>Pre-condition</td><td colspan="2">Manual re-sizing: the window Re-size feature (button) is available.</td></tr>
<tr><td rowspan="2">Dialogue</td><td>Action</td><td>Result</td></tr>
<tr><td>1 - Press and Hold AB on the Re-size button

2 - Press and Drag AB

3 - Release AB</td><td>1 - A wire-frame outline of the window is displayed; it is fixed at the top left-hand corner of the window.
2 - Dragging the wire-frame from the corner in the X and Y directions permits a two axes re-size.
3. Dragging the wire-frame outline in the X or the Y directions permits a single axis re-size.
3 - The current location of the Varying Aperture button becomes the new button position of a window re-drawn with the co-ordinates of the wire-frame. The wire-frame outline is erased.</td></tr>
<tr><td>Used HMI Objects</td><td colspan="2">[Window]</td></tr>
<tr><td>Associated Procedures</td><td colspan="2">P: [Open Window]
F: --</td></tr>
<tr><td>Particularities</td><td colspan="2">Font sizes do not change when windows are re-sized.</td></tr>
<tr><td>Evolution</td><td colspan="2">--</td></tr>
</table>

[Re-Scale Window]: Re-scale a window		
Objective	To change the scale of the content of a window. The aperture of the image window remains constant and the image is re-scaled to fill the re-sized window, i.e. the image gets bigger or smaller as the window is re-scaled.	
Requirement Ref.	[H.BL.HMI.5] **Rationale**: to support the controller's management of focus and attention.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	The window Re-scale feature (button) is available.	
Dialogue	**Action**	**Result**
	1 - Press and Hold AB on the re-scale button 2 - Press and Drag AB 3 - Release AB	1 - A wire-frame outline of the window is displayed; it is fixed at the top right hand corner of the window. 2 - Dragging the wire-frame from the corner expands/contracts it in the X/Y directions. 3 - The current location of the Re-scale button becomes the new button position of a window re-drawn with the co-ordinates of the wire-frame. The aperture of the image window remains constant and the image is re-scaled to fill the re-sized window. The wire-frame outline is erased.
Used HMI Objects	**[Window]**	
Associated Procedures	P: [Open Window] F: --	
Particularities	When a window is re-scaled to a different extent in the X and Y directions changing the aspect ratio of the window, the aspect ratio of the contents does not change. The displayed image retains the same centring and is re-scaled to maintain the X axis extent as constant.	
Evolution	--	

[Page-Scroll Window]: Scroll a window in pages		
Objective	To scroll a window in 'pages'.	
Requirement Ref.	[H.BL.HMI.5] **Rationale**: to support the controller's management of focus and attention.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	The scroll bar of the window is available.	
Dialogue	**Action**	**Result**
	1 - Single Click AB on the scroll bar body between the slider bar and upper arrow button	1 - Page scroll up
	2 - Single Click AB on the scroll bar body between the slider bar and lower arrow button	2 - Page scroll down
	3 - Single Click AB on the scroll bar body between the slider bar and left arrow button	3 - Page scroll left
	4 - Single Click AB on the scroll bar body between the slider bar and right arrow button	4 – Page scroll right
Used HMI Objects	**[Window]**	
Associated Procedures	P: [Open Window] F: --	
Particularities	The proportion of the page is defined as a parameter (default: half of the displayed page).	
Evolution	--	

<table>
<tr><th colspan="3">[Step-Scroll Window]: Scroll a window in incremental steps</th></tr>
<tr><td>Objective</td><td colspan="2">To scroll a window in incremental steps.</td></tr>
<tr><td>Requirement Ref.</td><td colspan="2">[H.BL.HMI.5]
Rationale: to support the controller’s management of focus and attention.</td></tr>
<tr><td>Actors</td><td colspan="2">Tactical Controller. Planning Controller.</td></tr>
<tr><td>Trigger</td><td colspan="2">Controller decision.</td></tr>
<tr><td>Pre-condition</td><td colspan="2">The scroll bar of the window is available.</td></tr>
<tr><td rowspan="2">Dialogue</td><td>Action</td><td>Result</td></tr>
<tr><td>1 - Single Click AB on the up arrow button
2 - Single Click AB on the down arrow button
3 - Single Click AB on the left arrow button
4 - Single Click AB on the right arrow button
5 - Press and Hold AB on any arrow button</td><td>1 - Image scroll down of an incremental step
2 - Image scroll up of an incremental step
3 - Image scroll right of an incremental step
4 - Image scroll left of an incremental step
5 - The displayed image is displaced continuously in the appropriate direction</td></tr>
<tr><td>Used HMI Objects</td><td colspan="2">[Window]</td></tr>
<tr><td>Associated Procedures</td><td colspan="2">P: [Open Window]
F: --</td></tr>
<tr><td>Particularities</td><td colspan="2">Each click results in a displacement of the displayed image in the opposite direction to the incremental pointer i.e. the pointer indicates the direction in which the window frame moves relative to the image.
The dimension of a scroll step is dependent on the content type of the window:
• The scroll step of a window displaying graphical images (such as the radar image) is a parameter (with default setting of 1/10th of the display).
• The scroll step of a window displaying textual information is 1 line of text.
The repeat rate in case of the continuous scroll (PH AB on an arrow button) has to be defined.</td></tr>
<tr><td>Evolution</td><td colspan="2">--</td></tr>
</table>

[Slider-Scroll Window]: Manually scroll a window		
Objective	To manually scroll a window.	
Requirement Ref.	[H.BL.HMI.5] **Rationale**: to support the controller's management of focus and attention.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	The scroll bar of the window is available.	
Dialogue	**Action**	**Result**
	1 - Press and Drag AB on the panning slider 2 - Release AB	1 - The slider moves accordingly. 2 - The screen image is redrawn on the same scale but with the centre of the display displaced in the same sense as the slider has been dragged, and in proportion to the displacement of the slider relative to the scale.
Used HMI Objects	**[Window]**, slider	
Associated Procedures	P: [Open Window] F: --	
Particularities	- In cases where the potential field of view is limited, the slider shall reach the end of the slider bars to correspond with the limit of possible movement of the display. - If the cursor is moved off the slider with AB still held down, the slider will continue to track the appropriate axis of cursor movement until AB is released, as if the cursor were still positioned on the slider (Retention area without limit).	
Evolution	--	

[Come on Top Window]: Move a window to foreground		
Objective	To transfer a window from its current position to the top of its stack.	
Requirement Ref.	[H.BL.HMI.4] **Rationale**: to support the controller's management of focus and attention. To support the re-prioritising of information.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	Part of the window frame is available.	
Dialogue	**Action**	**Result**
	1 - Single Click AB on the frame of the partially covered window	1 - The window moves to the top of its stack.
Used HMI Objects	**[Window]**	
Associated Procedures	P: [Open Window] F: --	
Particularities	The priority assigned to a window determines the ability to move it in front of other windows/objects: a window can only be moved in front of windows/objects of the same or of a lower priority.	
Evolution	--	

8.3.2 Menu Management

In this sub-section all the available menu management actions are presented. Menus are not iconifiable, not moveable, not user re-sizeable.

Interaction options

- Option 1: Menus are modal, i.e. interaction cannot take place anywhere else until the dialogue sequence initiated by opening the menu is completed or cancelled. This means that 2 clicks are necessary if a menu dialogue has to be cancelled: one click to cancel the menu, one other click to initiate the new interaction. The advantage of this solution is that of the 'psychological closure': an action is completed before another one can be initiated, introducing a notion of sequence and rhythm into the user's actions. For example, a menu always involves two clicks, one to open and one to close having made a selection or not. The drawback is that of being bound to make two clicks instead of one to initiate an intended action.
- Option 2: Menus are not modal, i.e. interaction can take place elsewhere, while the dialog sequence has been initiated by opening a menu. If a new interaction is initiated (by clicking on another dialogue object for example), this action **at the same time** cancels the initiated menu dialogue. The advantage of this solution is to correspond to the user's intention: 'I intend to cancel the current dialogue AND to initiate a new action', and to require only one click to achieve it.

Optional addenda to Option 2: an interaction of the type Press & Hold IB does not cancel the menu dialogue, allowing the user to have a 'quick look' at other information, for verification purposes for example, before completing the menu dialogue.

[Open Menu]: Open a menu		
Objective	To display a menu.	
Requirement Ref.	[H.BL.HMI.18] **Rationale**: to allow the controller to make choices between potential actions or parameter values in a context sensitive manner.	
Actors	Tactical Controller. Planning Controller	
Trigger	Controller decision.	
Pre-condition	The menu is not displayed.	
Dialogue	**Action**	**Result**
	1 - Single Click AB on the menu invoking field	1 - The menu is displayed at its default position.
Used HMI Objects	**[Menu]**	
Associated Procedures	P: [Select Aircraft] F: [Select Menu Item]	
Particularities	--	
Evolution	--	

[Scroll Menu]: Scroll a menu		
Objective	To scroll a menu.	
Requirement Ref.	[H.BL.HMI.18] **Rationale**: to trade-off size of menu area and the number of selections available.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	A scrollable menu is displayed.	
Dialogue	**Action**	**Result**
	Depending on the implemented scroll mechanism and the defined scrolling parameters, either only the [Step-Scroll Window] procedure (with arrow buttons) or also the [Page-Scroll Window] procedure and the [Slider-Scroll Window] procedure (with a scroll bar) will be available for scrolling a menu.	
Used HMI Objects	**[Menu]**	
Associated Procedures	P: [Open Menu] F: --	
Particularities	For the page-scroll of a menu, the default "page" parameter should be the 'number of displayed items-1' (this allows to keep one item of the previous/next 'page' on the top/bottom of the displayed list).	
Evolution	--	

[Select Menu Item]: Select an Item of a menu		
Objective	To select an option item of a menu.	
Requirement Ref.	[H.BL.HMI.18] **Rationale**: to allow the controller to make choices between potential actions or parameter values in a context sensitive manner.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	A menu is displayed.	
Dialogue	**Action**	**Result**
	1 - Single Click AB on an available option of the menu	1 - The action related to the selected item is launched or the selected parameter value is employed. The menu is removed.
Used HMI Objects	**[Menu]**	
Associated Procedures	P: [Open Menu] F: [Close Menu]	
Particularities	--	
Evolution	--	

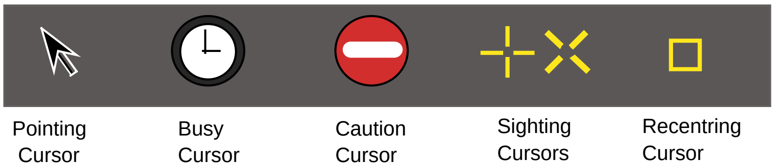
[Cancel Menu]: Cancel a menu dialogue		
Objective	To cancel an initiated menu dialogue without having made a selection.	
Requirement Ref.	[H.BL.HMI.24] **Rationale**: to support the controller's flexibility in switching focus and attention to meet task demands.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	A menu dialogue has been initiated (a menu is displayed).	
Dialogue	**Action**	**Result**
	1a - Single Click IB inside the menu region **OR:** 1b - Single Click AB or IB outside the menu region	1a or 1b - The dialogue is discarded. The menu is removed.
Used HMI Objects	**[Menu]**	
Associated Procedures	P: [Open Menu] F: --	
Particularities	Depending on the chosen Interaction Option, a Single Click AB or IB on an interactive object outside the menu initiates or not a new dialogue (see description above).	
Evolution	--	

[Close Menu]: Close (remove) a displayed menu after having made a selection		
Objective	To close (remove) a displayed menu dialogue after having made a selection.	
Requirement Ref.	[H.BL.HMI.23] **Rationale**: to make a selection, to minimise the complexity of the display and to indicate completion of the menu dialogue sequence.	
Actors	Tactical Controller. Planning Controller.	
Trigger	Controller decision.	
Pre-condition	A menu dialogue has been initiated (a menu is displayed).	
Dialogue	**Action**	**Result**
	1 - Single Click AB on an available option of the menu	1 - The action related to the selected item is launched. - The menu is removed.
Used HMI Objects	**[Menu]**	
Associated Procedures	P: [Select Menu Item] F: --	
Particularities	--	
Evolution	--	

8.4 'WIMP' Graphical User Interface Objects

[Worktop]	
Objective	The Worktop defines the limits of the space of interaction with the system. It is the background surface on which all other display elements are presented, and defines the lower limit of the range of screen priorities.
Requirement Ref.	[H.BL.HMI.1] [H.BL.HMI.2]
General Description	The worktop covers the entire addressable surface of the primary display. On the display, all the pixels of the worktop have the same RGB dynamic range. The visual presentation of the worktop is subdued in relation to other major display elements. The colour used is the *Worktop* colour.
Invocation	Always present.
Display position	Screen background (lowest interaction priority).
Illustration	No illustration.
Associated Procedures	--
Associated HMI Objects	--
Associated Conceptual Obj.	--
Particularities	--
Evolution	--

[Cursor]	
Objective	To indicate to the user the locus of interaction and possible actions on the interface.
Requirement Ref.	[H.BL.HMI.9] [H.BL.HMI.10] [H.BL.HMI.11] [H.BL.HMI.12] [H.BL.HMI.13] [H.BL.HMI.14]
General Description	All the cursor formats shall have sufficient contrast in relation to all backgrounds against which they can appear. Since the “pointing” cursor can be moved against varying backgrounds and crossing lines of different colours, the proposed options are: • a double black and white contour (**demonstrated solution**) • the outline of the cursor changes colour as a function of the current background colour. The following cursors should be provided, reflecting a number of aspects of system activity (see illustrations of possibilities below): • the Pointing cursor is the standard default cursor: an arrowhead pointing up and to the left. It is used for all pointing and window management functions. The hot spot[(1)] of this cursor is the point of the arrowhead. • the Busy cursor is used to indicate that the system is processing as a result of input. It reverts to the appropriate cursor when processing is complete. No other inputs are possible in the area while this is active. For the Clock (and the Hourglass) the hot spot is at the centre of the visible disk. The busy cursor should be dynamic (for example spinning clock hands or a spinning Hourglass). The speed of spin could effectively act as a secondary cue to the current loading on the system (on the assumption that it will slow as the system loading increases).[(2)] • Sighting cursors are used when the interaction has entered a mode. The shape is crosshairs of 10 mm in *Cursor* colour. The hot spot of this cursor type is the centre of the cross. • the Re-centring cursor is used during the offset of the display centre of the radar PVD. It is a 5 mm square in *Cursor* colour. The hot spot of this cursor is the centre of the square. • the Caution cursor is used to show that input is not currently possible in this region of the display, usually because some prior input requirement must be completed first. With a three button mouse, some inputs could be legal while others are not. Under these circumstances, the caution cursor shall only appear when the illegal button input is attempted, or when any button input would be illegal. The cursor reverts to previous cursor form when the pointer is moved from the illegal region or an attempt at an illegal input is terminated. The colour used for this cursor is the *Alert* colour. The hot spot is the centre of the disk. • In addition, other “function specific” cursors may be employed within editing modes to indicate the nature of the functions available, or to communicate other information to the operator (e.g. a change of cursor form to direct the controller’s attention to some urgent or outstanding action elsewhere on the interface). Such cursors will be defined in the specification of the associated function. [(1)] The HOT SPOT of a cursor is the pixel point within the cursor, which is returned as the value of the cursor location.

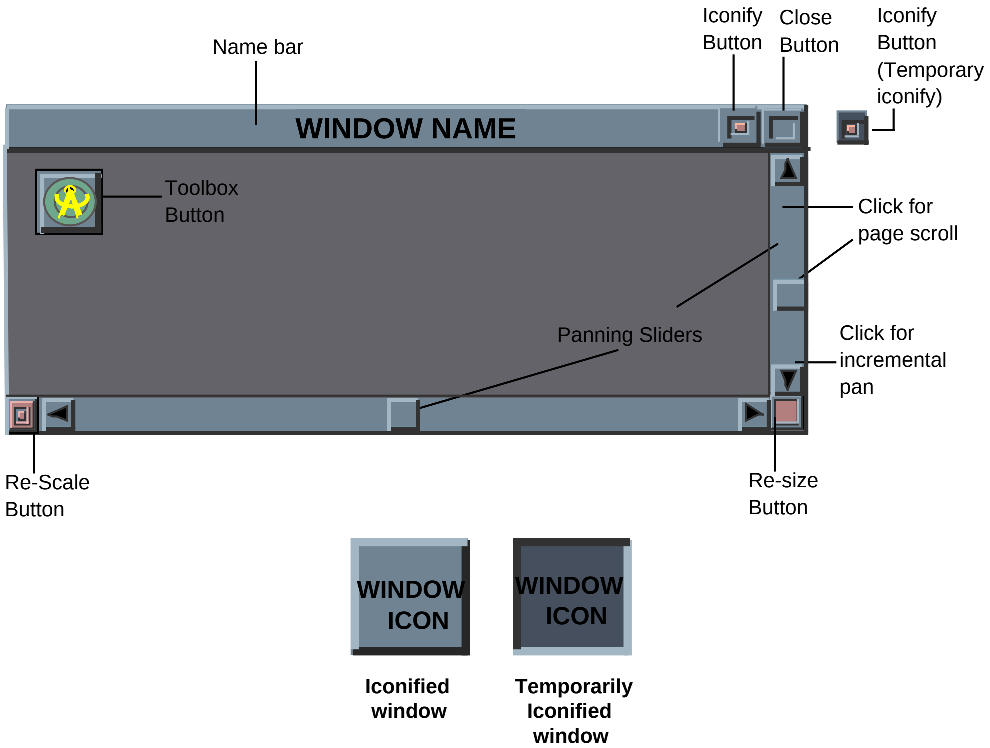
[Cursor]	
	(2) In cases where the input action requires a particularly lengthy computation, it may be appropriate to provide a specific system message, by means of a suitable mechanism, to indicate the progress of this activity. It may even be necessary to release the cursor to permit the controller to undertake other activities while awaiting the outcome of this process. Possible examples could include the running of an induced conflict probe, or other calls to tools, or the use of a datalink communication mechanism. However, this type of mechanism should be undertaken only with extreme caution.
Invocation	Always present.
Display position	Anywhere on the screen, depending on the user's action on the pointing device. Highest display priority.
Illustration	
Associated Procedures	All the user initiated procedures.
Associated HMI Objects	--
Associated Conceptual Obj.	--
Particularities	--
Evolution	--

<table>
<tr><th colspan="2">[Window]</th></tr>
<tr><td>Objective</td><td>To encapsulate different processes for the purposes of easy reorganisation of information within the display, and to provide access as needed to an organised display surface, presenting information of particular types:
o Tabular lists, Textual data
o Graphic images (e.g. maps)
o HMI tools
o Functional tools
o System messages, etc.</td></tr>
<tr><td>Requirement Ref.</td><td>[H.BL.HMI.3] [H.BL.HMI.4] [H.BL.HMI.5] [H.BL.HMI.6] [H.BL.HMI.7] [H.BL.HMI.8]</td></tr>
<tr><td>General Description</td><td>A window is defined by display attributes and its behavioural attributes. The visual appearance and behaviour of a window provide the user with cues as to its available properties. The following details all the possible features of a window (though many windows will employ only a subset of these elements):
• <u>Frame</u>:
The frame of a window is defined by the line style, the colour of the border enclosing the window and the type of corners (square or round) related to the type of the window. A window frame can include:
o <u>A header</u> (or name bar) is at least 6.0 mm thick, located along the top edge of each window frame. The number of lines of text contained in the header is specified as a parameter. The text font parameters and colour, and the background colour of the header part can be defined. The header portion of the window is not scrollable.
o <u>Scroll bars</u> (vertical and/or horizontal): if a window is defined to be scrollable, a scroll bar is displayed automatically along the bottom edge and/or the right hand edge of the window frame when the content of the window exceeds horizontally or vertically the displayed page. A scroll bar consists of 3 parts:
[1] the scroll bar body.
[2] the panning sliders: they are fixed size if the range of the possible display is not limited (e.g. radar window), and variable size if there is a fixed range of display possible, the length of the slider displaying the proportion of the range currently visible.
[3] Left / Right and Down / Up scrolling arrow buttons: they are drawn in full colour if scrolling is possible in the direction indicated. If not (no further items available for example), the colour of the arrow is modified so as to effectively reduce the contrast of the arrow area.
The display features of the scroll bar elements are identical for all the windows.
o <u>Function buttons</u>: a window frame can include the following buttons:</td></tr>
</table>

[Window]	
	[4] Iconify button: an iconified window ([Iconify Window] procedure) is represented by a small icon, bearing a graphical representation or text identifying its content. In case of Temporary iconification ([Temporary Iconify Window] procedure), the colour of the Iconify button and of the icon of the window is different (Button2Depressed) from that of a simply iconified window, to indicate the fact to the user (see illustration). See description of individual windows for each particular case. [5] Close button [6] Re-size button [7] Re-scale button Different visual forms and positions in the window frame should be employed for the function buttons. A suggestion is shown in the illustration below. • Background colour: The background colour of a window is a single colour displayed as the background for the entire window. Transparency should be an option for the background colour when the cursor is placed outside the window. An appropriately selected transparent background would allow the information placed underneath the window to be visible (although not necessarily legible), while allowing the content of the window itself to remain legible. • Size (default, minimum, maximum): The window sizes are defined in terms of X and Y axes, and are expressed in pixels or characters. Default size relates to the window size at simulation initiation and may vary according to data posting during the course of a simulation and preferences stored in the user preferences set. NOTE: All the window sizes given in this document are the **internal window area** sizes. • Associated Toolbox:(1) It should be possible to associate a toolbox with each window, whose contents (tools) are matched to the specific requirements of that window. This toolbox shall be movable only within its parent window. Where similar tools are required in different windows, they should be identical in form and operation. • Status: Parent / Child / Independent: Each window possesses one of these statuses. Child windows are employed within the context of interaction with their ‘parent’ window, and have the following characteristics: ∘ a child window changes its priority with its parent window; ∘ a child window cannot be moved outside its parent window; ∘ a child window can never be positioned behind its parent window; ∘ a child window can be moved in front of another family windows stack. Independent windows do not have child windows. • Window priority:

<table>
<tr><th colspan="2">[Window]</th></tr>
<tr><td></td><td>Windows may overlap (i.e. the general principle of window organisation is stacking).
A windows screen stacking priority is defined which determines the overlapping of their display representations.
Windows are initially displayed in accordance with their priorities. On display, a window with a given priority is displayed in front of objects or windows having the same or lower priority.
The proposed priorities are as follows:
◦ Cursor: priority 1 (highest priority),
◦ System Message Window: priority 2,
◦ Alert Window: priority 3,
◦ General Toolbox Window (and Clock): priority 4,
◦ Radar Plan View Display Window: priority 6 (so that interacting with the radar display will NOT cause it to be redrawn in front of the other displayed windows),
◦ All the other windows: priority 5.
• <u>Visual Feedback on window management</u>:
Immediate feedback of move and re-size window management functions is provided.

(1) The term ‘toolbox’ is used to describe the area carrying tools. The term ‘toolset’ is used to refer to a particular set of tools defined for use in a particular context.</td></tr>
<tr><td>Invocation</td><td>The display of a window is initiated either automatically by the system, or manually by the controller by designation of objects or invoking fields (Ref. the description of the [Open Window] and [De-Iconify Window] procedures).
When invoked, windows are displayed in accordance with their display priorities.</td></tr>
<tr><td>Display position</td><td>The window display position is defined in terms of X/Y of top left-hand corner, and is expressed in pixels or characters.
Default window position relates to the position of the window at simulation initiation, and may vary according to data posting during the course of a simulation, and preferences stored in the user preferences set.</td></tr>
</table>

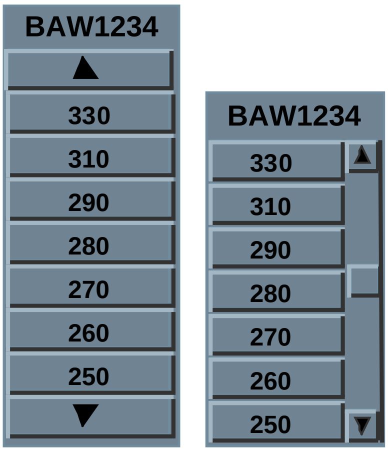
<table>
<tr><th colspan="2">[Window]</th></tr>
<tr><td>Illustration</td><td>
</td></tr>
<tr><td>Associated Procedures</td><td>[Open Window] [Close Window] [Iconify Window] [De-Iconify Window] [Temporary Iconify Window] [End Temporary Iconify Window] [Move Window] [Re-Scale Window] [Re-Size Window] [Slider-Scroll Window] [Page-Scroll Window] [Step-Scroll Window] [Come on Top Window]</td></tr>
<tr><td>Associated HMI Objects</td><td>[Worktop] [Switches Buttons]
Any object calling or being called by a window.</td></tr>
<tr><td>Associated Conceptual Obj.</td><td>--</td></tr>
<tr><td>Particularities</td><td>--</td></tr>
<tr><td>Evolution</td><td>--</td></tr>
</table>

<table>
<tr><th colspan="2">[Menu]</th></tr>
<tr><td>Objective</td><td>A menu is a dialogue object by which the user selects one or more inputs from within a range of possible inputs.</td></tr>
<tr><td>Requirement Ref.</td><td>[H.BL.HMI.18] [H.BL.HMI.19] [H.BL.HMI.20] [H.BL.HMI.21] [H.BL.HMI.22] [H.BL.HMI.24]</td></tr>
<tr><td>General Description</td><td>A menu appears as an area on the display surface, presenting a set of selectable items. The physical appearance of a menu shall be characterised by a set of display attributes:

<u>Frame</u>:
A menu frame can include :

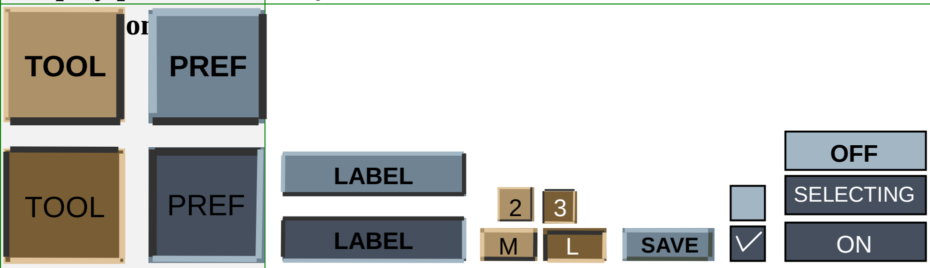
<u>A header</u> (or name bar) located along the top edge of each menu frame. The number of lines of text contained in the header is specified as a parameter. The text font parameters and colour, and the background colour of the header part can be defined. The header portion of the menu is not scrollable. A menu related to a particular aircraft always displays the identity (callsign) of that aircraft in the header.
<u>A scrolling mechanism</u>: if a menu is defined to be scrollable, it contains scrolling features which can be of different types:
[8] a scroll bar of the same type as that used for windows, situated to the right of the menu items,
[9] arrow buttons situated above and below the menu items (option shown in all the illustrations of these specifications),
[10] or any other scrolling mechanism.

<u>Background colour</u>:
The background colour of a menu is a single colour displayed as the background for the entire menu.
<u>Number of menu items</u>:
The number of displayed items is defined for each menu. These determine the size of the menu.
<u>Menu items presentation</u>:
A default item is defined and indicated in each menu. Several default items can be defined, depending on the current context when opening the menu.
The text font parameters of the menu items, colour of the available and unavailable items, and indication of the currently selected item can be defined.
The colour of the available items, which lie within Letters of Agreements (LoA) can also be defined.
</td></tr>
<tr><td>Invocation</td><td>The display of a menu is initiated by designation of fields or objects, called “initiating objects”.</td></tr>
<tr><td>Display position</td><td>The display position of the menu is defined relative to the invoking field or to the cursor position. Unless otherwise specified, when invoked, a menu is displayed so that the default value is under the cursor (to minimise the need for cursor movement).</td></tr>
</table>

[Menu]	
Illustration	 Illustration of a Flight Level menu with two different scrolling mechanisms.
Associated Procedures	[Open Menu] [Scroll Menu] [Select Menu Item] [Cancel Menu] [Close Menu]
Associated HMI Objects	[Aircraft Track] [Extended Radar Label] [Radar Label] [Sector Inbound List Window]
Associated Conceptual Obj.	--
Particularities	--
Evolution	--

[Switches Buttons]	
Objective	The function of the different switches and buttons is to provide access to tools or to a set of options amongst which the controller can make a selection in order to access information or to perform an action.
Requirement Ref.	[H.BL.HMI.15] [H.BL.HMI.16] [H.BL.HMI.17]
General Description	The buttons have different formats and colours providing the controller with information on their functions and characteristics (see illustration of the different options below): • Large buttons: give access to main elements of the HMI (windows, tools, menus…). ○ *Button1* colour(a): give access to main windows or tools. ○ *Button2* colour (b): give access to windows or menus in which additional choices should be made. The ON or selected state of both buttons is shown by *Button1Depressed* colour or *Button2Depressed* colour. • Medium buttons: give access to child windows or menus ○ *Button2* colour (c): give access to child windows or menus in which additional choices should be made, and initiate processing. The ON or selected state of these buttons is shown by *Button2Depressed* colour. • Small buttons (minimum 5mm x 5mm): ○ *Button1* colour (d): Radio[(1)] buttons for choosing between a range of options. The ON or selected state of Radio buttons is shown by *Button1Depressed* Colour and the button label in *TextSelected* colour. ○ *Button2* colour (e): Toggle and push buttons for operating and monitoring functions. The ON or selected state of these buttons is shown by *Button2Depressed* colour. ○ Switch buttons (f) are used for selecting values on parameters. The OFF or unselected form of Switch buttons is always in *Button2Highlight* colour. The ON or selected state is shown by *Button2Depressed* colour with a tick mark '✓' in *TextSelected* colour inside. Clicking AB on any Switch button toggles its state. ○ In addition to the standard Switch buttons, there are a few special Switch buttons (g), which are employed to manage and provide feedback for functions with a series of states. These buttons are in the same colour as the conventional Switch buttons, but are distinguished by their form and size in that they are rectangular, 5mm in height and up to 30mm in length, allowing the presentation of single word feedback on the state of processes. The word is presented in *TextUnselected* colour in OFF (unselected) state, and in *TextSelected* colour in ON (selected) state. (1) Switch buttons are distinguished from Radio buttons in that they can all be set

[Switches Buttons]	
	independently. Only one Radio button can be selected at a time.
Invocation	Not applicable.
Display position	Anywhere on the screen.



(a) (b) (c) (d) (e) (f) (g)

Associated Procedures	Any procedure involving an action on one of these buttons.
Associated HMI Objects	[Window] [Menu] Any object using any of these buttons.
Associated Conceptual Obj.	--
Particularities	--
Evolution	--

Template table summarising the display and behavioural attributes of Windows and Menus

The following table will be used to present a synopsis of the display and behavioural attributes of each defined window:

Window display attributes			<Window name>
Frame	Border	Line style	
		Line colour	
		Square / Round corners	
	Header	Number of lines	
		Background colour	
		Header text	
		Text colour	
		Text fonts parameters	
Background colour			
Default size			
Minimum size			
Maximum size			
Default position			
Priority on invocation			
Associated Toolbox			
Status (parent / child / independent)			
Window behavioural attributes			
Move			
Re-size			
Re-scale			
Iconify / De-iconify			
Temporary iconify			
Close			
Open			
Scroll / Pan (page, incremental, manual)			

The following table will be used to present a synopsis of the display and behavioural attributes of each defined menu:

Menu attributes		<Menu name>
Header	Number of lines	
	Background colour	
	Header text	
	Text colour	
	Text fonts parameters	
Menu items	Available item text colour	
	Unavailable item text colour	
	Text fonts parameters	
	Unselected item background colour	
	Selected item background colour	
	LoA unselected item background colour	
	LoA selected item background colour	
	Number of items	
Display position		
Scroll		
Initiating objects		

Page intentionally left blank

PART 5: BIBLIOGRAPHY, TERMINOLOGY AND CONTRIBUTORS

Page intentionally left blank

REFERENCES

[1] **CENA** PII Division Projects http://www.tls.cena.fr/divisions/PII/.

[2] **Cooke, M. and Marti, P. (2002).** *CoRe Project Document: Style Guides for ATM Development*, also available as EEC Note.

[3] **Pasquini, A. (2002).** *CoRe Project Document: Capture and Exploitation of Requirements*, also available as EEC Note.

[4] **Pierret, O. and Rousseau, J-M., (2002).** *CoRe Project Document: Recommendations for HMI Evaluation in the context of CWP Development - A Synthesis of Relevant Literature*, also available as EEC Note.

[5] **Jackson A., Geissel, M. and Dorbes, A. (2001).** Supporting the collective process of controller working position development for ATM in: *Proceedings of the 20th Digital Avionics Systems Conference (DASC)*, Daytona, October 2001.

[6] **Dorbes, A., Geissel, M. and Jackson A. (2001).** Integrating multiple viewpoints in the coherent design of advanced controller working positions in: *Proceedings of the 20th Digital Avionics Systems Conference (DASC)*, Daytona, October 2001.

[7] **Open Software Foundation (1991).** OSF/Motif Style Guide, Release 1.1, Prentice-Hall: Open Software Foundation.
A more up-to-date electronic version of the style guide can be accessed at http://w3.pppl.gov/misc/motif/MotifStyleGuide/en_US/TOC.html.

[8] **Apple Style Guide (original).** Apple Computer Inc.

[9] **Macintosh Human Interface Guidelines (2002)** http://developer.apple.com/techpubs/mac/HIGuidelines/HIGuidelines-2.html.

[10] **IBM (2002)** User Rights: IBM Site: http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/12.

[11] **Usability Net: Standards:** http://www.hostserver150.com/usabilit/tools/r_international.htm.

[12] **Gale, S. (1996).** A Collaborative approach to Developing Style Guides in: *Proceedings of ACM CHI Conference on Human Factors in Computing Systems*, Vancouver, Canada, April 13-18 (pp. 362-367) ACM Press.

[13] **ISO13407 (1999).** *Human-centred Design Processes for Interactive Systems*.

[14] **Billings, C.E. (1991).** Human-centred Automation: A Concept and Guidelines, *NASA Technical Memorandum 103885*.

[15] **Norman, D.A. and Draper, S. (1986).** *User Centered System Design,* Hillsdale N.J.: Lawrence Erlbaum Associates.

[16] **Nijhuis, H.B. (1998).** Automation Strategies: Evaluation of Some Concepts in Integrating Human Factors into the life cycle of ATM Systems, in: *Proceedings of the 3rd EUROCONTROL Human Factors Workshop,* (Luxembourg, 7-9 October 1998), Brussels: EUROCONTROL.

[17] **Dorbes, A. and Jackson, A. (1996).** *SA2 Phase 1 and 2 Facility Specification*, Version 1.0, Brétigny, France: EEC.

[18] **EUROCONTROL (1993).** *ODID IV Facility Specification*, Version A (Technical), September, Brétigny, France: EEC.

[19] **EATCHIP (1993).** *ODID (Operational Display and Input Development) Report and Guidelines*. EUROCONTROL.

[20] **EATMP (1999a).** *ATM Human Resources Programme: Programme Management Plan,* Edition 1.0. Brussels: EUROCONTROL.

[21] **EATMP Human Resources Team (2002).** *CoRe Requirements for ATM Working Positions: An Overview of the Project Activity.* HRS/HSP-006-REP-03. Edition 1.0. Released Issue. Brussels: EUROCONTROL.

[22] **EUROCONTROL (1991a).** *Common Operational Performance Specifications (COPS) for the Controller Working Position*, Version 6-91/1, Brussels: EUROCONTROL.

[23] **EUROCONTROL (1991b).** The ODID III Real-Time Simulation, *EUROCONTROL Experimental Centre Report N°242*, September, Brétigny, France: EEC.

[24] **EUROCONTROL (1995).** A Human-Machine Interface Reference System for En Route Air Traffic Control, *EUROCONTROL Experimental Centre Report N°292*, December, Brétigny, France: EEC.

[25] **EUROCONTROL PHARE DEMONSTRATORS** http://www.eurocontrol.be/projects/eatchip/phare/documentation/demonstrations/demonstrations.htm.

[26] **ICAO (1993).** *Human Factors Digest N°8*, Human Factors in Air Traffic Control, International Civil Aviation Organization Circular 241-AN/145, Montreal, Canada: ICAO.

[27] **ICAO (1994).** *Human Factors Digest N°11*, Human Factors in CNS/ATM systems. The development of human-centred automation and advanced

technology in future aviation systems, International Civil Aviation Organization Circular 249-AN/149, Montreal, Canada: ICAO.

[28] **EUROCONTROL PHARE (1993).** Role of Man Within PHARE*, PHARE Document 93-70-35*, Prepared by PHARE Working Group ‘Role of Man’, June, EUROCONTROL.

[29] **Hutchins, E. (1995).** *Cognition in the Wild,* Cambridge, MA: MIT Press.

[30] **Shiffrin, R.M. and Schneider, W. (1977).** Controlled and automatic human information processing: II Perceptual Learning, automatic attending, and a general theory. *Psychological Review 84*, pp. 127-190.

[31] **Ericsson K.A. and Simon, H.A. (1984)**. *Protocol Analysis: Verbal Reports as Data*, Cambridge, MA: MIT Press.

[32] **EUROCONTROL (2000).** *IMPACT: Methods and Tools to assess the impact of future ATM Systems on the controller’s job*, FINAL Report, Interactive CD, December. Brétigny, France: EUROCONTROL Experimental Centre.

[33] **Reynolds, L. and Metcalfe, C. (1992).** The Interim NATS Standard for the Use of Colour on Air Traffic Control Displays*. CS Report 9213*. Issue 1.2.

[34] **Neisser, U. (1976).** *Cognition and Reality: Principles and implications of cognitive psychology.* San Francisco: W.H. Freeman & Company.

[35] **Norman, D.A. (1986).** Cognitive Engineering in: D.A. Norman & S. Draper (eds.). *User Centered System Design,* N.J.: Lawrence Erlbaum Associates.

[36] **EATCHIP (1996b).** *EUROCONTROL Standard for On-Line Data Interchange*, Version 2.0. Brussels: EUROCONTROL.

[37] **Helander, M. (1988).** *Handbook of Human-Computer Interaction,* North Holland Elsevier, New York.

[38] **Hutchins, E.L., Hollan, J.D. and Norman D.A. (1986).** Direct Manipulation Interfaces in: D.A. Norman & S. Draper (eds.). *User Centered System Design,* N.J.: Lawrence Erlbaum Associates.

[39] **Ziegler, J.E. and Fähnrich, K.-P. (1988).** Direct Manipulation in: Helander, Martin (ed). *Handbook of Human-Computer Interaction,* North Holland.

[40] **Wilson, C.E. (2001).** Guidance on Style Guides: Lessons Learned, *Usability Interface*, Vol. 7, No 4.

[41] **EATMP Human Resources Team (2000b).** *Font Requirements for Next Generation Air Traffic Management Systems*, accompanied by a

Technical Appendix, *An Experimental Methodology for Selecting Fonts for Next Generation Air Traffic Management Systems*. HRS/HSP-006-REP-01 and REP-02, Editions 1.0. Released Issues. Brussels: EUROCONTROL.

[42] **Menuet, L. and Stevns, P. (1999).** *Denmark-Sweden Interface (DSI) Human-Machine Interface Specification: ACC/APP,* Version 1.2, May, Brétigny, France: EUROCONTROL Experimental Centre.

[43] **Mendoza-Navas, M. and Canu, V (2001).** *Italian Interface (ITI) User Requirements Document,* Version 1.1, Brétigny, France: EUROCONTROL Experimental Centre.

[44] **Connolly, D.W., Spanier, G. and Champion F. (1975).** Color Display Evaluation for Air Traffic Control, *FAA Report N° FAA-RD-75-39*.

[45] **Cardosi, K. and Hannon, D. (1999).** Guidelines for the Use of Color in ATC Displays, DOT/FAA/AR99/52.

[46] **Narborough-Hall, C.S. (1985).** Recommendations for applying colour coding to air traffic control displays*. Displays*, pp. 131-137.

[47] **David, H. and Roelofsen, J. (1986).** Use of Colour in Electronic Data Displays (Second Study). Task AT02a, WPDG 2/86/OS, EUROCONTROL Experimental Centre *Report N°199*, Brétigny, France: EEC.

[48] **Tognazzini, B. (1992).** *Tog on Interface*, Reading, MA: Addison-Wesley Publishing Company.

[49] **Boff K.R. and Lincoln, J.E. (1988).** *Engineering Data Compendium.* Human Engineering Division, Harry G. Armstrong Aerospace Medical Research Laboratory, Wright-Paterson Air Force Base, OH 45433.

[50] **Welford, A.T. (1968).** *Fundamentals of Skill,* London: Methuen.

[51] **Card, S.K., Moran, T.P. and Newell, A. (1983).** *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates.

[52] **Norman, D.A. (1981).** Categorization of Action Slips*. Psychological Review*, 88, pp. 1-15.

[53] **EATMP (2001).** *Towards a Co-operative ATS Concept: The COOPATS Concept*, Version 1.0, Brussels: EUROCONTROL.

[54] **EUROCONTROL (2000).** Requirements for the implementation of Automatic and Manual label anti-overlap functions. HRS/HSP-006 (CoRe), EUROCONTROL Experimental Centre *Note N°21/00*, Brétigny, France: EEC.

[55] **Chincholle, D. and Menu, J.P. (1995).** PATRICIA: La Palette de couleurs ergonomiques proposée pour les futurs écrans de contrôle aérien. *IMASSA Nº 95-27*, CENA/NT95704, CERMA, August, Brétigny, France.

[56] **Verplank, W. (1987).** *The Design of Graphical User Interfaces*. Tutorial presented at 3rd Conference of the British Computer Society Human-Computer Interaction Specialist Group, University of Exeter, England, 7-11 September 1987.

Page intentionally left blank

ABBREVIATIONS AND ACRONYMS

For the purposes of this document the following abbreviations and acronyms shall apply:

AB	Action Button *(on pointing device)*
ATM	Air Traffic Management
CENA	Centre d'Etudes de la Navigation Aérienne *(France)*
CoE SAS	Safety Analysis and Scientific Centre of Expertise *(EEC Brétigny, France)*
COPS	Common Operational Performance Specification [22]
CoRe (Project)	Core Requirements for ATM Working Positions (Project) *(EUROCONTROL, EATM(P), HRS, HSP)*
CPDLC	Controller Pilot DataLink Communication
CRNA	Centre Régional de la Navigation Aérienne
CWP	Controller Working Position
D	Distance to move a pointer *(Fitts' Law)*
DAS	Directorate ATM Strategies *(EUROCONTROL Headquarters, SD)*
DAS/HUM	Human Factors Management Business Division *(EUROCONTROL Headquarters, SD, DAS; formerly known as 'DIS/HUM')*
DIS	Director(ate) Infrastructure, ATC Systems and Support *(EUROCONTROL Headquarters, SDE)*
DIS/HUM	Human Factors and Manpower Unit *(EUROCONTROL Headquarters, SDE, DIS; today known as 'DAS/HUM')*
DM	Direct Manipulation
DSI	Denmark-Sweden Interface
EATCHIP	European Air Traffic Harmonisation and Integration Programme *(later renamed 'EATMP' and today known as 'EATM')*
EATM(P)	European Air Traffic Management (Programme), *(formerly known as 'EATCHIP')*

ECAC	European Civil Aviation Conference
ENAV	Ente Nazionale di Assistenza al Volo *(Italian Agency for Air Navigation Services)*
EONS	EUROCONTROL Open and Generic Graphic System
ESCAPE	EUROCONTROL Simulation Capability and Platform for Experimenting
FAA	Federal Aviation Administration *(US)*
GCWP	Generic Controller Working Position *(original name of the CoRe Project)*
GLUE	Generic Linkage for Unifying model Elements
GUI	Graphical User Interface
GUI	Guideline *(EATCHIP/EATM(P))*
HF	Human Factors
HFFG	Human Factors Focus Group *(EATM, HRT; formerly known as 'HFSG')*
HFSG	Human Factors Sub-Group *(EATCHIP/EATMP, HRT; today known as 'HFFG')*
HMI	Human-Machine Interface/Interaction
HRS	Human Resources Programme *(EATM(P))*
HRT	Human Resources Team *(EATCHIP/EATM(P))*
HSP	Human Factors Sub-Programme *(EATM(P), HRS)*
IAA	Irish Aviation Authority
IB	Information Button *(on pointing device)*
ICAO	International Civil Aviation Organization
IPRT	Internal Processing Response Time *(COPS)*
ITI (Project)	Italian Interface (Project)
ISG	Interface/Interaction Style Guide
ISO	International Standards Organisation
LFV	Luftfartsverket, *(Swedish Civil Aviation Administration)*
MT	Movement Time *(in Fitts' Law)*

NATS	National Air Traffic Services *(UK Service Provider)*
OLDI	On-Line Data Interchange *(a EUROCONTROL Standard)*
OSF	Open Software Foundation
PD	PHARE Demonstrator *(1, 2 or 3)*
PD	Press and Drag *(pointer action)*
PH	Press and Hold *(pointer action)*
PHARE	Programme for the Harmonisation of ATM Research in EUROCONTROL
PVD	Plan View Display
REFGHMI	REFerence Ground Human-Machine Interface
RGB	Red, Green, Blue
SB	Special Button *(on pointing device)*
SC	Single Click *(pointer action)*
SD	Senior Director, EATM Service Business Unit *(EUROCONTROL Headquarters, Belgium)*
SURT	Screen Update Response Time *(COPS)*
STCA	Short-Term Conflict Alert
SYSCO	System Supported Coordination *(extension to the OLDI Standard)*
UCSD	User-Centred System Design
UML	The Unified Modelling Language
WIMP	Window, Icon, Menu, Pointer
XHF	Transversal HF requirements
XFL	Exit Flight Level
XML	Extensible Markup Language

Page intentionally left blank

ACKNOWLEDGEMENTS

The authors would like to thank the many people who have contributed to the work contained within this document.

We received considerable support from our sponsors, DAS/HUM Division (EUROCONTROL Headquarters, Brussels), throughout the course of this project and in particular from Michiel Woldring as Manager of the EATM HRS Human Factors Sub-Programme (HFSP).

The HRT Human Factors Sub-Group (HFSG), now known as the Human Factors Focus Group (HFFG), has monitored and contributed to the work leading to the production of this guide.

Many people have contributed time, experience and the results of their own studies to contents of this guide, which has drawn not only on the work done within CoRe itself but also on work done through the PHARE Ground Human-Machine Interface (GHMI) group in support of all three PHARE Demonstrators and through the EATCHIP and EATMP Programmes where we have benefited from working alongside, Chris Brain, Poul Jorgensen, Doug Spragg and Alex Wandels (EUROCONTROL Headquarters, Brussels). In addition:

- Billy Josefsson (LFV, Sweden) provided access to examples of ATM style guides,
- Jean-Luc Vinot (CENA, France) provided us with the ORLY Fonts and much good advice.

We would also like to thank the controllers who gave unstintingly of their time and ideas during the three small scale CoRe Simulations used to test and refine the principles described in this document and the solutions employed in the exemplary interface:

- in Study 1, Jean-Luc Janszen, Peter Slingerland, Steve Bancroft, Stefan Oze & Yann Kermaquer (EEC, Brétigny, France).
- in Study 2, Salvatore Martina (ENAV, Brindisi, Italy), John McGrath, (IAA, Shannon), Kevin McGrath (IAA, Dublin), & Erwin Obergruber (Austrocontrol, Vienna),
- in Study 3, Bernard Diot (CRNA, Bordeaux, France), Alain Signamarcheix (CRNA, Aix, France), Gerard Sors (CRNA, Aix, France) & Andreas Stuhlik (Austrocontrol, Vienna).

Finally, we would like to thank Carine Hellinckx whose patience and precision as the contractor responsible for DAS/HUM Document Configuration has, as always, contributed much to improve the clarity and presentation of this document.

Page intentionally left blank

PART 6: TECHNICAL ANNEXES

Page intentionally left blank

ANNEX 1: THE REFGHMI COGNITIVE FRAMEWORK

Introduction

The following section describes the psychological framework, which was used as the basis for the original REFGHMI. Subsequent revision for inclusion as the CoRe Baseline has been influenced by the work of Neisser [34], and Hutchins [29] in extending Ecological Psychology to consideration of Human-Computer Interaction (HMI).

A Simple Model of Action at the Interface: Controller Activity as Multi-threaded Tasking [24]

Discussion within the PHARE Role of Man Group [28] had already identified that the Norman Seven Stage Model of User Activities ([15], Ch3 & 5) was a useful framework to employ when thinking about the controller performing a task at the interface. Space limitations prevent a detailed description of this model here. The reader is referred to the original references and the brief summary in the PHARE document.

However, certain characteristics of the controller activity suggested that some elaboration of this type of framework was necessary in order to characterise the pattern of activity as opposed to the performance of individual tasks.

While it is not inherent in the Norman Model itself, the psychological literature has a tendency to discuss human task performance in terms of sequence of activity in single psychological tasks in comparative isolation. Firstly, in Air Traffic Control any single task occurs in the context of many others, running before, after and concurrently. There is certainly facilitation, competition, dependency and other complex interaction between psychological tasks.

Secondly, it is difficult to characterise the controller's activity in terms of single goals except in the highest and most general of terms, e.g. 'controlling air traffic'. In reality, at any one time, the controller is trying to satisfy a large number of goals simultaneously. Indeed very often, a single piece of control activity may be directed at multiple objectives. Alternatively, an action may also be directed at general situation maintenance objectives rather than specific ATC objectives, e.g. to simplify the general operational situation.

Figure 7: The Norman Model extended to include the Supervisory Loop (from [24])

Finally, a very important part of all controller roles is using the information available to prioritise and react to the many ATC tasks in an appropriate and timely manner. We have tried to specifically support this element of controller activity in the interface described here and this is linked very firmly to many of the tactics and strategies described in [24], §6.2 and §6.3. In so doing we have imagined a simple extension to the Norman Model by adding a 'supervisory' scanning cycle identifying the activities to be managed. This is illustrated in Figure 7 and described in the following section.

Supervisory scanning and task management

One of the findings of the ODID IV simulation ([19], Annex VII) was that both the Planning and Tactical Controllers had a basic pattern of activity which consisted of a general background scanning pattern around certain critical elements of the control position. This scanning pattern would be interrupted to perform a sequence of actions relating to a single or groups of aircraft, after which the controller would return to the background activity. The interruptions could be of an information gathering nature, of an executive nature, or one followed by the other. This general pattern of behaviour could differ in detail for individuals but was followed in outline by all controllers. We would argue that this general pattern of activity is probably typical of most continuous complex human task performance.

The extension to the Norman model involves the addition of this general background supervisory scanning activity providing the binding between the more classical elements of the existing model. (The hierarchical nesting of the Norman Model can be considered as describing the present representation, but we feel that making this particular level explicit makes visualisation of the controller activity much easier.) The other slight modification is recognition that a large number of goals may prevail at any one time.

The function of the supervisory loop is management of the continuous task activity. It is here that intentions are generated to perform task-related actions. (Note that action may be no more than gathering additional information, by directing perceptual resources or, at more cost, by making inputs that make more information accessible.) The general idea is that the controller operates at the supervisory level until:

a) conditions arise which allow action to achieve existing goals;

b) some event arises , and is recognised as requiring action, e.g. new aircraft enters system, conflict is detected;

c) criteria values are reached which require action, i.e. aircraft deviates, a coordination is not completed in the expected time.

When an intention is generated the controller breaks the pattern and embarks on an ATC activity.

Assumptions about the scanning loop

The model makes a number of psychological assumptions about the nature of this scanning loop:

a) Performing this activity should make efficient use of cognitive resources. This is achieved partly through the fact that it is skilled and very highly practised. The suggestion is also that it should operate as an AUTOMATIC PROCESS in the sense employed by Shiffrin and Schneider [30].

b) It does not necessarily cease to operate when the controller is performing a more specific task although it may operate at:

- lower efficiency;
- with higher thresholds or, more likely, slower information accumulation to reach threshold for important events[37].

c) There is a tendency to always return to the loop after other tasks are performed.

[37] The purpose of this assumption is to explain interruption of tasks to react to special conditions and alarms.

Assumptions about task execution

Similarly, there are a number of assumptions related to the task performance:

a) Seeing a task through to completion is 'psychologically satisfying'. We call this the principle of **Psychological Closure** (a term that has been used elsewhere with a similar meaning).

b) The performance of tasks may take place as either Automatic Processing, Controlled Processing [30] or a combination of the two.

c) When in Automatic Mode, it is only practicable to interrupt a task at the end of one of the automatic 'chunks'. This is analogous to the limitation on the ability to generate verbal reports at similar stages of activity described by Ericsson and Simon [31].

d) In Controlled Processing mode, there will be points in the completion of a task where interruption will be less costly and frustrating. Effectively, this will correspond to the completion of 'subtasks' where some measure if psychological closure is provided.

e) At such break points the controller will be more susceptible to detection of warnings and other changes in the priorities of information on the interface.

f) An important aspect of the scanning is that there is an incidental learning effect as additional information is absorbed. This information may accumulate over time.

The last of these assumptions is potentially of considerable importance. The information gathered in this way plays an important role in helping the user to navigate in the display. It MAY also be an important contributor to maintaining the controller's situational awareness (or picture). A corollary of this, which may be testable, is that an interface that directs the user's attention in too focussed and systematic a way, may impair the development or maintenance of situational awareness.

ANNEX 2: THE PHARE ROLE OF MAN GROUP

Keeping The Controller 'In-The-Loop' (as reported in [24])

The PHARE Role of Man group document [28] suggested that one important element of keeping the operator in the loop is that *liability*, *responsibility*, *understanding* and *power to act* (or *authority*) must be co-located with the operator at the necessary critical system states [28], §2.3.1.2.e.

Understanding and the *power to act* are often addressed in the construction of interfaces, albeit in an indirect way through the prioritisation and presentation of information and the definition of the interaction possibilities.

Liability, the legal significance of controller actions, can be considered as reflected in a variety of measures (which may also have other motivations). For example, the recognised importance of including controllers in the design process, the cultural prerequisite for validation through large scale simulation, and the fact that the final word in using an element of a system or not practically lies with the controllers. All of these recognise the controller's legal status. On the less positive side, data recording and incident reporting procedures also relate to this aspect. It could reasonably be argued that these elements also reflect the notion of *responsibility*. However, it may also be appropriate to ask:

a) If there is a requirement to reflect *responsibility* in the interface design process itself and, if so,

b) how this could be achieved.

At the time that the PHARE PD1 HMI was developed, there had been exploration of these two questions, but PHARE's initial analysis suggested that a number of the changes being proposed in PHARE were likely to have implications for controller responsibility. It was proposed that these were associated with two transitions:

- Firstly, the transition between a paper strip system with telephone links to adjacent sectors and centres and the type of working method described in ODID and the PHARE PD1 Reference system (i.e. a stripless system with SYSCO type [36] electronic exchange of aircraft flight data between centres).
- A second transition, where datalink was employed to mediate the transmission of data and instructions between the ground and the aircraft.

The argument ran that, in the paper strip system, the role of the CWP interface has been to present the operator, in a clear and unambiguous way, with information reflecting events 'in the real world' and the intentions and requirements of aircraft. This information allows the controller to make decisions and perform appropriate operational actions. These actions are not made via the interface, or at least not via the computational element of the interface. The transmission of decisions to the aircraft is made via R/T, to other controllers via the telephone systems, and to colleagues through direct speech. In paper strip systems, even the records of actions are independent of the computational elements for the greater part. The data within the computational part of the system is derived from sensors and the flight plan processing systems.

Interactions with the system are primarily concerned with manipulating the information presentation to the controller himself.

In 'stripless' systems new elements are introduced:

a) In order to benefit from the advanced displays provided by the system, the controller has to make inputs following actions so that the system is kept up-to-date. These inputs are still made following true ATC actions that have taken place and REFLECT the real world (e.g. clear the aircraft to a new level and update the system through a menu input).

b) However a certain number of controller actions now pass THROUGH the system, e.g. proposals for coordination, and the distribution to other controllers of data reflecting controller actions. The first class of information is now anticipating actions in the system and, for the first time, there is the possibility that incorrect interface input values can have DIRECT effects on control procedures. However, this effect is not strong and the consequences of such input actions can be limited by the nature of other procedures. For example, if frequency is not automatically transferred with the input of a TRANSFER on the interface.

At the level of the Advanced Systems, where planning inputs are made to the system which then transmits them directly to the aircraft via datalink, the transition is complete. Here the controller manipulates the interface in order to manipulate events in the real world. An incorrect interface interaction can now result, almost directly, in incorrect behaviour in the real world. A whole new significance is now attributed to interface inputs. There may be problems associated with the fact that a significant and possibly irrecoverable action can be undertaken with very great facility.

It was argued that these transitions have direct implications for the way in which the interface should address the possibility of input errors and their consequences.

ANNEX 3: THE RATIONALE FOR THE USE OF COLOUR IN THE REFGHMI

The Problem (Historically) (as reported in [24])

While it is the opinion of the authors that it is highly artificial to treat the use of colour as a separate from other techniques used to prioritise the display and saliency of information in a graphical interface, it is also accepted that the use of colour has been identified for many years as an issue by the Air Traffic Management Research Community [44],[46],[47]. For this reason, the following section deals specifically with the application of colour in the REFGHMI. It is however, respectfully suggested that it is time for the debate to move on and that for the future we should speak simply of the issue of appropriate display priorities and mechanisms rather than identifying colour uniquely as the issue.

It is also considered that the colour system integration achieved for the PD1 Reference System and the current REFGHMI[38], was one of the most important developments of the PD1 GHMI Design Activity.

Two different approaches

At the time of preparation of the Specifications for the PD1 Reference System, the use of colour in the design of ATC displays had been based mainly on designer judgement, trial and error, and consideration of suitable stereotypic interpretations of the use of colour. In Europe, there were two main exceptions to this, the work of the ODID group itself, and the work sponsored by the UK Civil Aviation Authority on the use of colour in, firstly, radar displays and subsequently other ATC displays [33]. The UK activity is hereafter referred to as the 'Interim NATS Standard'. These two activities took very different approaches to the use of colour but both arrived at a much more systematic and principled application of colour than had previously been available to the ATC community.

Interim NATS Standard

The Interim NATS Standard [33] provides a structured rationale for the use of colour based on the principles and literature of human visual perception. It then provides a system of layered palettes[39], symbology and presentation principles for the display of the information required for generic secondary radar displays, with high levels of legibility and discriminability.

To the observer, the most striking aspects of the displays which result from following the NATS Guidelines are the harmonious pastel colours; the effective use of transparency and the infilled labels for all aircraft on the display.

[38] And now employed in the CoRe baseline.

[39] It was originally intended that these layers should be reflected in system hardware.

ODID approach

The **ODID approach** has a different emphasis. It derives from the attempt to establish an *operational task-oriented system* for the use of colour. The initial approach was empirical but, over the series of ODID simulations, transcended the issues of identification of specific colours to produce a series of related display principles described in [19].

The central theme of the ODID approach is the way in which colour can be used to support the logic of the controller's task. Colour is employed to clearly reflect distinctions central to the controller's 'cognitive model' of which aircraft states are critical for task identification, planning and decision making. The logic was mainly developed in ODID III and further refined in ODID IV where it met with considerable controller approval[40].

Integration of the approaches (PHARE PD1)

The approach taken in preparation of the PHARE PD1 GHMI, and expanded here, was to review the two application systems and then to attempt integration based on the best features of both. The critique is summarised in the following sub-section together with the principles used to drive the integration. Since both systems were founded on sound principles, the integration task proved both simple and direct with little compromise required from either approach. The limited experience with the PD1 system at the time of writing seems to indicate, that while 'tuning' may be required with individual colours, the integration is proving to be better than either of its parents. The details of the compromise as followed in PD1 and the present REFGHMI are described in at the end of this annex.

Critique and Issues in Exploiting the Two Schemes

It appears that the mechanism recommended for labels (or plaques) in the Interim NATS Standard Document [33] is based primarily on a desire to optimise the legibility of the text of the aircraft label (while still allowing a well designed hierarchy of information saliency for the supportive map and airspace information). Extraction of the aircraft information by the controller can be considered as consisting of two components. The first component involves location of a specific aircraft through the mechanisms of *locating*, *reading* and *identifying* its callsign. This will be referred to as VISUAL SEARCH. The second involves the extraction of the flight information once the correct aircraft has been identified. This can be considered (generally) as READING[41]. In this later

[40] The controllers who participated in ODID IV were unanimous in their approval of the way in which colour was employed. Only two points of dissatisfaction were noted. Firstly, the use of the STCA Red colour on the callsign of aircraft in conflict was not liked. Secondly, the Grey employed for aircraft that had been handed over suffered from a reduction in legibility, as well as the reduction in saliency which was intended.

[41] For those more familiar with the literature on visual perception, it is recognised by the authors that this distinction between visual search and reading represents a simplification. We have already referred in ANNEX 1 to a notion of an overall SCANNING or MONITORING activity which involves more than just the Radar PVD and precedes the identification of a particular target. At the other end,

area of label text processing, it seems highly probable that the NATS Standard will be superior to the results so far achieved in ODID.

The information on which NATS label design criteria are based derives from a thorough review of the colour literature covering both the psychophysics of vision and the application of colour in information displays. Most of the available literature in psychophysics deals with repeated single instance, stimulus presentations. The information display literature deals primarily with structured text displays of a comparatively homogeneous nature. Indeed, the application of colour is often used to provide additional structure to allow more rapid access to such displays. Consequently, the criteria optimise:

a) The unambiguous perception of colour information.

b) The legibility of text. (Effectively ease of READING.)

It is argued that other factors play a critical part in the controller's use of the RPVD, particularly those relating to VISUAL SEARCH, and that an adequate design solution requires consideration of a number of elements. The solution eventually proposed for PD1 and the REFGHMI takes account of other factors and offsets some loss in legibility during search activity against improvements in other areas, especially the control processes of the visual search behaviour itself. After the search activity has located a candidate target, the legibility advantages are regained for confirmation and subsequent information extraction.

The following two factors suggest the importance of other considerations in controlling search activity:

a) Although the PVD contains text, it is a heterogeneous mixture of both graphical information and text. Further, it is constantly changing, but in comparatively systematic ways (e.g. aircraft do not make large jumps to different locations on the display).

b) The controller interacts with the display over an extended period of time. There is knowledge of the structure, and the history of development, which can play an important part in the user's navigation within the display.

It is this area of navigation within the display and the initial LOCATION of relevant targets that is better supported by the ODID mechanisms. There is evidence from the literature on eye movement control during reading,[42] from psychophysics and from letter recognition literature, to suggest that outline shape information can be used to direct search behaviour. Consequently the

---

the READING processes may also include an element of template matching as well as the fuller reading processes.

[42] Unfortunately, the references are not immediately to hand. I recollect an article in Cognitive Psychology before 1981, (perhaps by Just and Carpenter) which demonstrated that while individual alphanumeric characters can be identified only within a small region close to the centre of the visual field of a fixation (3-4 characters), coarser, lower spatial frequency information could be processed out to 9 or 10 characters. It was postulated that this allowed sufficient information to be extracted to identify small prepositions and endings like 'ing' in order to permit more efficient placement of the next fixation during reading. Similarly, there is work by David Navon (circa 1977) which shows that in letter recognition, lower spatial frequencies are processed before higher ones, and work on the psychophysics of vision which again suggests the prior processing of low spatial frequencies.

These lines of work converge to suggest that outline shape information could be employed to guide search behaviour.

ODID solution allows size, shape and colour to assist in the search process. Because of the minimum information principle, it is probable that the shape of an ODID label provides the controller with information that a particular aircraft requires more or less control action even before the identity of the aircraft can be extracted. This shape information is lost by the homogeneous rectangular contour of the infilled label.

Another consideration is the visual masking effect that will be produced by the edge contours of the infilled label, and is likely to impair recognition of the characters within the label. This effect will be stronger if the edges of the label are closer to the text and if the contour is stronger (i.e. if the plaques are edged with black). Further, the existence of all the infilled radar labels introduces a lot of vertical and horizontal contour information (the edges of the labels). This is an artefact of the mechanism employed to produce improved label contrast. It seems probable that this contour information serves no useful purpose during visual search and scanning processes and, on the contrary, will act as noise impairing the processes.

Finally, there is the problem of the greater effect of label overlap produced by the infilled label. It is true that infilling makes the top label much easier to read, but it impairs the ability to identify the lower label as well as to extract its more detailed contents. What is more important is that any input mechanism so far suggested to flip the labels requires that the mouse cursor be moved to the area of overlap, even if it does not require an additional button press. Not only does this create an additional, and comparatively task-irrelevant, activity for the user, but also it implies that the attention of the user must have already focused on the screen location. As has already been argued, location of the correct aircraft becomes more difficult with infilled labels, so the cost in impairment of controller efficiency is potentially high.

Conflict Resolution: Integrating the Approaches

In summary, four implementation issues were identified in integrating the two systems to avoid the identified weaknesses and take advantage of the strong points of the approaches.

a) It was not clear that current hardware could readily support the direct hardware implementation of the layering structure of the NATS Interim Standard.

b) The NATS palette structure [33] was likely to require a small, but significant, amount of extension in order to deal with the additional requirements generated by:

   i) The general infrastructure of the windowing environment.

   ii) The provision of visual texture and highlighting required to support a Motif-type look[43].

c) There was a potential conflict between the mechanisms employed by ODID to communicate clearance status of an aircraft (by general label profile and size) and task status (by the colour of text of the label), and the mechanism

[43] This look is not merely decorative. The textural quality of the environment provides mechanisms to indicate the status of radio buttons, the icon/buttons of the toolbox, etc.

of fixed-size infilled labels employed in the NATS Interim Standard. The latter uses infilled labels to provide both task status and a stable contrast level for the (black text) label information against the changing colours of the map background.

d) The infilled label causes a novel form of label overlap problem which requires additional input mechanisms[44] and user focus in order to be able to read overlaid label information.

In order to achieve a practical reconciliation of these few differences the following approach was recommended for both PD1 and the REFGHMI:

- The palette sets recommended in [33] should be adopted irrespective of the ability of the PD1 hardware to implement the full, layering philosophy.
- The palette sets should be extended in a minimal and systematic way to provide support for the textural characteristics of the window system infrastructure.
- The principal (non-textural) colour elements of the windowing infrastructure would be drawn, where possible, from the existing NATS palettes, especially from the background layer palettes. Should this fail, minimal extension of the palettes would be undertaken by the design team.
- The ODID solution of coloured text and unbounded labels should be employed to retain the label profile information presently in ODID. The logic of the text colouring would follow the state distinctions employed in ODID for En-route Sectors. The actual colours employed for the text should (as far as possible) be drawn from the existing NATS palette set.
- However, when the controller locates the cursor within an aircraft label, that label will be highlighted by converting into the appropriate form derived from the Interim NATS Standard [33]. This would avoid the requirement for additional input mechanisms to avoid the compounded label overlap problem associated with infilled labels.

The resulting palettes are illustrated in ANNEX 4.

[44] A totally opaque label is much more effective in obscuring the information of an overlapped label. Currently, all the mechanisms suggested to deal with this problem require movement of the mouse pointer to the location of the overlap.

Page intentionally left blank

ANNEX 4: THE CURRENT CORE BASELINE PALETTE

Colour	Logical name(s)	% RGB	0-255 prop.	Colour	Logical name(s)	% RGB	0-255 prop.
	Black TextUnselected SpeedVector TrackSelected	0, 0 ,0	0, 0 ,0		Window Menu Button2	44, 51, 57	112, 130, 145
	White Transferred In TextSelected LeaderLine Trail TrackUnselected	100, 100, 100	255, 255, 255		WindowHighlight MenuHighlight Button2Highlight	64, 71, 77	163, 182, 196
	Assumed	93, 93, 78	238, 238, 200		Button2Depressed	24, 31, 37	69, 79, 94
	Advanced HandleLight	92, 68, 68	235, 173, 173		Background RangeRings	48, 45, 45	122, 115, 115
	Concerned	72, 61, 28	184, 156, 71		BackgroundHighlight BeaconHighlight DeselectUnavailable	75, 75, 75	191, 191, 191
	Coordination	100, 41, 60	255, 105, 153		Shadow	20, 20, 20	51, 51, 51
	NotConcerned Beacon MinorRoute Airfield	65, 65, 65	166, 166, 166		Worktop	29, 32, 29	74, 82, 74
	Alert	100, 10, 10	255, 26, 26		MajorRoute	44, 46, 48	112, 117, 122
	Warning Cursor RangeBearing ElasticVector StoredValue	100, 90, 12	255, 230, 31		LandMass	36, 34, 34	92, 87, 87
	LoA	44, 66, 55	112, 168, 140		WaterMass SelectUnavailable	39, 39, 41	99, 99, 105
	LoAHighlight	64, 86, 75	163, 219, 191		SectorOnLand	39, 37, 37	99, 94, 94
	Trajectory	46, 88, 31	117, 224, 79		SectorOnWater	42, 42, 42	107, 107, 107
	HandleDark	70, 50, 50	178, 127, 127		RestrictedOnLand	37, 30, 30	94, 77, 77
	Button1	68, 57, 41	173, 145, 105		RestrictedOnWater	39, 34, 36	99, 87, 92
	Button1Highlight	88, 77, 61	224, 196, 156		RestrictedOnSectorOnLand	38, 33, 33	97, 84, 84
	Button1Depressed	48, 37, 21	122, 94, 54		RestrictedOnSectorOnWater	42, 37, 37	107, 94, 94
	Button1Restriction	65, 21, 20	166, 54, 51		Interaction		0, 255, 255

Page intentionally left blank

ANNEX 5: THE USE OF TRANSPARENCY IN THE INTERFACE

To be added at a later date as functionality becomes available.

Page intentionally left blank