EUROPEAN TELECOMMUNICATION STANDARD

DRAFT
pr **ETS 300 707**
June 1996

Source: EBU/CENELEC/ETSI-JTC Reference: DE/JTC-TTEXT-EPG

ICS: 33.020

Key words: broadcasting, TV, data, transmission, protocol, Teletext, electronic programme guide, TV Guide

European Broadcasting Union EBU UER Union Européenne de Radio-Télévision

Electronic Programme Guide (EPG); Protocol for a TV Guide using electronic data transmission

ETSI

European Telecommunications Standards Institute

ETSI Secretariat

Postal address: F-06921 Sophia Antipolis CEDEX - FRANCE
Office address: 650 Route des Lucioles - Sophia Antipolis - Valbonne - FRANCE
X.400: c=fr, a=atlas, p=etsi, s=secretariat - **Internet:** secretariat@etsi.fr

Tel.: +33 92 94 42 00 - Fax: +33 93 65 47 16

Contents

Foreword

This draft European Telecommunication Standard (ETS) has been produced by the Joint Technical Committee (JTC) of the European Broadcasting Union (EBU), Comité Européen de Normalisation ELECtrotechnique (CENELEC) and the European Telecommunications Standards Institute (ETSI) and is now submitted for the Public Enquiry phase of the ETSI standards approval procedure.

NOTE: The EBU/ETSI JTC was established in 1990 to co-ordinate the drafting of ETSs in the specific field of broadcasting and related fields. Since 1995 the JTC became a tripartite body by including in the Memorandum of Understanding also CENELEC, which is responsible for the standardization of radio and television receivers. The EBU is a professional association of broadcasting organisations whose work includes the co-ordination of its Members' activities in the technical, legal, programme-making and programme-exchange domains. The EBU has Active Members in about 60 countries in the European Broadcasting Area; its headquarters is in Geneva *.

* European Broadcasting Union
Case Postale 67
CH-1218 GRAND SACONNEX (Geneva)
Switzerland

Tel: +41 22 717 21 11
Fax: +41 22 717 24 81

Proposed transposition dates	
Date of latest announcement of this ETS (doa):	3 months after ETSI publication
Date of latest publication of new National Standard or endorsement of this ETS (dop/e):	6 months after doa
Date of withdrawal of any conflicting National Standard (dow):	6 months after doa

Blank page

1 Scope

This draft European Telecommunication Standard (ETS) specifies the data formats for data broadcasting applications in general, and, specifically, for the Electronic Programme Guide (EPG) or TV Guide, also known by TV Info. The data required for these applications is transmitted by means of a data broadcasting protocol using the Vertical Blanking Interval (VBI) lines of a TV signal. It is also intended to coexist with other data-broadcasting applications running on affordable TV-sets and VCRs. The EPG enables a receiver / decoder to store a programme database (and, if available, a navigation database), which can help the viewer to find the programmes of his preference.

2 References

This ETS incorporates by dated and undated reference, provisions from other publications. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this ETS only when incorporated in it by amendment or revision. For undated references the latest edition of the publication referred to applies.

[1] Draft prETS 300 708: "Data Transmission within Teletext".

[2] Draft prETS 300 706: "Enhanced Teletext Specification".

[3] prETS 300 231: "Television Systems; Specification of the domestic video Programme Delivery Control system (PDC)".

[4] ETS 300 468: "Specification for Service Information (SI) in Digital Video Broadcasting (DVB)".

[5] ISO 639: " Code for the representation of languages".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of this ETS, the following definitions apply:

bundle: The data carrier as provided by ETS 300 708 [1]. It comprises a number of streams.

composite: A composite decoder is able to receive and to decode EPGs from more than one network and to combine them into a single composite EPG.

control data: This term is used to denote a combination of data fields in the EPG data structures which undergo a special encoding procedure.

escape sequence: A means of introducing additional, more complex display components into text strings (e.g. national characters, pictures).

event: A predefined action which may be selected in a product's user interface menu.

far programme: A programme which is not a near programme.

Filtered (EPG): The EPG presented to the user is filtered if it contains less information than the EPG transmission. The filter in the decoder disregards unwanted or undesirable items, e.g. networks or programmes of certain types.

full EPG: A Multiple Channels EPG which, as a broadcast type includes navigation and sorting information, or as a decoder type makes use of such information if it is available.

hamming 8 / 4: A code for error protection as used within Teletext transmission. It allows single bit errors to be identified and corrected, and double bit errors to be detected.

header: The banner introducing the menu on the screen.

level 1.5, level 2.5, level 3.5: These are Teletext presentation levels.

menu: An arrangement of events displayed via a product's user interface, requiring user interaction.

multiple channel: An EPG broadcast type which comprises information on programmes from more than one network on which the EPG is delivered, or an EPG decoder type which can acquire and display information on programmes from more than one network regardless of the channel to which the receiver is tuned to.

navigation: The user interaction via menus leading to the selection of information.

near programme: A programme that starts during the course of the day or the next day up to at least the end of the evening's programmes.

nibble: A data entity of 4 bits.

page: In the sense of memory capacity this is space for a Teletext page or 1 KB.

parity: A code for error detection as used within Teletext transmission.

point-and-click: An easy way to program a TV or VCR from within the EPG by a cursor which is moved over the wished programme and a confirmation of the action is issued.

stream: A data sequence organised in a block structure as described in ETS 300 708 [1].

string: A data type covering text strings and references.

string data: This term is used to denote a combination of data fields in the EPG data structures which undergo a special encoding procedure.

structure: A data type.

teletext: A data delivery system within television transmission.

this channel: An EPG broadcast type which comprises information only on the programmes of the network on which the EPG is delivered, or an EPG decoder type which can only acquire and display information on the programmes of the channel to which the receiver is tuned to.

3.2 Abbreviations

For the purposes of this ETS, the following abbreviations apply:

AI	Application Information (Structure)
ATS	Auto Tuning System
BI	(Data Broadcasting) Bundle Inventory
CA	Conditional Access
CI	Conditional Access Information Structure
COP	Code of Practice
DRCS	Dynamically Redefinable Character Sets
DVB	Digital Video Broadcast
EPG	Electronic Programme Guide, also referred to as TV Guide or TV Info
ETS	European Telecommunication Standard
HI	Helper Information (Structure)
LCD	Liquid Crystal Display
LI	Language Information (Structure)
LSB	Least Significant Bit
MI	Message Information (Structure)
MIP	Magazine Inventory Page
MSB	Most Significant Bit
NI	Navigation Information (Structure)

OI	OSD Information (Structure)
OSD	On Screen Display
OSI	Open System Interconnection
PDC	Programme Delivery Control
PI	Programme Information (Structure)
PTY	Programme TYpe
TI	Title Information (Structure)
TOP / FLOF	Means of navigation within certain Teletext pages
TV	TeleVision (set)
UI	Update Information (Structure)
VBI	Vertical Blanking Interval
VCR	Video Cassette Recorder
VPS	Video Programming System
VPT	Video Programming by Teletext
WST	World Standard Teletext

4 Introduction

4.1 Structure of this ETS

- Introduction:

The introduction subclause outlines the relationships between all parties involved in collecting, providing, broadcasting, formatting and presenting a TV Guide. The nature of an EPG as a database stored in a product and accessed by the user via on screen menus is also introduced. To accommodate the wide range of products and broadcasts possible with such a system the idea of "Types of EPG“ is introduced.

- Presentation Techniques:

The image seen by the user is crucial to the success of the EPG. The menus and screen layouts are very far removed from the traditional image of Teletext. In short the product is now responsible for the menus on the screen layout. The user will be able to perform interactively with these menus. It is essential that the information provider be aware of possible screen layouts to ensure that the information is displayed in a pleasing and correct way. Therefore this clause outlines a "code-of-practice" between TV setmaker and information provider on "how the information will look".

- Installation and Consistency:

Care needs to be taken to ensure that products can handle the volume of data efficiently and accurately. Also products will probably be able only to handle one Full EPG; however more than one might be available so how to choose the right one is important.

- Scheduling:

The parties involved in providing a TV Guide need flexible and efficient means to modify (delete, insert, alter) data already stored in the receiver's EPG database.

- The Data Groups (clauses 8 to 11):

Now that the functions and requirements of the TV Guide have been listed in the preceding clauses the required data is grouped together in functional blocks, and the detailed encoding and numbering of the data takes place.

NOTE: This ETS is independent of the transport layer (according to OSI layers). EPG data will be broadcast via a Teletext data channel. The actual transportation system is laid down in ETS 300 708 [1].

4.2 How does an EPG benefit a TV / VCR user

In the era of cable / satellite TV a user can expect to receive maybe 30 channels offering perhaps more than 1 000 programmes per day. A system to allow the user to quickly and efficiently find the programme of his choice is becoming ever more attractive. While history has shown that paper can be a very useful medium to convey programme information, simply giving information is no longer enough.

The value of an EPG to a user is to be informed of the most interesting programmes that fit his viewing criteria. Now the user can see if a programme of his choice is available within the next few days and on what channel. Or, the user can select to be informed of the best programmes by means of the rating an information provider has associated with the programme data. Similar attributes such as the language of the programme, its subtitles and audio description or the indication of the unsuitability of the programme for viewing by children can be included.

Integration into the products will also support more comfortable ways of interacting with the devices - e.g. VCRs can be programmed (also for VPS / PDC) by "point-and-click" paradigms rather than complicated manipulation of cryptic keys.

It has also been an additional goal to define a protocol that could allow a TV or VCR with current analogue signals to have a comparable functionality and those planned with DVB. To this extent some (sub)clauses e.g. the definition of programme types have been taken straight from the DVB specification. Future products will strive to minimise the differences between signals coming from an analogue source and those from a digital one when presenting information to the user.

The EPG provides the functionality required by the viewer to select the programmes that are to be viewed and providing an easy route to transfer this information to the TV set or Video recorder by:

- "storing" the data as a database in the TV set or VCR;

- "separating" the way the information is presented or displayed from the way in which the data is transmitted;

- allowing the viewer to selectively sort and store information according to his "preferences";

- using a pre-defined refreshing sequence so that the most critical information is always available;

- using storage in the end-product so that the viewer has instant access to information about future programmes and the network operator can reduce the bandwidth required for an optimal performance.

The EPG also provides the VPS / PDC recording pre-selection codes which can be used by receivers to perform a VPS / PDC controlled recording.

The presentation techniques used in the EPG are very critical to its success. The two extremes are:

- the set collates all the data and decides how to display the information; and
- each TV Guide provider can decide how his information is to be presented.

The main conclusion in this ETS is that a common presentation technique is required, i.e. a solution somewhere in the middle. The solution is based on a minimum OSD requirement of Level 1.5 Teletext. Displays that can be enhanced through Level 2.5 and 3.5 Teletext to full bit-mapped image based systems.

However, a general purpose OSD template has to be used. This is the subject of clause 5.

By using a general purpose template it will be possible for:

- TV Guide providers to influence the layout and create their own identity;
- set makers to offer advanced OSD and powerful sorting techniques;
- viewers to obtain a standardized, consistent interface within one product.

4.3 Types of EPG

To allow a range of broadcasts and products the notion of "Type of EPG" is introduced.

This means transmitting information for this channel, multiple channels and Full EPG.

As a minimum a receiver would require 4 KB of storage.

There are three possible types of EPG available from all these combinations:

- This Channel = The User gets a this channel EPG;
- Multiple Channel = The user gets a multiple channel EPG;
- Full EPG = The user gets a Full EPG (navigation added).

Table 1: Types of EPG

	This Channel Service	**Full EPG Service**
Sets with ≤ 4 KB of Memory	This Channel	This Channel or Multiple Channel
Sets with > 4 KB of Memory	This Channel	Multiple Channel

The type of EPG that a user gets depends on the transmitted database and the receiving equipment. The above table gives a simple presentation of what the user can get. The user can never get more than what the service includes or what the receiver can decode.

NOTE: Refer to annex G and clause 7, for the description and use of TV programme information from normal Teletext service within EPGs.

4.4 EPG data transmission

Supplying the information in the form of data rather than the traditional visual text information will allow new powerful sorting and display techniques. These techniques will enable users to find the programmes they want and with the right amount of background information. A database is maintained within the product with specially encoded information. Viewers can access this information for display according to their wishes, guided by the information and style of the received data.

A service will include data for its own network's programmes, or a service will include data about its own and other networks' programmes. In any case, the data in the EPG is always formatted in the same way.

The data is sent invisibly to the viewer e.g. via the Teletext data channel (as hidden pages / packets as described in ETS 300 708 [1]). In this case EPG data has to co-exist with other Teletext transmissions.

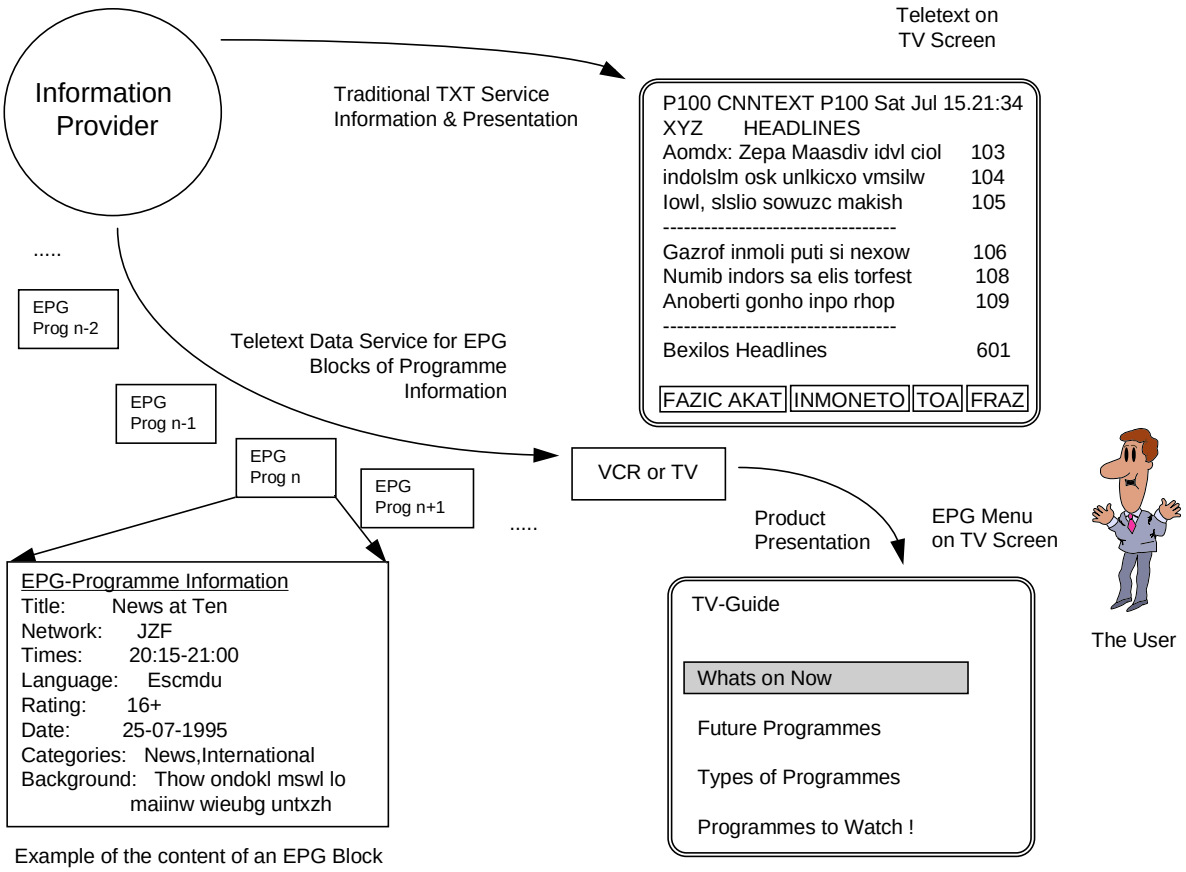


Figure 1: EPG Data Transmission

4.5 Delivery and reception of the EPG

For the application of an Electronic Programme Guide the required data protocol shall take the needs of several parties into account. This subclause develops simple models for the delivery of EPG data to the decoder by defining the logical functions involved in these processes. It should be noted that the logical functions may all be performed by single or multiple independent entities:

listing provider: provides the listing information for multiple TV services;

TV Guide provider: assembles this information to form a Multiple Channels EPG;

network operator: transmits the television programmes and the EPG data;

TV service provider: provides the layout of television programmes for transmission.

Similarly, the EPG engine in the television set or VCR can be split into the following logical functions:

decoder: responsible for collecting and decoding the transmitted EPG data;

EPG database: database of stored EPG data, on which filtering functions can be performed;

presentation engine: responsible for the presentation of EPG data under the control of the viewer.

The relationship of these logical functions differs between This Channel, Multiple Channels and Full EPG data, and is illustrated in the following.

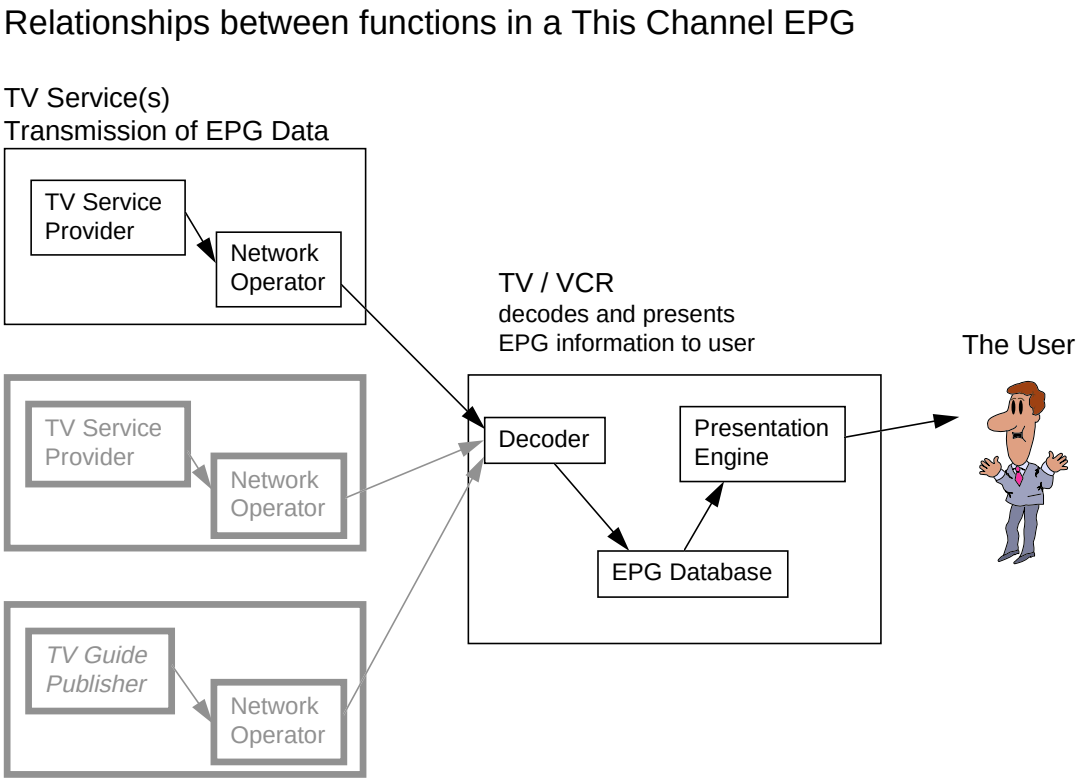


Figure 2: Functions in a This Channel EPG

The This Channel EPG data is generated by a TV service provider (or via a TV Guide provider) for his service only. This data is broadcast by a network operator. It is received by the decoder while tuned to that service. It may be stored by the decoder to reduce the response time to the viewer.

The Multiple Channels EPG data is provided by a TV Guide provider from information obtained from one or more TV listing providers. The data from one or more TV Guide providers is broadcast by a network operator. It is received by the decoder while tuned to that service. The data is stored by the decoder as a database, which may then be searched by the viewer using functions provided by the presentation engine.

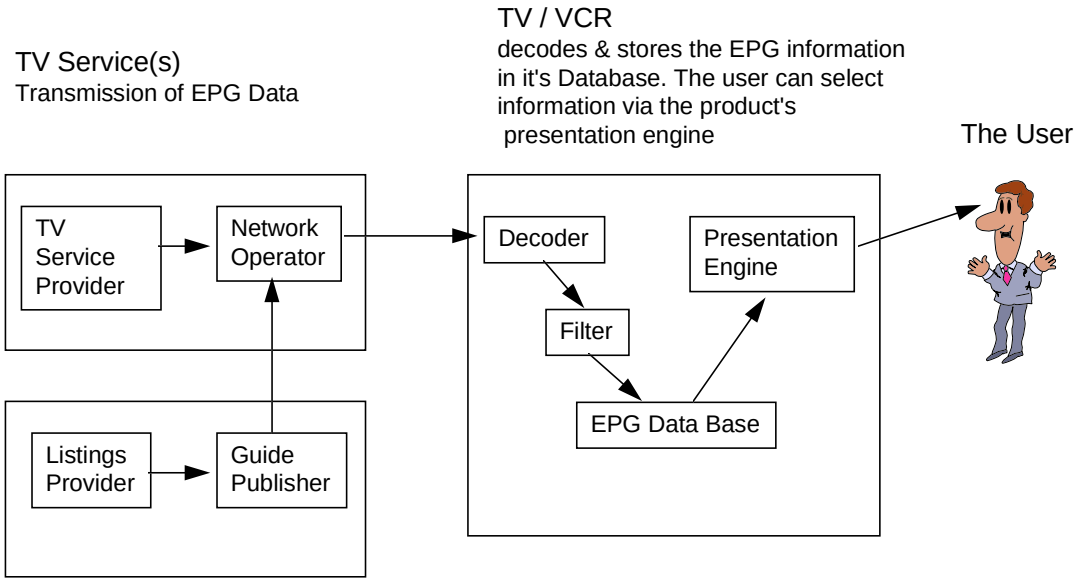


Figure 3: Relationships between functions in a Multiple Channel EPG

A decoder may use the broadcast EPG data in a number of ways. For example, by scanning a number of channels for This Channel EPG data, it is possible for a decoder to build a composite EPG database, on which several presentation functions may be performed.

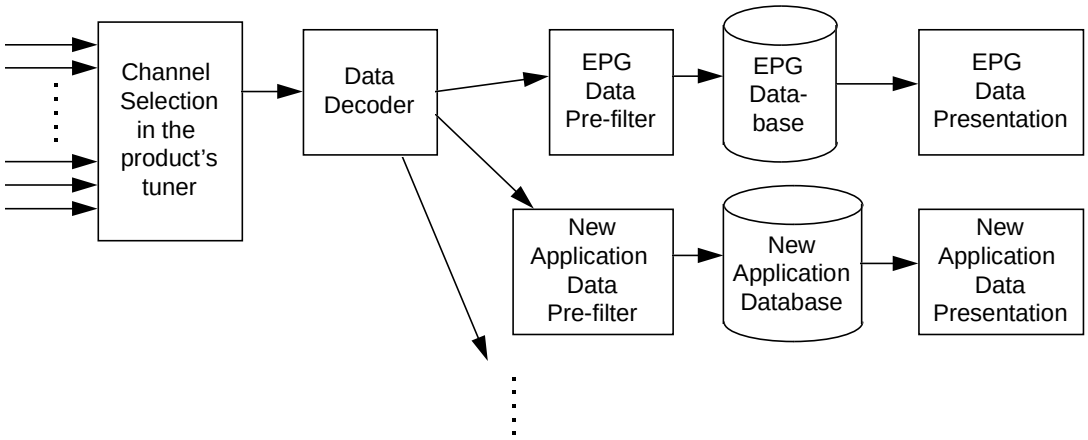


Figure 4: Functional blocks of an EPG decoder

4.6 Summary of EPG types

Service providers or network operators can combine any EPG type with any Teletext presentation level. Backwards compatibility of the enhancements shall be ensured by the service provider or network operator. The bit-mapped Full EPG can be enhanced by using the possibilities of Level 3.5 Teletext presentation, such as the bit-mapped DRCS.

4.6.1 This Channel EPG

The TV Guide provider is supplying information concerning the current programme and programmes that will come next (up to several days ahead), on his own channel. Some additional information such as advertising / messages can also be sent in a format compatible with at least Level 1.5.

The minimum requirements are:

Data: title, information, themes etc. about the current programme from the channel to which the receiver is tuned to;

Storage: a product should have at least 1 KB memory to store the information;

OSD: the TV shall have at least a Teletext Level 1.5 display;
the VCR shall have at least a monochrome display of 10 rows by 20 columns.

The product is obliged to display the TV Guide provider's advertising / messages and header in specific display areas which are flexible in both size and position. In case the TV Guide provider is not supplying this information, the receiver shall adopt a default layout with at least the current time, the current day and the channel or service identification. Examples are given in clause 5.

4.6.2 Multiple Channels EPG

The TV Guide provider is supplying information about the programmes on many networks for several days ahead. A menu (presentation) structure can also be sent to help with the handling of the information. Some presentation enhancements such as advertising / message box and header, can also be sent in a format compatible with Teletext Level 2.5, 3.5, or bit-mapped displays, (with a default to Level 1.5).

The minimum requirements are:

Data: title, information, themes etc. about the programmes of multiple TV networks for several days;

Storage: a product should have at least 256 KB of memory capacity for either composite or Full EPG;

NOTE: The product can also selectively store information as the user requires.

OSD: The TV and VCR shall have at least a Teletext Level 1.5 display, for TV preferably Level 2.5.

- These products can support the Full EPG including thematic (both according to fixed programme types as defined in DVB and also to programmable sorting types) and temporal sorting;

- The information has to be acquired in the background (i.e. without any special action from the user). The products shall be able to display the contents of any menu as specified by the service provider. However, the set can also add its own specific items to the menus;

- The receiver may have filtering mechanisms to select only part of the information which is provided by the service provider due to memory restrictions or through user preferences.

5 Presentation techniques

The success of an EPG heavily relies on the hardware resources available in the product, such as "storage capacity", "display capabilities", "computing power", etc. To match the different types of EPGs and their corresponding minimum requirements, several categories of menus can be distinguished.

5.1 Examples of menus for EPG types

Example of a This Channel
(Now & Next + Broadcaster Message)

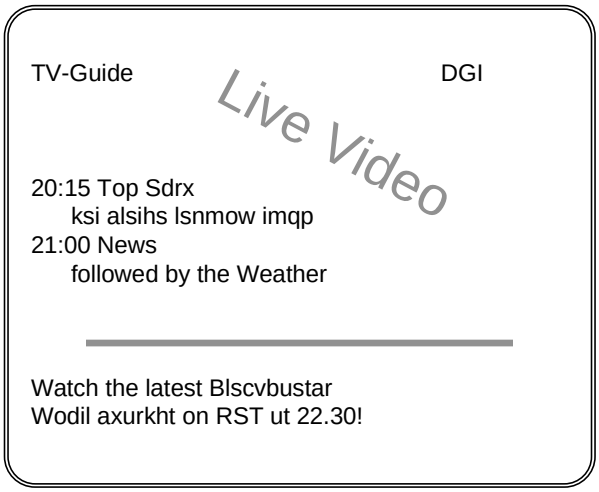


Figure 5: EPG Presentation Example 1

Example of a This Channel EPG
(Today + Linked Message)

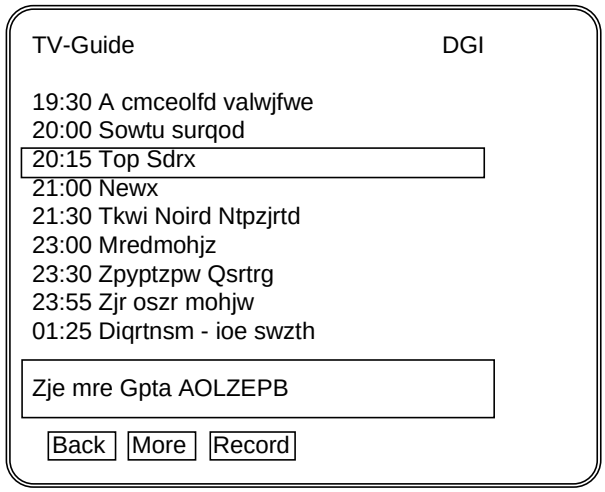


Figure 6: EPG Presentation Example 2

Example of a Multiple Channel EPG

Figure 7: EPG Presentation Example 3

Menu Layout using a Bitmapped display

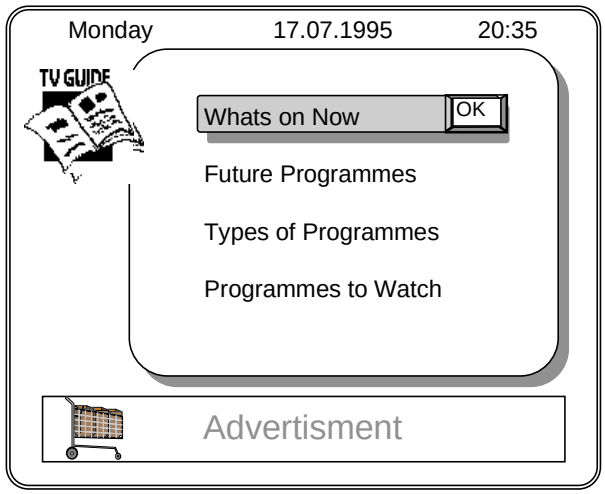


Figure 8: EPG Presentation Example 4

The menus shown above depend on the receiver's capabilities (e.g. record for VCR), the type of hardware (e.g. OSD, memory etc.) fitted and on the information provided by the TV Guide provider.

For a This Channel EPG the product shall provide some means of scrolling through the information. It might be useful to allow typical product specific elements e.g. to allow a VCR to record a programme (as shown in the above example). For a Full EPG the amount of information requires even more advanced menu layouts. Examples are described in more detail in this subclause and in subclauses 5.2 and 5.5.

A Full EPG TV Guide provider can supply the receiver with a menu structure. The following illustrations indicate the sort of menus that may be created.

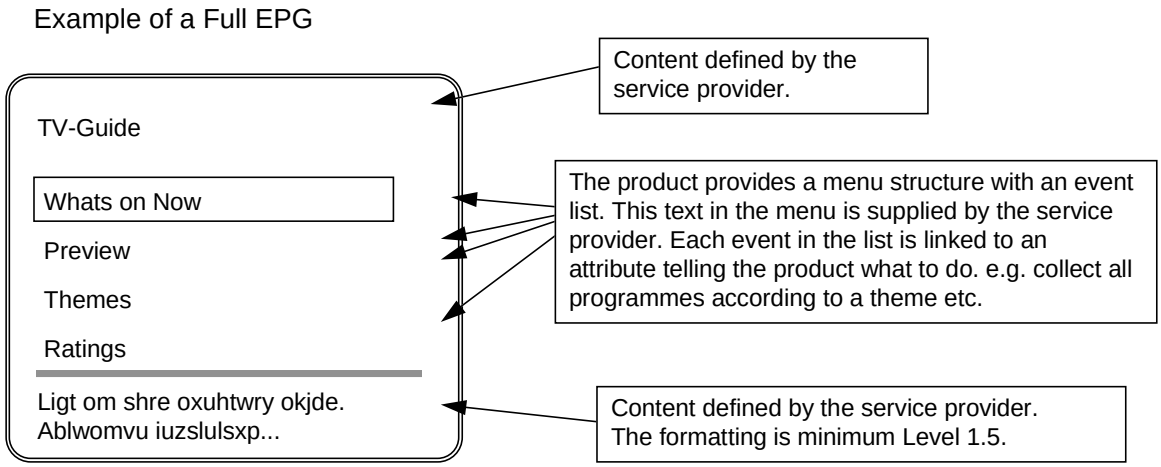


Figure 9: EPG Presentation Example 5

The user has selected the "What's on Now" option. The product now gathers all the relevant information from it's database and presents the information to the user as shown in the representation below. The actual presentation will depend on the end product.

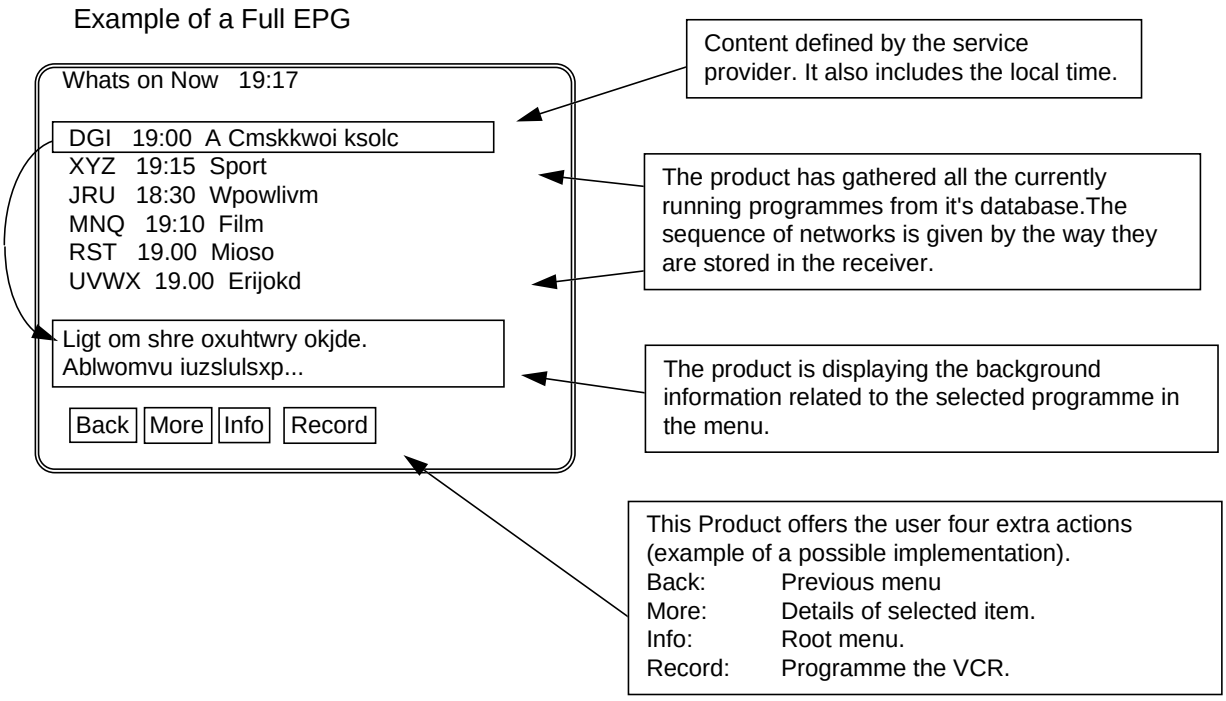


Figure 10: EPG Presentation Example 6

The following menu would occur when the user selects the "Themes" option. Here the product has gathered all the themes used within the current database and presents the results to the user.

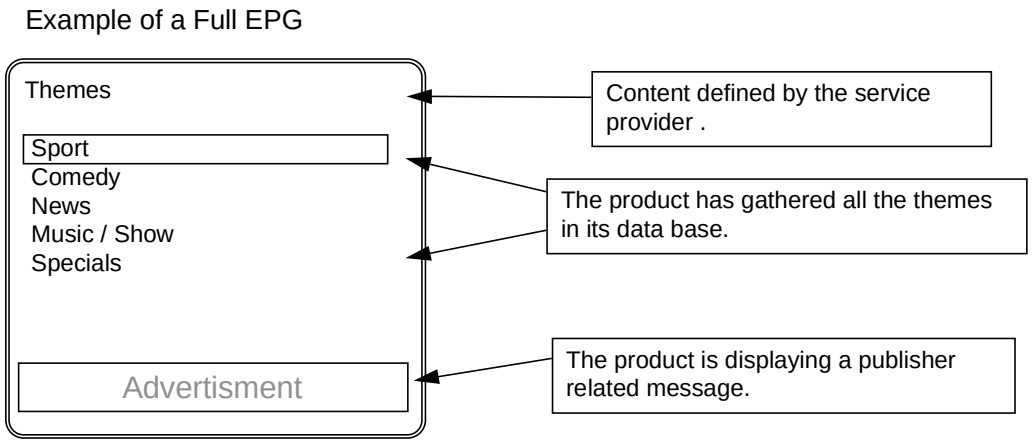


Figure 11: EPG Presentation Example 7

The user now gets an overview of the main themes. However each programme in the database is linked not only to a main theme but also to a sub-theme. After choosing "Sport" the user will be supplied with a list of all the sub-themes related to "Sport".

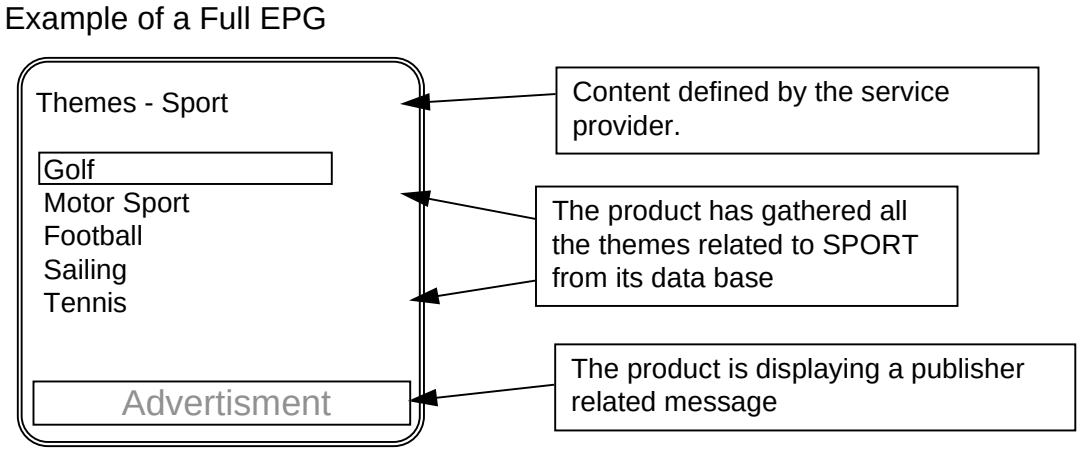


Figure 12: EPG Presentation Example 8

The user can see that his favourite sport of Golf is available. By confirming this choice the product will now produce a list showing all the golf programmes.

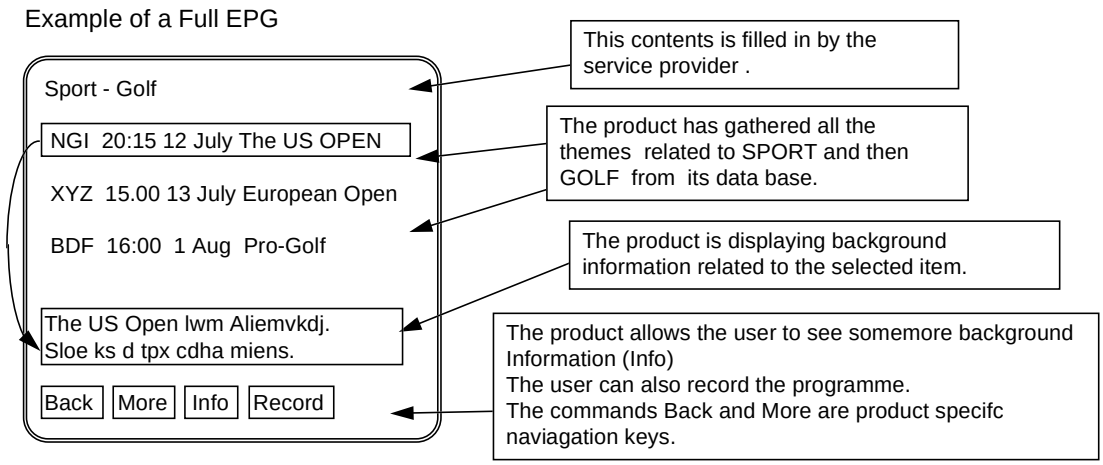


Figure 13: EPG Presentation Example 9

The following is an example of a product which has a new and attractive menu layout. The example below is based on a cursor controlled OSD. The product contains predefined bitmaps. This means that it is not always necessary for a TV Guide provider to download new bitmaps e.g. the Logo for a TV Guide (the open book) comes from the product. The TV Guide provider can still download his own bitmaps when required e.g. the shopping trolley used in the advertising box.

Example of an alternative Menu Layout with a Product using bitmapped display:

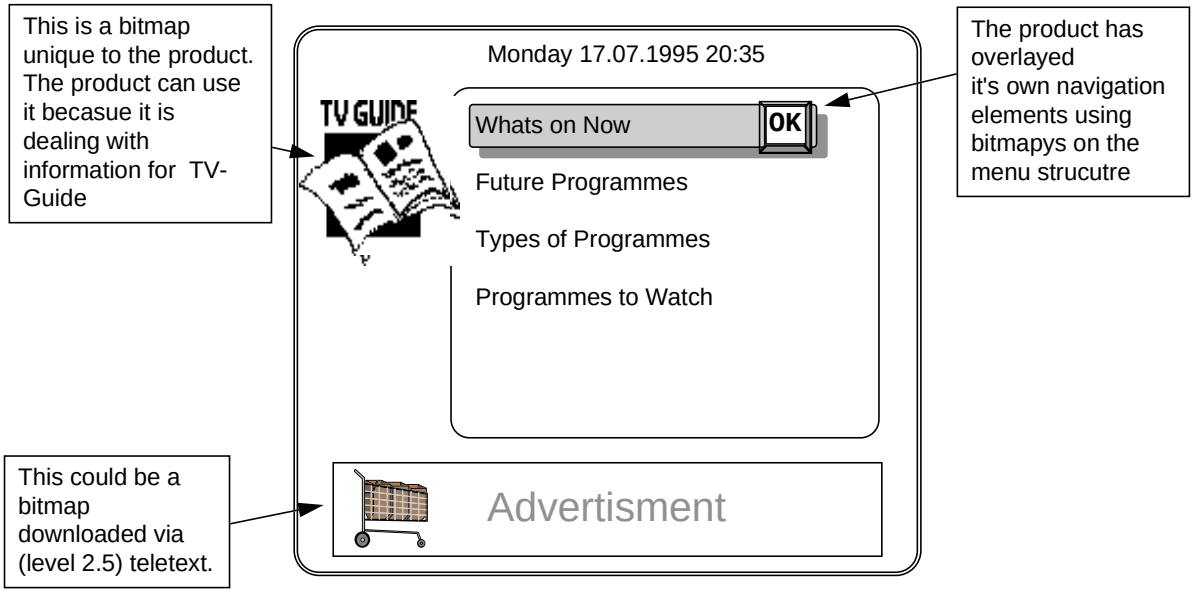


Figure 14: EPG Presentation Example 10

5.2 OSD menu template

The display screen is based on Level 1.5 Teletext with a screen aspect ratio of 4:3 implying a default display of 25 rows by 40 columns. The screen is split into four areas:

Table 2: Overview on EPG Display Areas

Area	Comment
Header Area	The header content and size is defined by the TV Guide provider. (See NOTE)
Event Area	The visual layout and navigation elements are defined by the product. The information is supplied by the TV Guide provider.
Message Area	Messages sent by the TV Guide provider and corresponding with the selected event (i.e. programming information in the case of a TV Guide).
Navigation Area	This can be used by the product to generate its own navigation system
NOTE: In case of a *composite* decoder the *content* of the header area is defined by the set. Refer to subclauses 3.1, 4.5 and 5.4 for description of the composite decoder.	

The Event area is laid out by the product with information from the TV Guide provider. The default content of the event list (This Channel EPG) contains programme information (time and title) starting with the current programme. For a Full EPG different contents may be filled into the event list as dictated by the TV Guide provider navigation system. This is explained in clause 5.5. The Message area is controlled by the service provider via an attribute. The attribute tells the product which TV Guide provider message (if any) is to be displayed. The message may have nothing to do with the EPG, e.g. advertising.

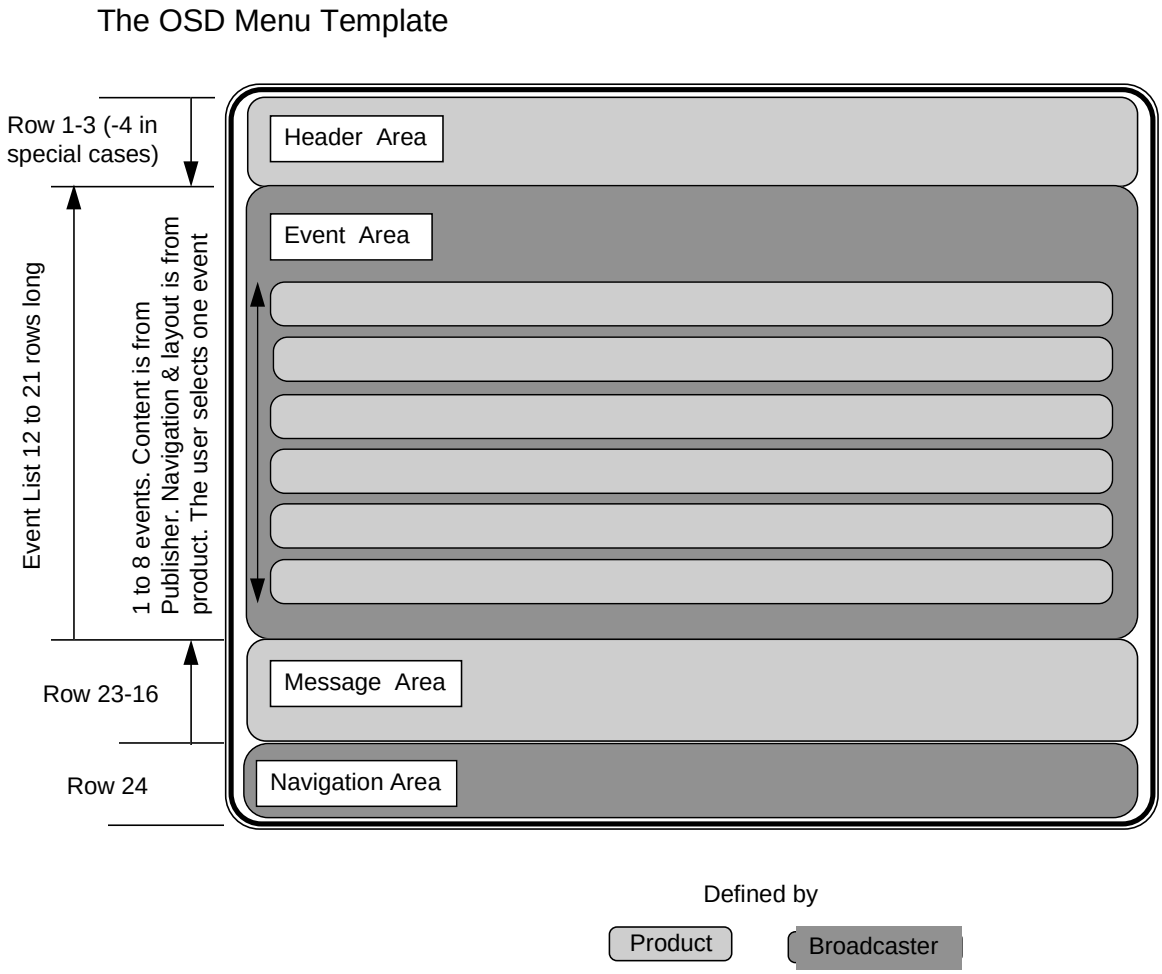


Figure 15: The OSD Menu Template

5.3 OSD menu organisation

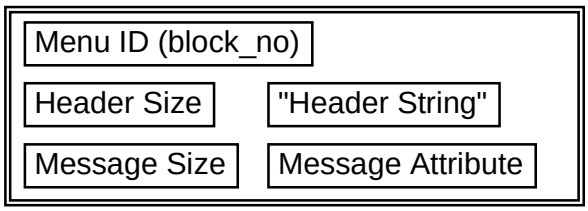


Figure 16: The OSD Menu Organisation

The TV Guide provider needs to deliver the data outlined in the OSD Menu Structure for all types of EPG. This data is used to fill in the menu template. A Full EPG may need to display alternative headers which will be dependent on the navigation scheme used. The Message attribute tells the product what sort of message to place into the message box. A message may be linked to the selected event, e.g. the TV Guide provider tells the product to display programme specific (background) information, or the messages may be unrelated to the selected event, e.g. advertising.

For a This Channel EPG, the event list will include the current and following programmes. In a Full EPG a completely different event list can be created by means of the linked menu list which is described in subclause 5.5.2.

5.4 Presentation modes, window owner / area size

Table 3: EPG Display Areas Owners

<table>
<tr><th>TYPE</th><th colspan="2">This Channel</th><th colspan="3">Multiple Channels</th></tr>
<tr><td>Area definition</td><td>This Channel</td><td>This Channel (Filtered)</td><td>Multiple Channels (Composite)</td><td>FULL EPG

(filtered)</td><td>FULL EPG

(Bitmapped filtered)</td></tr>
<tr><td>EPG header</td><td>TV Guide provider rows 1-3</td><td>TV Guide provider rows 1-3
Filter row 4</td><td>Manufacturer rows 1-4</td><td>TV Guide provider rows 1-3</td><td rowspan="3">The whole screen can have areas by x,y addressing</td></tr>
<tr><td>Event (Picture in text allowed)</td><td colspan="4">Manufacturer - remaining part measured from top of message to bottom of EPG header</td></tr>
<tr><td>Message (Bitmap allowed)</td><td colspan="4">TV Guide provider - fixed from row 23 upwards to row 16</td></tr>
<tr><td>Navigation</td><td colspan="4">Reserved for Manufacturer - row 24</td><td></td></tr>
</table>

Explanation of the variations:

Filtered: The viewer has the possibility to make the receiving set ignore specific information according to his personal preference;

Composite: This receiver can combine the information from various This Channel EPGs to form a "composite Multiple Channels EPG".

Bitmapped: A full EPG decoder with bitmap graphics capability is a possible future extension.

5.5 EPG menu organisation (Full EPG only)

With a Full EPG, the TV Guide provider transmits a menu organisation which will be followed by the product to provide the user with service specific additional navigation means through the EPG database. This function is only useful for a Full EPG as the information from a This Channel EPG may be too limited. In all EPG types the bottom of the menu organisation will be a screen layout conforming to subclause 5.2.

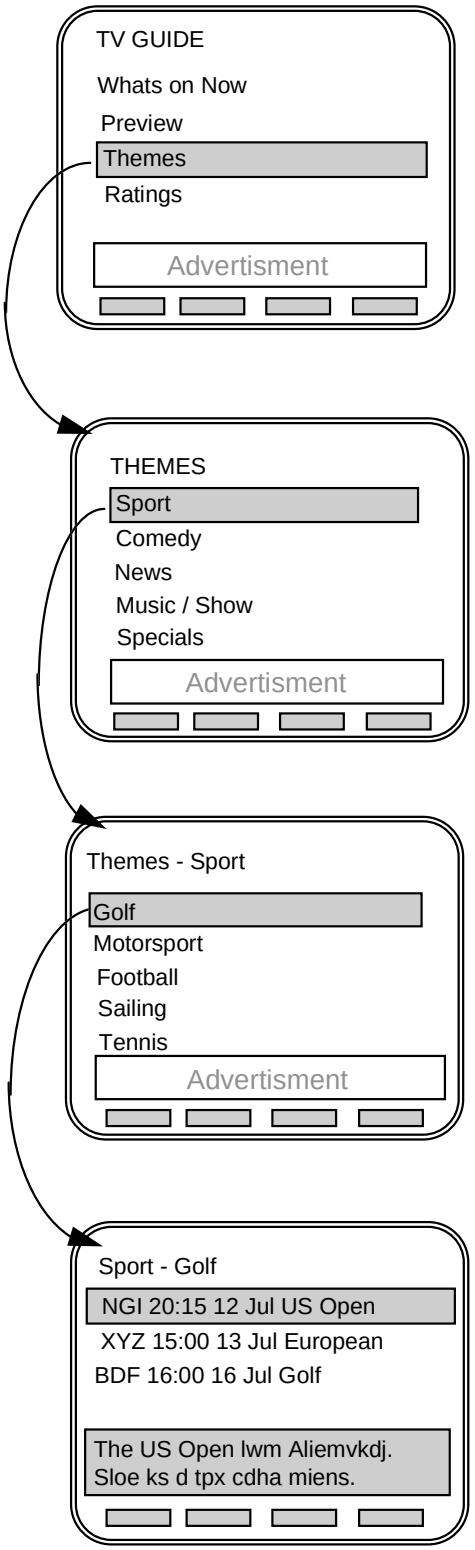
5.5.1 Example of a tree organisation

The EPG menu organisation is in the form of a tree. Each node in the tree is a list of text. There is a defined starting point called the root. Each item in a list is, in turn, connected to another list. Eventually each item in the lists will be linked into a menu which is organised according to the standard menu structure described previously, or is the defined end of the branch of the tree (e.g. a leaf).

As the viewer navigates through the menus, the product is collecting the sorting criteria needed to gather the information from its database.

In the following example it can be seen how the product uses the navigation system to display programmes according to a category and not just according to time (which is the normal default sorting system used in This Channel).

Consider the example of a tree organisation for a typical EPG using thematic sorting.



This is the top level menu in the TV Guide. The header has been defined by the service provider. The text for the events comes form the service provider. The attribute for the event "Themes" has been defined as a second level menu

The product now constructs a second menu according to the wishes of the service provider. The Header has been defined by the service provider. The text for the events comes form the service provider. The attribute for the event sport has been defined as third level broadcster menu.

The product now constructs a third menu according to the wishes of the service provider. The Header has been defined by the service provider. The text for the events comes form the service provider. The attribute for the event Golf has been defined as a specific sorting category.

The product now constructs the basic Menu. The Header has been defined by the service provider. The text for the events comes form the product. The attribute for the event has been defined as a link to the message box by the product. This is the basic TV-Guide menu as seen also in EPGs that do not support service specific navigation.

Figure 17: Example for an EPG Navigation Sequence

5.5.2 Linked menu list description

Each node of the transmitted tree structure is precisely identified with respect to the parent node and the children nodes. Each node contains a List with TV Guide provider defined text strings. Each navigation structure is headed with an ID identifying this list. Each event in the list contains a link ID pointing to the next menu. Each event also contains a sorting attribute which will allow the product at the end of the navigation tree - when it has its leaf menu i.e. the "Generic OSD Template" - to determine the sorting criteria to select the information.

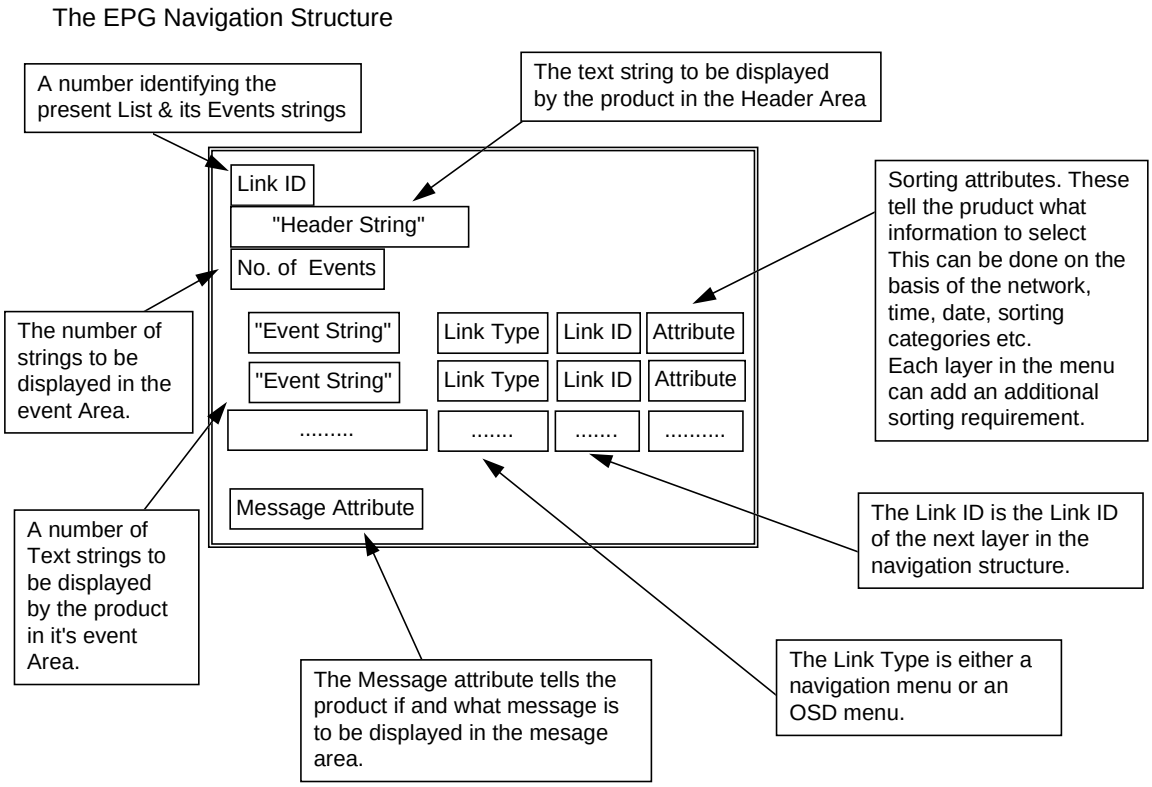
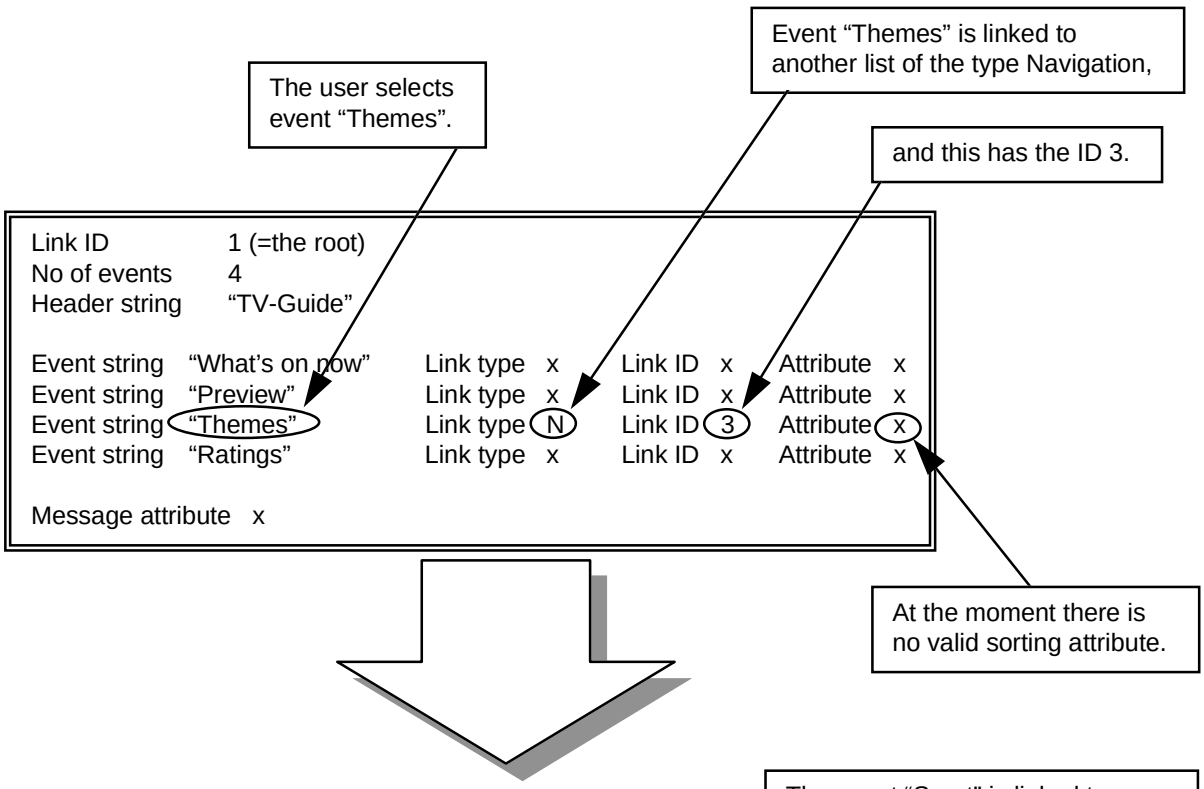
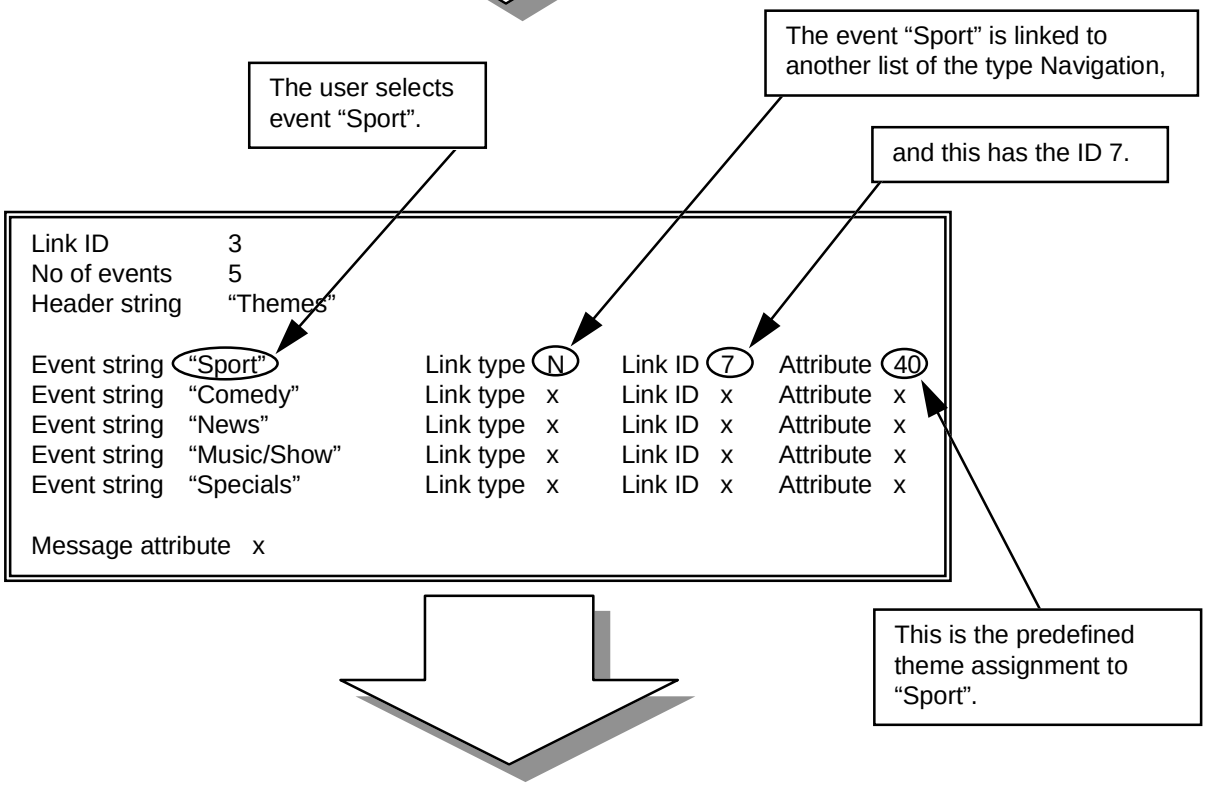


Figure 18: Elements of the Linked Menu List

The following data shall be filled into the navigation tables to generate the "Golf" example of subclause 5.5.1.





(continued next page)

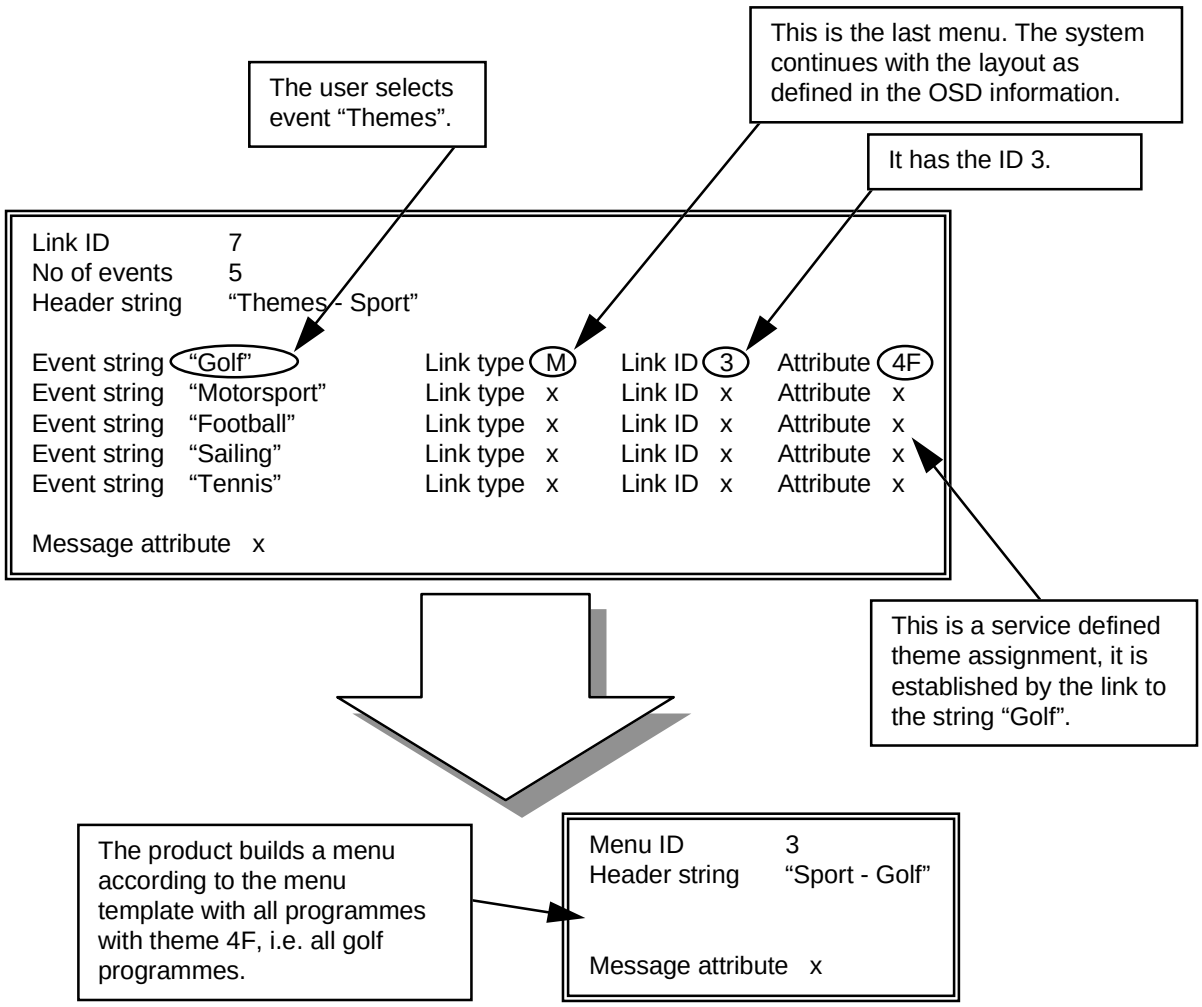


Figure 19: Example for the Linked Menu List

5.5.3 Attributes

The attributes referred to in this ETS are in fact filter criteria. In the absence of attributes, the default behaviour of a TV set is defined by the TV set maker.

Attributes (referred to also as event attributes) provide a powerful means for the EPG to filter programmes according to the user's preference. This preference, e.g. sports, is expressed as a combination of a criterion and a desired value. The criterion refers to one of the fields in the programme information structure.

The criteria that can be used are:

- DVB programme category (theme[]: value);
- Custom category (sortcrit[] : value);
- Editorial rating (editorial_rating : value);
- Parental rating (parental_rating : value);
- Language (language : value);
- Start / Stop time;
- Date.

In order to enable more sophisticated filtering, the preference can be expressed as a combination of multiple attributes.

The attributes are attached to the events (programmes), as described earlier in subclause 5.5.2. Therefore attributes can occur at various nodes in the navigation tree. The criteria are applied at a leaf node of the navigation tree, where a programme list is shown according to the generic OSD template. For this purpose, the criteria of all the parent nodes are combined to determine which programmes have to be shown at the leaf node.

If more than one criterion is defined, the resulting criteria are combined. If multiple instances of the same criterion type are specified, a logical OR operation is performed. For example, when two attributes define different languages, all the programs that support either language are shown. Note that this operation is not always possible, for example in the case of time-related criteria. If several different criteria are specified, a logical AND operation is performed. For example, when language is specified as 'English' and theme DVB 1 as 'Movie', only the English-language movies are shown.

6 Installation and consistency in the EPG

6.1 Version Number

To enable decoders to deal with incomplete or partially inconsistent EPG databases, a "Version Number" is transmitted.

A new Version Number (incremented by one) indicates that major parts of the database have been changed. Then a decoder shall refresh its database completely. This occurs usually once per 24 hours.

Several events can initialise a new version of the EPG databases:

- if a day's programmes have been completed and the data for a new day is to be sent; or
- if re-scheduling of the programmes occur during the day.

6.2 TV Guide menu, installation and change

If more than one EPG is available, the user has to choose which one he wants to use.

To help the user in his choice, the following information has to be displayed in the products own installation system:

- identification name of service provider;
- type of service, i.e. This Channel, Multiple Channels or Full EPG;
- general characteristics, description; and
- the changing of channel and guides shall also be supported.

6.3 Unique identification

To allow a product to very rapidly identify the status of the TV Guide, the following information is supplied continuously in the data transmission system (refer also to clause 11):

- If a TV Guide is present and its type;
- Service provider, e.g. the TV Guide provider name;
- TV Guide "Version Number";
- Range of programmes supported;
- Time to guide completion.

The identification of the service has to be available via a characteristic of the number of channels and number of days per channel.

7 Scheduling

The transmission of the EPG data takes place continuously, preferably 24 hours a day. The complete database is retransmitted at regular intervals.

Within an EPG service it may be necessary to distinguish parts of the database by repetition (or refresh) rates, quantity of data or streams (see annex A).

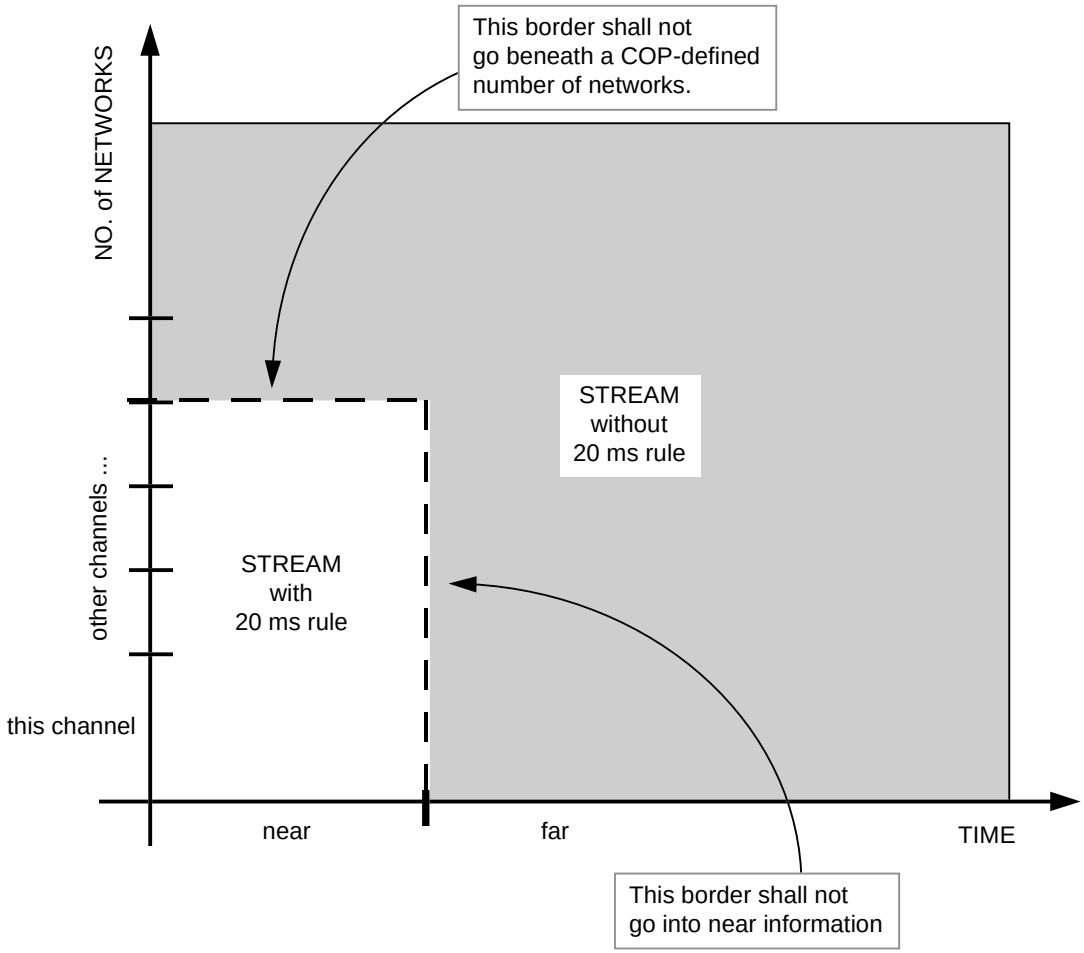


Figure 20: Transmission of EPG Data

The broken line separates the part of the EPG information which shall be transmitted in a stream obeying the 20 ms rule from the part which does not need to obey the 20 ms rule. At least all Near Programme Information of at least one COP-specified number of networks, including "This channel", shall be in the stream which obeys the 20 ms rule. Any further information shall go into the stream which does not need to obey the 20 ms rule.

The stream which obeys the 20 ms rule shall carry a complete EPG. The two streams together shall also carry a complete (yet more comprehensive) EPG.

Near information is further divided into "now" and "next" (and the remainder). The "now" information is the current programme or blank. "Next" information comprises up to 4 programmes immediately after the "now" one. The network operators or service providers are obliged to transmit the This Channel "now" and "next" information with a repetition rate of at least once per every 10 seconds.

The current programme title, perhaps in a simplified form, can also be found in packet 8/30. Any EPG decoder is free to read this packet and display its information in order to guarantee the nearest programme to immediately show up.

7.1 Update Version Number

Service providers are free to make changes to the database being transmitted and put these changes on air whenever they like. However, the decoder does not know when an update has been issued or when it has to look for it. Moreover decoders may be switched off while changes are issued. In a Full EPG it may take up to 20 minutes (depending on the amount of data included) until the set has re-acquired the database. Meanwhile the user may have had a look to the old database and e.g. attempted to programme a timer. Update provides the means by which the user of a Full EPG can be informed quickly which items are subject to changes.

Updates usually are given due to late changes in programmes schedules, e.g. for a current event. They are sent very frequently in order to ensure a quick response by the decoders. Updates affect near or far information. If a near programme has been changed there is no need for any extra indication of that change. However, when a far programme is changed, an indication is given in the "Update Block" (see subclause 11.9). The update indicates which blocks of a 'complete' but older database have to be replaced. This is a means to enable the decoder to quickly correct its database.

A different situation is being covered at each transition between days. At that moment near and far information are moved one day ahead, part of the far information becomes the near information (and shall be re-allocated accordingly). In this case, or whenever the update mechanism described above is overloaded, a new "Version Number" has to be issued (causing the update table to become empty). See also subclause 6.1.

The rules:

- Update Information applies only to Full or Multiple Channels EPGs; it is optional, not a must;

- Update Information only contains references to the blocks changed;

- Update Information is to be sent with a repetition rate of once every 10 to 20 seconds. If more than 64 changes are necessary, a new Version Number shall be issued;

- Update Information is linked to a database. Whenever a new version is released, the Update Information is empty.

8 Main data groups in an EPG

The types of data used in the TV Guide can be clustered into the following data structures. The structures are defined to fit within a proposed Teletext data transmission system. The following subclause gives an overview on how the information in an EPG is structured.

8.1 Data structures used in the EPG

Note that the data structures (AI, PI, NI, OI, MI, UI, LI, TI, HI and CI) are all identical regardless of the type of EPG. They vary only in their repetition rate and quantity of data carried.

Application Information (AI) is sent most frequently. It informs the product on various characteristics of the EPG (Electronic Programme Guide). It contains:

- a provider identification;
- the number of programme information;
- general information on the networks, whose programmes are conveyed;
- a version number of the EPG;
- the total number of blocks for several data types.

Programming Information (PI) conveys parameters of programmes such as: network identification, start time, stop time, VPS / PDC code (if available), themes, language / subtitles descriptions, title and contents descriptions. It contains:

- the network;
- the start date and time, stop time and VPS/PDC label;
- themes, categories, ratings;
- subtitle, language and other information;
- a title;
- short and long textual descriptions.

Navigation Information (NI) contains a lists of events used for service specific navigation. The list is headed with a link identification. Each item in the list may contain a pointer to another list allowing menu trees to be built. The event attribute is the sorting criterion to be carried out by the product. It contains:

- a menu identification;
- the header;
- the number of events;
- string and attributes for each event.

OSD Information (OI) is used in the menus of a This Channel EPG and in the menus of a Full EPG at the end of a navigation sequence. It contains:

- a menu identification;
- sizes of the several presentation areas;
- a header;
- a message.

Message Information (MI) comprises a number of text strings to allow the service provider or network operator to display non-programme related data. It contains:

- a number of references to messages.

Update Information (UI) applies to Full EPGs only and comprises an inventory of blocks that have been changed since a new version was last issued. It contains:

- a number of references to changed programmes.

Language Information and **(Sub-) Title Information** comprise additional programme related information with respect to languages and subtitles of the programmes. They contain:

- a number of sets of languages or languages with associated page numbers, respectively.

Helper Information defines pages of the normal Teletext service which are referenced from the EPG. This helps the decoder to acquire data in advance and thus speed up its response time.

Besides the structures listed above there is an extra structure providing information on Conditional Access (CA). Its contents is beyond the scope of this ETS.

An instance of any structure is called a "block". Each block of information should be regarded as a distinct entity. The blocks are positioned in the transmission data stream in accordance with the rules laid down in ETS 300 708 [1]. On the decoder side the blocks are identified by application_id and datatype_id elements (see clauses 9 and 11).

9 Data representations in electronic info media

This clause outlines the common elements of the data structures as used in all electronic info media applications including EPG. All applications are obliged to obey this general coding, because:

- it provides the link between application and data transportation ETS 300 708 [1]; and
- existing decoders shall be able to ignore unknown application data without impact on the decoding of known applications, thus allowing new applications to be introduced in the future.

Individual data elements, or "fields", are defined and allocated a number of bits in "Syntax" tables. The fields are concatenated to form a "structure". As usual within Teletext, the concatenation starts with the LSB of the first field, which provides the LSB of the concatenated sequence, and continues till the MSB of the last field.

An instance of any structure is called a "block".

Example:

Table 4: An Example Datatype

Example_datatype	No. of Bits
field_A	3
field_B	7
field_C	6

is concatenated to:

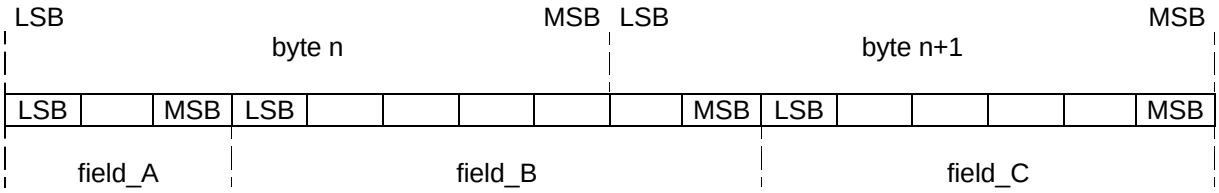


Figure 21: Bit representation of an Example Datatype

NOTE: Following concatenation, the field boundaries may not align with byte boundaries.

9.1 Syntax for the Overall Data Header

All electronic info media blocks have the format of the Overall Data Header:

Table 5: Syntax for the Overall Data Header

Overall Data Header	No. of Bits
application_id	5
block_size	11
data	variable

9.2 Semantics for the Overall Data Header

Table 6: Semantics for the Overall Data Header

Field	Semantics
application_id	is a number unambiguously linked to one application in a bundle. It is the means by which different applications are distinguished. Care shall be taken that each application within a bundle has a unique application_id. The value 0 is reserved for the Bundle Inventory (see clause 10).
block_size	gives the length of the block in bytes, excluding the predefined elements (i.e. application_id and block_size).
NOTE: application_id is identical to the Block Type as defined in ETS 300 708 [1], block_size is identical to the Block Length as defined in ETS 300 708 [1].	

10 Bundle Inventory

The first and compulsory application of any bundle is the Bundle Inventory (BI). It is intended for housekeeping purposes to assist service providers, network operators and decoders, but it does not convey any data for users. For each application (except for itself) it defines an identifier and links it to a code which identifies the application type. Applications (as e.g. EPG; see clause 11) are then referenced by their identifier.

The Bundle Inventory (BI) makes use of a single entity called the Bundle Information Structure.

10.1 Bundle Information Structure

10.1.1 Syntax for the Bundle Information Structure

Table 7: Syntax for the Bundle Information Structure

Bundle Information Structure	No. of bits
application_id = 0	5
block_size	11
no_of_applications	8
for (i=**1**;i**<=**no_of_applications;i++) {	
application_type	16
}	
checksum	8
NOTE: The application_id for the Bundle Inventory has the fixed value of 0.	

The encoding of the specific data fields (no_of_applications, application_type, and checksum) is done in the same way as described in annex A.3 for control data.

10.1.2 Semantics for the Bundle Information Structure

Table 8: Semantics for the Bundle Information Structure

Field	Semantics
no_of_applications	This is the number of applications transported with this bundle, excluding the Bundle Inventory. This field is 8 bits wide, nevertheless its value shall be less than 32, so the 3 most significant bits in this field shall always be set to 0.
application_type	The code for the application as defined in subclause 10.2. The application's application_id is assigned the current loop index (i).
checksum	This field allows extra error detection. The sum includes application_id, block_size, no_of_applications and application_type. It is calculated before application of the Hamming encoding and checked after the Hamming decoding. Checksum is calculated as follows: All mentioned data fields are concatenated and divided into nibbles in the same way as when encoding prior to transmission (see ETS 300 708 [1] and annex A). The nibbles then are expanded into 8 bit bytes with the most significant bits set to 0. These bytes are summed (modulo 256) and the checksum is 0x100 - sum of bytes.

10.2 Table of applications

Table 9: Table of Applications

Application	application_type
The EPG as defined in this ETS.	0x0000
NOTE: This table shall be amended whenever a new application is specified.	

11 EPG data types

This clause includes the syntax and semantics of the data structures used in the EPG application.

There are a number of EPG Data Structures, valid with all EPG types. The EPG data structures utilise their own general data structure, which, of course, incorporates the Overall Data Header as defined in clause 9.

11.1 EPG's general data structure

The EPG's general data structure is based on the general data structure for electronic info media (see clause 9) and conforms to the requirements of the transport mechanism as described in ETS 300 708 [1]. See also annex A.

The idea behind the EPG's general data structure is to ease the encoding and decoding of data. Generally the EPG data can be divided into two categories: control and string data. The data in these categories is treated differently before being fed into the transmission path. Control data conveying indices, times, lengths, references, etc. is Hamming 8/4 protected and string data carrying text is supplemented by a parity bit per character. Furthermore the control data fields are arranged so that the structures are easy to process rather than easy to read.

In the following syntax tables every data field is tagged with its category: a 'C' indicates control data, an 'S' string data. Refer to annex A for further details on the encoding of control and string data prior to transmission via Teletext.

11.1.1 Syntax for EPG Data Structures

Table 10: Syntax for EPG Data Structures

EPG data structure	No. of bits	Category
application_id	5	
block_size	11	
CA_mode	2	
_copyright	1	C
control_block_size	9	
checksum	8	C
datatype_id	6	C
specific_control_data	control_block_size * 8 - 14	C
specific_string_data	(block_size - control_block_size * 2 - 3) * 8	S

NOTES: Although denoted to be control data, the fields CA_mode, _copyright and control_block_size are not counted in control_block_size.
There shall be an odd number of nibbles in the control data and this is achieved by adding fill-up bits set to 0 when required.

11.1.2 Semantics for the EPG Data Structures

Table 11: Semantics for EPG Data Structures

Field	Semantics
CA_mode	indicates any access constraints applicable to the whole block. See below for details.
_copyright	indicates whether the contents of the block are subject to copyright. See below for details.
control_block_size	Indicates the quantity of control data, including any fill_up bits, within the block. Unit of measurement is (no. of control bits) / 8. Thus a value of N indicates the presence of $8 \times N$ control bits.
checksum	This field allows extra error detection. The sum includes application_id, block_size, and all fields marked as control data except itself. Checksum is calculated as follows: all mentioned data fields are concatenated and divided into nibbles in the same way as when encoding prior to transmission (see ETS 300 708 [1] and annex A). The nibbles then are expanded into 8 bit bytes with the most significant bits set to 0. These bytes are summed (modulo 256) and the checksum is 0x100 - sum of bytes.
datatype_id	is a pre-defined identifier of the current data structure (data type). Every application has its own datatype_id table. This enables the different data structures to be means to distinguished between the different data structures. The values are assigned in the table in subclause 11.12.3. Each electronic info media application has its own datatype_id table.
specific_control_data	is data exclusively used by the designated data type for control purpose and which does not convey textual information.
specific_string_data	is data exclusively used by the designated a data type for conveying textual information.
NOTE: After reception and decoding (see annex A) of a block, control data is arranged in nibbles as described. In case of successful decoding (no detectable error in control data) an extra error check can be done by following the rule for calculation of checksum and comparing the result to the checksum received.	

11.1.3 Conditional Access (CA) and copyright

An entire EPG or parts thereof may be placed under Conditional Access (CA). CA enables service providers or network operators to sell their product exclusively to customers who are prepared to pay for it. There is a data field in every EPG block conveying information about the level of CA to which this block is subject. This data field is called CA_mode throughout this ETS, and its semantics are defined as follows:

Table 12: Semantics for CA_mode

CA_mode	Semantics
00	Access is free to everyone.
01	Access is allowed only if an input code evaluates correctly.
10	Access is allowed only if the decoder's identification code evaluates correctly.
11	Access is allowed only if both an input code and the decoder's identification code evaluate correctly.

Encryption may be applied to those data fields only, which do not belong to a common data structure, i.e. those data fields which are summarised as specific_control_data or as specific_string_data in subclause 11.1.1.

NOTE: The evaluation method, the nature and source of input and decoder identification codes and any applicable encryption of EPG specific data are outside the scope of this ETS.

The contents of any block of an EPG may be copyright protected. By doing so service providers or network operators restrict the use of the contents of the block exclusively to their EPG. There is a data field in every EPG block indicating whether its contents are subject to copyright. This data field is called _copyright throughout this ETS, and its semantics are defined as follows

Table 13: Semantics for _copyright

_copyright	Semantics
0	No copyright protection on this block.
1	The entire block is subject to copyright, so that no part of the block may be used in combination with any other application, nor by any other application, nor shall it be copied to a composite EPG. It shall be used exclusively within the application to which the block belongs.

11.2 Application Information Structure

This structure provides the header of the application, it is sent very frequently.

There shall be exactly one instance of this data type in any EPG.

This structure provides the name of the EPG provider, a list of networks supported, a Version Number and several items of housekeeping data. For each network listed, a name or identifier is defined together with other network specific information.

11.2.1 Syntax for the Application Information Structure

Table 14: Syntax for the Application Information Structure

Application Information Structure	No. of Bits	Category
application_id	5	
block_size	11	
CA_mode	2	
_copyright	1	C
control_block_size	9	
checksum	8	C
datatype_id	6	C
service_name_length	5	C
no_of_networks	8	C
epg_version_number	6	C
epg_version_number_swo	6	C
time_date	32	C
EPG_version_expiry_date	6	C
no_of_navigation_info	16	C
no_of_osd_info	16	C
no_of_message_info	16	C
no_of_navigation_info_swo	16	C
no_of_osd_info_swo	16	C
no_of_message_info_swo	16	C
no_of_updates	1	C
for (j=0;j<no_of_networks;j++) {		
cni[j]	16	C
LTO[j]	8	C
no_of_days[j]	5	C
netwop_name_len[j]	5	C
default_alphabet[j]	7	C
prog_start_no[j]	16	C
prog_stop_no[j]	16	C
prog_start_no_swo[j]	16	C
prog_stop_no_swo[j]	16	C
Teletextpage[j]	11	C
}		
fill_up	0 ... 7	C
for (i=0;i<service_name_length;i++){		
text_char	8	S
}		
for (j=0;j<no_of_networks;j++) {		
for (k=0;k<netwop_name_len[j];k++) {		
text_char	8	S
}		
}		

11.2.2 Semantics for the Application Information Structure

Table 15: Semantics for the Application Information Structure

Field	Semantics
service_name_length	Defines the length of the text name the service provider wants to be called by.
no_of_networks	Defines the number of networks supported.
epg_version_number	The Version Number of that part of the EPG information which is transmitted over a stream that obeys the 20 ms rule (see subclauses 6.1 and 7.1).
epg_version_number_swo	The Version Number of that part of the EPG information which is transmitted over a stream not obeying the 20 ms rule (see subclauses 6.1 and 7.1).
time_date	The date and time of publication of this version of the EPG in Universal Time, Co-ordinated (UTC) and Modified Julian Date. This field is coded as 16 bits giving the 16 LSBs of MJD followed by 16 bits coded as 4 digit in 4 bit Binary Coded Decimal (BCD). If undefined, all bits are set to 1. Example: 93-10-13 12:45 = 0xC0791245
EPG_version_expiry_date	The date and time of expiry of the EPG defined as an offset to the date and time of publication (field time_date). The offset is measured in units of 1 hour.
no_of_navigation_info	The number of Navigation Information blocks in this application which are transmitted in a stream obeying the 20 ms rule.
no_of_osd_info	The number of OSD Information blocks in this application which are transmitted in a stream obeying the 20 ms rule.
no_of_message_info	The number of Message Information blocks in this application which are transmitted in a stream obeying the 20 ms rule.
no_of_navigation_info_swo	The number of Navigation Information blocks in this application which are transmitted in a stream that does not obey the 20 ms rule.
no_of_osd_info_swo	The number of OSD Information blocks in this application which are transmitted in a stream that does not obey the 20 ms rule.
no_of_message_info_swo	The number of Message Information blocks in this application which are transmitted in a stream that does not obey the 20 ms rule.
no_of_updates	The number of Update Information blocks comprised in this application. See also subclause 7.1.
cni	The Country Network Identifier as defined for packet 8/30 format 2 as defined in ETS 300 231 [3]. All bits set to 0 for networks without official CNI, such networks shall be identified by a netwop_name. NOTE: A network is referenced by netwop_no which is the index in the cni loop (j).
LTO	The local time offset of the network operator. This is a signed value in units of 1 / 4 of an hour.
no_of_days	The number of days programme information is provided for that network operator.
netwop_name_len	Defines the length of the text name of the network operator. The text name is compulsory only for network operators who are not allocated an official CNI, for others it is voluntary.
default_alphabet	The code for the character set to be used in title and other strings concerning this programme (see ETS 300 706 [2], table 24).
prog_start_no	The block_no of the programme information block with the earliest programme on that network in the application which is transmitted in a stream obeying the 20 ms rule.
prog_stop_no	The block_no of the programme information block with the latest programme on that network in the application which is transmitted in a stream obeying the 20 ms rule.
prog_start_no_swo	The block_no of the programme information block with the earliest programme on that network in the application which is transmitted in a stream that does not obey the 20 ms rule.
prog_stop_no_swo	The block_no of the programme information block with the latest programme on that network in the application which is transmitted in a stream that does not obey the 20 ms rule.

(continued)

Table 15 (concluded): Semantics for the Application Information Structure

Field	Semantics
service_name_length	Defines the length of the text name the service provider wants to be called by.
Teletextpage	A page number (without subcode specification) of the network's Teletext service where to find programme information.
fill_up	Don't care bits filling up the size of the control data group to an integral multiple of 8. Refer also to subclause 11.1.1 and annex A.
text_char	This is a 7 bits character set code which is placed into an 8 bits field right adjusted, the MSB is yet undefined (refer to annex A.4). The code is selected from the default alphabet unless there is a valid shift (for shifts refer to the explanation of the escape sequences in subclause 11.12.1). If there is a code < 0x20 the display shall take the same presentation as with Teletext Level 1.5, except the codes 0x0A, 0x0B, 0x18 and 0x1B which shall be replaced by 0x20.

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2 and 11.1.3.

11.3 Programme Information Structure

This structure describes the programmes. Every block contains one programme description including network, scheduled times, ratings, themes, and textual and other descriptions.

The block_no shall be sorted by start_time. Besides the block_no also the netwop_no is necessary to unambiguously define one programme information block.

11.3.1 Syntax for the Programme Information Structure

Table 16: Syntax for the Programme Information Structure

Programme Information Structure	No. of Bits	Category
application_id block_size	5 11	
CA_mode	2	C
_copyright	1	
control_block_size	9	
checksum	8	C
datatype_id	6	C
block_no	16	C
netwop_no	8	C
start_time	32	C
stop_time	16	C
escape_sequences	see subclause 11.12.1	C
title_length	8	C
_pil	20	C
no_themes	3	C
no_sortcrit	3	C
editorial_rating	3	C
parental_rating	4	C
feature_flags	8	C
background_reuse	1	C
descriptor_looplength	6	C
for (k=0;k<no_themes;k++) {		
theme	8	C
}		
for (k=0;k<no_sortcrit;k++) {		
sortcrit	8	C
}		
for (k=0;k<descriptor_looplength;k++) {		
descriptor_type	6	C
descriptor_id	6	C
descriptor_eval	8	C
}		
if (background_reuse)		
background_ref	16	C
else {		
escape_sequences	see subclause 11.12.1	C
shortinfo_length	8	C
longinfo_stringtype	3	C
if (longinfo_stringtype==0) {		
escape_sequences	see subclause 11.12.1	C
longinfo_length	8	C

(continued)

Table 16: Syntax for the Programme Information Structure (concluded)

Programme Information Structure	No. of Bits	Category
}		
else if (longinfo_stringtype==1) {		
escape_sequences	see subclause 11.12.1	C
longinfo_length	10	C
}		
else if (longinfo_stringtype==2) {		
page_reference	24	C
row_spec	5	C
col_spec	6	C
length	6	C
}		
else if (longinfo_stringtype==3) {		
page_reference	24	C
row_spec	5	C
col_spec	6	C
row_spec	5	C
col_spec	6	C
}		
else if (longinfo_stringtype==4)		
page_reference	24	C
}		
fill_up	0 ... 7	C
for (k=0;k<title_length;k++) {		
text_char	8	S
}		
for (k=0;k<shortinfo_length;k++) {		
text_char	8	S
}		
if (longinfo_stringtype==0 || longinfo_stringtype==1)		
for (k=0;k<longinfo_length;k++) {		
text_char	8	S
}		

11.3.2 Semantics for the Programme Information Structure

For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3 and 11.2.2.

Table 17: Semantics for the Programme Information Structure

Field	Semantics
block_no	This is a number which is unambiguously linked with one instance of a data structure. Blocks are enumerated starting from 0, and every next block_no is by one higher than the last one. An exception of this rule is valid for Programme Information only, in this case the start value may be different from 0, and in this case it may wrap around from 65 535 to 0 (unless this causes an ambiguity).
netwop_no	The network operator defined by the index to the table of network operators in the Application Information.
start_time	The start time of the event In Universal Time, Co-ordinated (UTC) and Modified Julian Date (MJD) (see also time_date). If undefined all bits set to 1.
stop_time	The stop time of the event in Universal Time, Co-ordinated (UTC). This field is coded as 16 bits coded as 4 digit in 4 bit Binary Coded Decimal (BCD). If undefined all bits set to 1.
escape_sequences	This field precedes the length specification of a transparent string. It is used to add special information to strings and it is explained in subclause 11.12.1.
title_length	The number of characters in the programme title. The title string itself is given at the end of the structure in a for-loop going from 0 to title_length-1.
_pil	The PDC PIL for the programme. If the network conforms to VPS then this field contains the VPS label for the programme. If neither VPS nor PDC is supported, this field shall be given an indefinite value as specified in ETS 300 231 [3].
no_themes	The number of content identifiers attached to the programme. The content identifiers themselves are found in the for-loop going from 0 to no_themes-1.
no_sortcrit	The number of sorting criteria attached to the programme. The criteria themselves are found in the for-loop going from 0 to no_sortcrit-1.
editorial_rating	See subclause 11.12.4.
parental_rating	See subclause 11.12.4.
feature_flags	A number of flags indicating ... (from LSB towards MSB): Dolby surround, PAL+, Stereo / 2 sound, black and white, ... (reserved).
background_reuse	This bit tells whether string and descriptions are taken from another Programme Information Block or defined here for the network operator defined by netwop_no.
descriptor_looplength	The length of the loop with descriptors. In essence, a descriptor is a pointer to a more or less complex description structure. See subclause 11.12.2.
theme	The content identifier. See subclause 11.12.7.
sortcrit	A sorting criterion free for coding by the service provider or network operator.
descriptor_type	This is the type of the descriptor attached to the programme. See subclause 11.12.2.
descriptor_id	This is the identification of the descriptor attached to the programme. See subclause 11.12.2.
descriptor_eval	This field is reserved for future use. See subclause 11.12.2.
background_ref	The block_no of the Programme Information Structure which comprises the shortinfo and longinfo fields for this programme.
shortinfo_length	The number of characters in the short (verbal) description of the programme. The description itself is given at the end of the structure in the for-loop going from 0 to shortinfo_lengh-1.

(continued)

Table 17 (concluded): Semantics for the Programme Information Structure

Field	Semantics
longinfo_stringtype	The type which is used to define the long (verbal) description of the programme. See also subclause 11.12.1. Only values 0, 1, 2, 3 and 4 are allowed. In case of transparent strings (stringtypes 0 or 1) the description itself is found at the end of the structure in the for-loop going from 0 to longinfo_length-1.
longinfo_length	The number of characters in the long (verbal) description of the programme. See also semantics of longinfo_stringtype.
page_reference	The number of the Teletext page, including subcode, where the long (verbal) description of the programme has to be fetched from. This is a reference to the TV programme information in the normal Teletext service of the network which is also broadcasting EPG. The format of this field is given below. See also subclause 11.12.1. In case of stringtype 2 the text has to be fetched beginning at the position indicated by row_spec and col_spec counting the number of characters (length). In case of stringtype 3 the two succeeding pairs of row_spec and col_spec indicate a rectangular area which contains the text to be fetched. In case of stringtype 4 the text is spread all over the Teletext page and the decoder shall display the entire Teletext page as part of the EPG.
row_spec	The number of the row (in a normal Teletext page referenced to by page_reference) in which the long (verbal) description of the starts or ends.
col_spec	The number of the column (in a normal Teletext page referenced to by page_reference) in which the long (verbal) description of the starts or ends.
length	The number of characters of the long (verbal) description of the programme.

Format of the field page_reference:

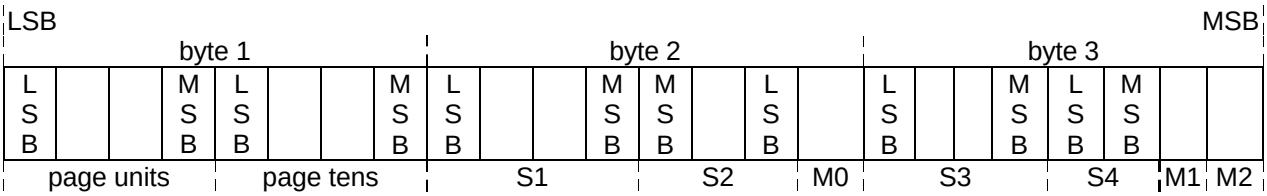


Figure 22: Transmission of EPG Data

Explanation:

The pagenumber is composed of a magazine number, page tens and page units. Page tens and units are 4 bits entities. Magazine number is composed of 3 bits marked M2, M1 and M0 with M0 as Least Significant Bit (LSB).

The subcode is composed of S4, S3, S2 and S1 with S4 as most significant part. S1 and S3 comprise 4 bits each, S2 is a 3 bits and S4 is a 2 bits entity.

11.4 Language Information Structure

This structure conveys the definitions of language descriptors. A language descriptor contains a set of languages. The different sets of languages are distinguished by means of their identification.

11.4.1 Syntax for the Language Information Structure

Table 18: Syntax for the Language Information Structure

Language Information Structure	No. of Bits	Category
application_id block_size	5 11	
CA_mode	2	C
_copyright	1	
control_block_size	9	
checksum	8	C
datatype_id	6	C
block_no	16	C
netwop_no	8	C
lang_descr_no	6	C
for (i=0;i<=lang_descr_no;i++) {		
descriptor_id	6	C
language_no	4	C
for (j=0;j<language_no;j++) {		
language	24	C
}		
}		
fill_up	0 ... 7	C

11.4.2 Semantics for the Language Information Structure

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3, 11.2.2 and 11.3.2.

Table 19: Semantics for the Language Information Structure

Field	Semantics
netwop_no	The network operator defined by the index to the table of network operators in the Application Information.
lang_descr_no	The number of language descriptors. The language information is contained in the following loop.
descriptor_id	The identification of the descriptor, which it is referenced by.
language_no	The number of languages within this descriptor.
language	A language code according to the ISO 639 norm [5].

11.5 (Sub-) Title Information Structure

This structure conveys the definitions of subtitle descriptors. A subtitle descriptor contains a set of languages with a corresponding Teletext page, where the subtitles in that language are allocated. The different sets of languages and pages are distinguished by means of their identification.

11.5.1 Syntax for the (Sub-) Title Information Structure

Table 20: Syntax for the (Sub-) Title Information Structure

(Sub-) Title Information Structure	No. of Bits	Category
application_id	5	
block_size	11	
CA_mode	2	C
_copyright	1	
control_block_size	9	
checksum	8	C
datatype_id	6	C
block_no	16	C
netwop_no	8	C
subtt_descr_no	6	C
for (i=0;i<=subtt_descr_no;i++) {		
descriptor_id	6	C
subtt_no	4	C
for (j=0;j<subtt_no;j++) {		
language	24	C
teletext_page	24	C
}		
}		
fill_up	0 ... 7	C

11.5.2 Semantics for the (Sub-) Title Information Structure

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3, 11.2.2, 11.3.2 and 11.4.2.

Table 21: Semantics for the (Sub-) Title Information Structure

Field	Semantics
subtt_descr_no	The number of subtitle descriptors. The subtitle information is contained in the following loop.
language_no	The number of subtitle within this descriptor.
teletext_page	This is the Teletext page where the subtitles in the corresponding language are found. This field is composed in exactly the same way as the fields page_reference.

11.6 Navigation Information Structure

The presentation information table provides the description of the menu structure. See also subclause 5.5. This structure provides information on the nodes in the navigation tree, defining one node per block. It gives the layout information for the navigation screens, the messages to be displayed, the links to the child nodes in the navigation tree and the attributes (criteria) which characterise the navigation path.

11.6.1 Syntax for the Navigation Information Structure

Table 22: Syntax for the Navigation Information Structure

Navigation Information Structure	No. of Bits	Category
application_id	5	
block_size	11	
CA_mode	2	
_copyright	1	C
control_block_size	9	
checksum	8	C
datatype_id	6	C
block_no	16	C
header_size	3	C
event_number	4	C
escape_sequences	see subclause 11.12.1	C
header_length	8	C
message_size	3	
message_attribute	8	
for (j=0;j<event_number;j++){		
next_link_type	2	C
next_link_id	16	C
event_attribute	see subclause 11.12.4	C
event_stringtype	3	C
if (event_stringtype==0) {		
escape_sequences	see subclause 11.12.1	C
event_length	8	C
}		
else if (event_stringtype==1) {		
escape_sequences	see subclause 11.12.1	C
event_length	10	C
}		
else if (event_stringtype==2) {		
page_reference	24	C
row_spec	5	C
col_spec	6	C
length	6	C
}		
else if (event _stringtype==3) {		
page_reference	24	C
row_spec	5	C
col_spec	6	C
row_spec	5	C
col_spec	6	C
}		
}		
fill_up	0 ... 7	C
for (k=0;k<header_length;k++)		
text_char	8	S
if (event_stringtype==0 || event_stringtype==1)		
for (k=0;k<event_length;k++) {		
text_char	8	S
}		

11.6.2 Semantics for the Navigation Information Structure

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3, 11.2.2, 11.3.2, 11.4.2 and 11.5.2.

Table 23: Semantics for the Navigation Information Structure

Field	Semantics
header_size	The number of lines for the header.
event_number	The number of events to be displayed in the menu.
message_size	The number of lines for the message area is 1+message_size.
message_attribute	Defines which message is to be displayed in the message area. See subclause 11.12.6.
next_link_type	The type of the next menu linked. See subclause 11.12.5.
next_link_id	The block_no of the next menu linked. See subclause 11.12.5.
event_attribute	The code telling the product what to do. See subclause 11.12.4.
event_stringtype	The type which is used to define the text of the event. See also subclause 11.12.1. Only values 0, 1, 2 and 3 are allowed. In case of transparent strings (stringtypes 0 or 1) the text itself is found at the end of the structure in the for-loop going from 0 to event_length-1.
header_length	In combination with the preceding escape_sequences and the for-loop going from 0 to header_length-1, this field defines a headline given as transparent string.

Refer also to subclause 5.2.

11.7 OSD Information Structure

This structure, like the Navigation Information Structure, conveys descriptions of the menus used throughout in This Channel EPGs, and in Full EPGs only at the end of the navigation sequence.

11.7.1 Syntax for the OSD Information Structure

Table 24: Syntax for the OSD Information Structure

OSD Information Structure	No. of Bits	Category
application_id block_size	5 11	
CA_mode	2	C
_copyright	1	
control_block_size	9	
checksum	8	C
datatype_id	6	C
block_no	16	C
header_size	3	C
message_size	3	C
message_attribute	8	C
if (block_no==0) {		
escape_sequences	see subclause 11.12.1	C
message_length	10	C
}		
escape_sequences	see subclause 11.12.1	C
header_length	8	C
fill_up	0 ... 7	C
for (k=0;k<header_length;k++)		
text_char	8	S
if (block_no==0)		
for (k=0;k<message_length;k++)		
text_char	8	S

11.7.2 Semantics for the OSD Information Structure

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3, 11.2.2, 11.3.2, 11.4.2, 11.5.2 and 11.6.2.

Table 25: Semantics for the OSD Information Structure

Field	Semantics
header_size	The number of lines for the header.
message_size	The number of lines for the message is 1+ message_size.
message_attribute	Defines which message is to be displayed in the message area.
message_length	In combination with the preceding escape_sequences and the for-loop going from 0 to message_length-1, this field defines a message given as transparent string. This string is in one block of OI only, therefore this block has a special meaning which is described in annex C.

Refer also to subclause 5.2.

In a composite receiver the header is defined by the manufacturer (see subclause 5.4).

11.8 Message Information Structure

This structure provides a number of publisher defined messages that need not relate to a selected event.

11.8.1 Syntax for the Message Information Structure

Table 26: Syntax for the Message Information Structure

Message Information Structure	No. of Bits	Category
application_id block_size	5 11	
CA_mode	2	C
_copyright	1	C
control_block_size	9	C
checksum	8	C
datatype_id	6	C
block_no	16	C
number_messages	8	C
for (j=0;j<number_messages;j++) {		
message_id	8	C
message_stringtype[j]	3	C
if (messag_stringtype==0) {		
escape_sequences	see subclause 11.12.1	C
message_length	8	C
}		
else if (message_stringtype==1) {		
escape_sequences	see subclause 11.12.1	C
message_length	10	C
}		
else if (message_stringtype==2) {		
page_reference	24	C
row_spec	5	C
col_spec	6	C
length	6	C
}		
else if (header_stringtype==3) {		
page_reference	24	C
row_spec	5	C
col_spec	6	C
row_spec	5	C
col_spec	6	C
}		
}		
fill_up	0 ... 7	C
if (message_stringtype==0 || message_stringtype==1)		
for (k=0;k<message_length;k++) {		
text_char	8	S
}		
NOTE: The message_id 0 is reserved for programme information (short_info).		

11.8.2 Semantics for the Message Information Structure

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3, 11.2.2, 11.3.2, 11.4.2, 11.5.2, 11.6.2 and 11.7.2.

Table 27: Semantics for the Message Information Structure

Field	Semantics
number_messages	The number of messages in this block.
message_id	A number unique for each message. See subclause 11.12.6.
message_stringtype	The type which is used to define the message text. See also subclause 11.12.1. Only values 0, 1, 2 and 3 are allowed. In case of transparent strings (stringtypes 0 or 1) the text itself is found at the end of the structure in the for-loop going from 0 to message_length-1.

11.9 Update Information Structure

In a Full EPG this structure provides the block numbers that have been changed since the release of a version. There shall be not more than one block of Update Information. See also subclause 7.1.

11.9.1 Syntax for the Update Information Structure

Table 28: Syntax for the Update Information Structure

Update Information Structure	No. of Bits	Category
application_id block_size	5 11	
CA_mode	2	C
_copyright	1	
control_block_size	9	
checksum	8	C
datatype_id	6	C
number_changes	8	C
for (i=0;i<number_changes;i++){		C
netwop_no	8	C
block_no	16	C
}		C
version_number	6	C
fill_up	4	C

11.9.2 Semantics for the Update Information Structure

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3, 11.2.2, 11.3.2, 11.4.2, 11.5.2, 11.6.2, 11.7.2 and 11.8.2.

Table 29: Semantics for the Update Information Structure

Field	Semantics
version_number	Links Updates to the epg_version_number_swo of Application Information.
number_changes	The number of block numbers following.
netwop_no	The index to the list of network operators of the Application Information.
block_no	The block_no of the affected block.
NOTE: Both block_no and netwop_no unambiguously define a "Programme Information Block" and indicate that it has been changed.	

11.10 Helper Information Structure

This structure conveys a number of Teletext pages. Any page in the list shall be used in a Teletext reference in the EPG, but the list need not be complete. The purpose is to speed up the acquisition process by allowing complex decoders to acquire the Teletext pages in a background process. The Teletext pagenumbers shall be sorted by descending priority. These structures may be broadcast relatively infrequently.

11.10.1 Syntax for the Helper Information Structure

Table 30: Syntax for the Helper Information Structure

Helper Information Structure	No. of Bits	Category
application_id block_size	5 11	
CA_mode	2	C
_copyright	1	
control_block_size	9	
checksum	8	C
datatype_id	6	C
block_no	16	C
no_of_pages[0]	6	C
for (i=0;i<=no_of_pages[0];i++)		
page_reference	24	C
no_of_pages ETS 300 708 [1]	6	C
for (i=0;i<=no_of_pages ETS 300 708 [1];i++) {		
page_identifier	4	C
page_reference	24	C
}		
fill_up	4	C

11.10.2 Semantics for the Helper Information Structure

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3, 11.2.2, 11.3.2, 11.4.2, 11.5.2, 11.6.2, 11.7.2, 11.8.2 and 11.9.2.

Table 31: Semantics for the Helper Information Structure

Field	Semantics
no_of_pages	The number of Teletext pages in the following loops. The first loop (no_of_pages [0]) defines a set of Level 1 or 1.5 pages from the normal Teletext service, which contribute text, including complete pages, to the EPG. The second loop (no_of_pages ETS 300 708 [1]) defines a set of pages which contribute text, objects and DRCS data for use within the EPG, when the equipment is capable of a Teletext Level 2.5 display or higher.
page_identifier	This value identifies the function of a page indicated, and the presentation levels at which its use is required. See table below.

Table 32: Description of the page_identifier

Value of page_identifier	Description
0x0	Basic Level 1 / 1.5 text page, required at Level 2.5 / 3.5.
0x1	Reserved for future use.
0x2	GPOP, required at Level 2.5 / 3.5.
0x3	POP, required at Level 2.5 / 3.5.
0x4	GDRCS, required at Level 2.5 / 3.5.
0x5	DRCS, required at Level 2.5 / 3.5.
0x6, 0x7	Reserved for future use.
0x8	Basic Level 1 / 1.5 text page, required at Level 3.5 only.
0x9	Reserved for future use.
0xA	GPOP, required at Level 3.5 only.
0xB	POP, required at Level 3.5 only.
0xC	GDRCS, required at Level 3.5 only.
0xD	DRCS, required at Level 3.5 only.
0xE, 0xF	Reserved for future use.

11.11 Conditional Access (CA) Information Structure

This structure delivers information concerning CA. Conditional Access Information shall be given only in applications that incorporate CA.

The semantics of the data part of the structure are outside the scope of this ETS. This ETS solely lays down a framework for such data.

11.11.1 Syntax for the Conditional Access Information Structure

Table 33: Syntax for the Conditional Access Information Structure

Conditional Access Information Structure	No. of Bits	Category
application_id block_size	5 11	
CA_mode	2	C
_copyright	1	
control_block_size	9	
datatype_id	6	C
checksum	8	C
block_no	16	C
CA_data		C
fill_up	0 ... 7	C

11.11.2 Semantics for the Conditional Access Information Structure

NOTE: For explanation of fields not included in the following list refer to the previous semantics clauses, i.e. subclauses 11.1.2, 11.1.3, 11.2.2, 11.3.2, 11.4.2, 11.5.2, 11.6.2, 11.7.2, 11.8.2, 11.9.2 and 11.10.2.

Table 34: Semantics for the Conditional Access Information Structure

Field	Semantics
CA_data	This is a privately defined data field carrying information on conditional access.

11.12 Additional information

11.12.1 Strings

Strings usually consist of a number of characters, which are concatenated to form some text which can be read by a person (transparent string). In the EPG strings are defined in a more general way, including the usual one. Here strings are also used as references to Teletext, i.e. as a pointer to part of or an entire Teletext.

String definitions as found in the syntax tables incorporate, in the most general cases, a stringtype, and dependant on this stringtype the syntax (and definition) of the string differs. So the convention of strings is enhanced by a type specifier (in the following called string_type), defining whether a transparent string is given (as usual), or if there is a reference and where to the reference is. Transparent strings are further divided into transparent short and transparent long, which distinguish in their maximum length only.

Table 35: Semantics of string_type

string_type	string_data are:
0x0	transparent short string
0x1	transparent long string
0x2	a piece of (tele-) text defined by a long page reference, row, column and length
0x3	a piece of (tele-) text in a rectangle with a long page reference
0x4	an entire Teletext page with a long page reference
0x5 ... 0x7	reserved for future expansion

In case of string_type 2 or 3 only text has to be read, i.e. the pure characters, no colours, no graphics, but as it would appear reading Teletext, i.e. including any national options selected and any applicable packet 26s. In case of string_type 0x4, however, a complete Teletext page has to be inserted as if Teletext mode was active, which means that the set actually has to do the switch to the Teletext mode for the viewer, and return from Teletext to the EPG as if the Teletext page were "part" of the EPG.

In any case the Teletext text has to be taken from the channel that carries the EPG data.

Only transparent strings (string types 0 and 1) may have "escape sequences" attached. An escape sequence gives additional information to the string, it may affect characters (in order to create language specific letters such as 'ñ' or 'ç') or give a link (include) a picture, video clip, etc. An escape sequence is made up of a pointer - indicating the location of the string to be affected - and a value - defining how to compose a character (like the packet 26s for national characters) or what to include (picture, clip,...) and where to get it from. The method of creating national characters is taken straightforward from Teletext packet 26 mechanism, with extra modes for additional purposes. The escape sequences are located in front of the length specifier.

A string field including escape sequences in a data type is generally composed as:

Table 36: General String Types Syntax

string	No. of Bits	Category
string_type	3	C
if (string_type==0 || string_type==1) {		
no_of_escapes	8	C
for (i=0;i<no_of_escapes;i++) {		
insert_pos	depending on string_type, see below	C
escape_mode	6	C
escape_data	depending on escape_mode, see below	C
}		
}		
string_data	depending on string_type, see below	depending on string_type, see below
NOTE: This table reflects the general string type definition. Data structures in the EPG application may by definition adopt specific types of strings and reject all others. In this case the structure of that "string subset" is allowed to omit unused fields of the general string type definition.		

insert_pos indicates the affected character position. The following table lays down the number of bits in this field.

Table 37: Syntax for string_type

string_type	No. of Bits in insert_pos	
0x0	8	
0x1	10	
0x2, 0x3, 0x4		not applicable
0x5, 0x6, 0x7	8	yet undefined string types

escape_mode is a mode indicator defining the use of **escape_data**.

Table 38: Syntax for escape_mode

escape_mode	No. of Bits in escape_data	Semantics of escape_data
0x00	7	Reserved for future use.
0x01	7	Select a mosaic character from the G1 set. The entry is defined by escape_data.
0x02, ..., 0x05	7	Reserved for future use.
0x06	7	Insert a local real-time clock. The format is defined by escape_data, see below.
0x07	7	Reserved for future use.
0x08	7	Latching shift to the character and national option set selected by escape_data. The shift is valid from the point of invocation up until the end of the string or until cancelled by the selection of another character set. The character codes are taken from the string.
0x09	7	Select an alphanumeric character from the current G0 set. The entry is defined by escape_data.
0x0A	7	Reserved for future use.
0x0B	7	Select an mosaic character from the G3 set. The entry is defined by escape_data.
0x0C, ..., 0xE	7	Reserved for future use.
0x0F	7	Select an alphanumeric character from the current G2 set. The entry is defined by escape_data.
0x10, ..., 0x1F	7	Add a diacritical mark to a character from the current G0 set. The diacritical mark is defined by the 4 LSBs of escape_mode referencing an entry in column 4 of the current G2 character set. The G0 character is defined by escape_data.
0x20, ..., 0x27	14	Reserved for future use.
0x28	14	Single shift to the character and national option set selected by the 7 LSBs of escape_data. The character is defined by the 7 MSBs of escape_data. The shift is valid at the point of invocation only.
0x29	14	Limited latching shift to the character and national option set selected by the 7 LSBs of escape_data. The shift is valid from the point of invocation for the number of characters given by the 7 MSBs of escape data. The character codes are taken from the string.
0x2A, ..., 0x3F	14	Reserved for future use.

NOTE: Semantics of escape_data are very much alike the mechanisms provided by packet 26 in Teletext (see ETS 300 706 [2]). If a decoder cannot display a character, it shall display a fallback or default character. The method of replacing characters shall be reasonable and sensible, and in this sense decoder manufacturers are free to provide such methods.

Table 39: Definition of Time Formats (escape_mode == 0x06)

escape_data	Format	Explanation
0x00	HH:MM	2 digits hours, colon, 2 digits minutes, e.g. " 9:30" or "09:30"
0x01	HH	2 digits hours
0x02	MM	2 digits minutes
0x03	SS	2 digits seconds
0x04, ..., 0x0F		Reserved for future use.
0x10	HH:MM:SS	2 digits hours, colon, 2 digits minutes, colon, 2 digits seconds e.g. "09:30:00"
0x11, ... 0x7F		Reserved for future use

11.12.1.1 String types

If allowed, the escape sequences as defined above, precede the string's length specified. Escape sequences are allowed with transparent strings only.

11.12.1.1.1 Transparent Short String

Table 40: Syntax for Transparent Short String (string_type == 0x0)

string_data	No. of Bits	Category
_length	8	C
for (j=0;j<_length;j++){		
text_char	8	S
}		

_length is a simple 8 bits number, giving the number of characters comprised in the transparent string.

text_char is explained in subclause 11.2.2.

11.12.1.1.2 Transparent Long String

Table 41: Syntax for Transparent Long String (string_type == 0x1)

string_data	No. of Bits	Category
_length	10	C
for (j=0;j<_length;j++){		
text_char	8	S
}		

Same as transparent short string except the range of **_length.**

text_char is explained in subclause 11.2.2.

11.12.1.1.3 Reference to a (Tele-)text string

Table 42: Syntax for Referenced String (string_type == 0x2)

string_data	No. of Bits
page_reference	24
row_spec	5
col_spec	6
length	6

col_spec is the 6 bits Teletext column number.

length is a 6 bits number defining the number of character to be read.

page_reference is defined in subclause 11.3.2.

row_spec is the 5 bits Teletext row (packet) number.

11.12.1.1.4 Reference to a (tele-)text rectangle

Table 43: Syntax for Referenced String (string_type == 0x3)

string_data	No. of Bits
page_reference	24
row_spec	5
col_spec	6
row_spec	5
col_spec	6

The two pairs of co-ordinates (row_spec, col_spec) define a rectangular area where the text has to be read from.

The fields are explained in subclause 11.12.1.1.3.

11.12.1.1.5 Reference to an entire Teletext page

Table 44: Syntax for Referenced String (string_type == 0x4)

string_data	No. of Bits
page_reference	24

page_reference is defined in subclause 11.3.2.

11.12.2 Descriptors

Descriptors are a common and efficient means to add zero or more descriptions to a programme. In case of EPG such descriptions are languages, subtitles etc. The descriptions are transmitted in dedicated datatypes (i.e. similar to a table of descriptions), and a programme that needs one of the descriptions points to its description by a) which table (language or title) and b) which entry (index).

A table is addressed by the descriptor_type. It takes the value of the datatype_id of the structure carrying the table.

The entry of the so-addressed table is identified by the descriptor_id. This identification is given in the description loop of the structure, by which it is referenced also.

The field descriptor_eval is reserved for future use.

11.12.3 DATATYPE_ID (EPG only!)

Table 45: Values for datatype_id

datatype_id	means
0x00	reserved
0x01	Application Information Structure
0x02	Programme Information Structure
0x03	Navigaton Information Structure
0x04	OSD Information Structure
0x05	Message Information Structure
0x06	Update Information Structure
0x07	Language Information Structure
0x08	(Sub-) Title Information Structure
0x09 ... 0x3D	reserved for future expansion
0x3E	Conditional Access Information Structure
0x3F	Helper Information Structure

11.12.4 EVENT_ATTRIBUTE

As introduced in subclause 5.5.3, there are several kinds of attributes, and attributes also may be combined to create a new attribute. So the field event_attribute is a data structure in itself, it is variable in length. Its length depends on:

a) what kind of attribute it is (i.e. how much additional information); and
b) how many attributes are joined together.

Table 46: Syntax for event_attribute

event_attribute	No. of Bits
number_unit_attributes	4
for (j=0;j<number_unit_attributes; j++)	
kind_of_attribute	8
attribute_info	see below
}	

Table 47: Semantics for kind_of_attribute

kind_of_attribute	Meaning	attribute_info
0x00	start time	time_code
0x01	stop time	time_code
0x02	relative date	date_offset
0x10	first program	prog_offset
0x11	last program	prog_offset
0x18	network operator	netwop_no
0x20, ... 0x27	theme (kind_of_attribute - 0x20)	theme
0x30, ... 0x37	sorting criterion (kind_of_attribute - 0x30)	sortcrit
0x40	editorial rating	editorial_rating
0x41	parental rating	parental_rating
0x50	features	feature_flags
0x58	language	language
0x59	subtitle language	subtitle_language
others	reserved for future expansion	
NOTE: Theme and sorting criterion are allowed to have multiple instances, in which case kind_of_attribute is used to enumerate these instances (0x20, 0x21, ...; 0x30, 0x31).		

Table 48: Syntax and semantics for attribute_info

attribute_info	No. of Bits	
time_code	16	Same coding as stop_time in the Programme Information Structure. See description below.
date_offset	4	0 = today, 1 = tomorrow, ... (unsigned value).
prog_offset	7	0 = current program, 1 = next program, ... (unsigned value).
theme	8	Same as in the Programme Information Structure (subclause 11.3). See subclause 11.12.7.
sortcrit	8	Same as in the Programme Information Structure (subclause 11.3).
editorial_rating	3	Same as in the Programme Information Structure (subclause 11.3).
parental_rating	4	Same as in the Programme Information Structure (subclause 11.3).
feature_flags	8	Same as in the Programme Information Structure (subclause 11.3).
language	24	ISO 639 norm, same as in the Language Information Structure (subclause 11.4).
subtitle_language	24	ISO 639 norm, same as in the Language Structure (subclause 11.4).

11.12.4.1 Attribute descriptions

Start time, stop time: this attribute is used to specify a time slot in terms of time. The set has to find out those programmes, which run within that time slot. If start time and stop time are valid (00:00 <= time <= 23:59) this refers to an absolute time slot. If no date is specified the timeslot refers to today, unless the timeslot has completely elapsed, in which case the timeslot for the next day is shown. If a relative date is specified the timeslot for that day is shown. If the start time has the value 0xFFFF the current time is taken as the start time. The end time is taken as a relative time period from the current time. If a date is specified, the schedule for that day at the current time for the specified period is shown. If no start time is specified the current time is taken as the start time and the end time is taken as an absolute time. If a date is specified, the schedule for that day at the current time for the specified period is shown. If no stop time is specified the rest of the day (until 23:59) is selected.

Relative date: This attribute is used to specify a time slot in terms of date. The set has to find out those programmes, which run within that time slot of that day.

First program, last program: this attribute is used to specify a number of programmes (to be displayed). To do so, the programmes are enumerated, starting with 0 for the current one. If no last program is specified then only one programme is selected, if no first program is specified then the first programme is the current one.

All others are defined in the same way as described in subclause 11.3.

11.12.4.2 Combining attributes

If more then one criterion is defined, the resulting criteria are combined. If multiple instances of the same criterion occur with different values an OR operation is performed. For example, when two attributes define different languages all the programs that support either one of the languages are shown. Note that this combination is not always defined, for example in the case of time-related criteria. If several different criteria occur, an AND operation is performed. For example, when language is specified as 'English' and theme DVB 1 as 'Movie', only the English-language movies are shown.

Example sortings:

Recommended programs:	editorial_rating
Own channel / set of channels:	netwop_no
German / French / Dutch spoken programs:	language
Category sorting:	theme [1] or theme [2], or own category with sortcrit
Morning / afternoon / evening:	start time and stop time with valid times
Programmes the next hour:	start time = 0xFFFF and stop time
Current / next program, next 5 programs:	first item and last item

Another example: - Live soccer game

Here the service provider uses the sorting criteria in addition to the theme coded 0x43. So guide provider will set a code in the sort criteria. This will be passed with the text "Soccer Live" in the header string (see subclause 5.5.2) which will be displayed to the viewer to select the live soccer programmes.

11.12.5 NEXT_LINK_TYPE

Table 49: Semantics for next_link_type

next_link_type	The event is to be linked to ...
0x00	undefined
0x01	the Navigation Information Structure which fulfils block_no == next_link_id
0x02	the OSD Information Structure which fulfils block_no == next_link_id
0x03	reserved for future expansion

11.12.6 MESSAGE_ATTRIBUTE

Table 50: Semantics for message_attribute

message_attribute	In the message area the product has to display ...
0x00	the short_info of Programme Information
0x01, ..., 0xFF	the message of Message Information which fulfils message_id == message_attribute

If a list of programmes is being displayed, an available short_info shall supersede the message in the message area.

11.12.7 Sorting categories

NOTE: The EBU PDC COP group has investigated several table carrying programme content / type identifications with a view to arriving at a PTY table for ETS 300 231 [3].

The most comprehensive solution, maintaining maximum commonality between DVB (ETS 300 468 [4]), PDC (ETS 300 231 [3]) and EPG requirements would be to use a form of the DVB 'Content_nibble Level 1 and 2 assignments' in which the coding range is limited from 0x0 0x0 to 0x7 0xF.

Up to now, most of the PDC-equipped videocassette recorders (VCRs) that have been sold handle a series code which is defined from 0x8 0x0 to 0xF 0xE both for programming and recording control, and therefore have problems with PTYs in the range from 0x7 0xF to 0xA 0xF.

If the programme type is expanded to 0xA 0xF the viewer will only be able to programme some PTYs, and this will undoubtedly cause an adverse reaction impacting on all network operators and manufacturers alike. Furthermore PDC programming method B used by current PDC-equipped VCRs specifies 7 bits (within packet X/26) to handle the PTY, and can therefore not support PTYs as they are currently defined in the DVB specification.

It is therefore proposed that a modification be made to the DVB 'Content_nibble Level 1 and 2 assignments' table which moves the existing categories in coding range 0x8 0x0 to 0xA 0xF to within the range 0x0 0x0 to 0x7 0xF, with the addition of code 0x3 0xF as an emergency code and 0x4 0xC as a local sports category.

In PDC the coding space 0x8 0x0 to 0xF 0xE is used to identify television programmes in the same series. It is believed that the use of such series codes is of great benefit to the viewer and network operator in the selection and recording of programmes, and we recommend its adoption by DVB and EPG.

Table 51: Theme assignments

theme	Description
0x00 ... 0x0f	undefined content
	Drama and Films
0x10	movie (general)
0x11	detective / thriller
0x12	adventure / western / war
0x13	science fiction / fantasy / horror
0x14	comedy
0x15	soap / melodrama / folklore
0x16	romance
0x17	serious / classical / religious / historical drama
0x18	adult movie
0x19 ... 0x1E	reserved for future use
0x1F	user defined
	News / Current Affairs / Social
0x20	news / current affairs (general)
0x21	news / weather report
0x22	news magazine
0x23	documentary
0x24	discussion / interview / debate
0x25	social / political issues / economics (general)
0x26	magazines / reports / documentary
0x27	economics / social advisory
0x28	remarkable people
0x29 - 0x2E	reserved for future use
0x2F	user defined
	Show / Game Show / Leisure hobbies
0x30	show / game show (general)
0x31	game / show / quiz / contest
0x32	variety show
0x33	talk show
0x34	leisure hobbies (general)
0x35	tourism / travel
0x36	handicraft
0x37	motoring
0x38	fitness and health
0x39	cooking
0x3A	advertisement / shopping
0x3B ... 0x3E	reserved for future use
0x3F	user defined
	Sports
0x40	sports (general)
0x41	special events (e.g. Olympic games, World Cup etc.)
0x42	sports magazines

(continued)

Table 51: Theme assignments (concluded)

theme	Description
0x43	football / soccer
0x44	tennis / squash
0x45	team sports / excluding football
0x46	athletics
0x47	motor sports
0x48	water sports
0x49	winter sports
0x4A	equestrian
0x4B	martial arts
0x4C	local sports
0x4D ... 0x4E	reserved for future use
0x4F	user defined
	Children / Youth / Education / Science
0x50	children's youth programmes (general)
0x51	pre-school children's programmes
0x52	entertainment programmes for 6 to 14
0x53	entertainment programmes for 10 to 16
0x54	informational / educational / school
0x55	cartoons / puppets
0x56	educational / science / factual topics (general)
0x57	nature / animals / environment
0x58	technology / natural sciences
0x59	medicine / physiology / psychology
0x5A	foreign countries / expeditions
0x5B	social / spiritual sciences
0x5C	further education
0x5D	languages
0x5E	reserved for future use
0x5F	user defined
	Music / Ballet / Dance
0x60	music / ballet / dance (general)
0x61	rock / pop
0x62	serious music / classical music
0x63	folk / traditional music
0x64	jazz
0x65	musical / opera
0x66	ballet
0x67 ... 0x6E	reserved for future use
0x6F	user defined
	Arts / Culture (without music)
0x70	Arts / Culture (without music, general)
0x71	performing arts
0x72	fine arts
0x73	religion
0x74	popular culture / traditional arts
0x75	literature
0x76	film / cinema
0x77	experimental film / video
0x78	broadcasting / press
0x79	new media
0x7A	arts / culture magazines
0x7B	fashion
0x7C ... 0x7E	reserved for future use
0x7F	user defined

Annex A (normative): Transmission and coding

A.1 Transmission format

The transmission of an EPG shall use the Page Format - Clear (as defined in ETS 300 708 [1]), and it should preferably use streams 1 and 2.

The following points are defined in ETS 300 708 [1]:

- the Teletext page number where to find the EPG (i.e. 1DF and redefinable in the MIP);

- the minimum time interval between successive pages (500 ms for streams which follow the 20 ms rule, and 200 ms for streams which need not obey the 20 ms rule).

A.2 Coding of the Overall Data Header

The coding method for the Overall Data Header, comprising application_id and block_size, is defined in ETS 300 708 [1]. It is repeated here for completeness only.

Application_id and block_size are merged in a 16 bits wide data entity, to every 4 bits (nibble) of it the Hamming 8 / 4 code according to ETS 300 706 [2] shall be applied.

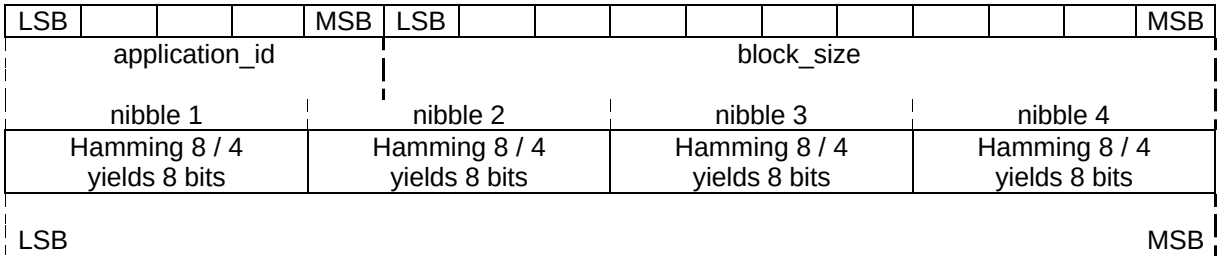


Figure 23: Structure Header as defined in ETS 300 708 [1]

A.3 Coding of Control Data

All control data (marked 'C' in the Category column in the syntax tables) shall be arranged analogously with the way described in clause 9 and in annex A.2 to form a large compound, which then shall be sliced into nibbles. (Then the checksum calculation shall take place.) Each of the nibbles shall be encoded with the Hamming 8/4 code according to ETS 300 706 [2].

A.4 Coding of String Data

All string data (marked 'S' in the Category column in the syntax tables) shall be encoded character-wise. Though the fields text_char are defined to be 8-bit character codes >0x7F are not defined. The Most Significant Bit (MSB) of any character code shall be overwritten with an odd parity bit according to ETS 300 706 [2].

Annex B (normative): Future extensions

B.1 Rules for the design of future extensions to the EPG application

Implementors of applications shall be aware that there is the possibility of additions being made to structures for future extensions to the applications. Extensions shall obey the general rules as described in clause 11, and these extensions shall always be at the end of the control data block and / or at the end of the string data block of existing structures, or new structures may be created.

Decoders not supporting such extensions shall simply discard data which is not recognised (i.e. data not specified at the time of construction of the decoders).

Annex C (normative): The use of OSD information

C.1 The use of OSD information in simple decoders

As outlined in subclause 5.5, OI is used to define the layout etc. for display of programme information, whereas NI does it for pure navigation screens.

As stated earlier, there will be different types of EPG broadcasts, and there will be different types of EPG decoders, and all possible combinations shall work. Compatibility is primarily guaranteed by a common data structure, and decoders which cannot process a special part of the protocol, shall discard it. They are able to do so for there is a block_size in every block.

A different compatibility problem is present in the case of a Full EPG broadcast received by a "simple" decoder, i.e. a decoder which cannot handle navigation or which can handle one single layout only (This Channel and Composite decoders). Such decoders shall discard NI and use the OI with block_no equal 0 only.

For service providers this means that they shall allocate OI intended for these "simple" decoders to block_no 0.

Annex D (normative): Minimum EPG broadcasts

D.1 Structures

An EPG application shall comprise the following number of instances of structures (blocks):

Table 52: Requirements for EPG Broadcast

Data Type	Required number of blocks
Application Information Structure	exactly 1
Programme Information Structure	any number
Navigation Information Structure	any number
OSD Information Structure	any number
Message Information Structure	any number
Update Information Structure	maximum 1
Language Information Structure	any number
(Sub-) Title Information Structure	any number
Conditional Access Information Structure	any number
Helper Information Structure	any number
NOTE: If there is no OI a decoder is free to present it's own style.	

Annex E (normative): Allowed string types

E.1 Structures

The following table enumerates all string types allowed in the structures. Structures not included do not comprise strings.

Table 53: String Types in EPG Data Structures

EPG Data Structure	String Types allowed
Application Information Structure	transparent with fixed maximum length only
Programme Information Structure	programme title: transparent short only short info: transparent short only long info: all types specified in subclause 11.12.1
Navigation Information Structure	header: transparent short only event: all types specified in subclause 11.12.1, except reference to a whole Teletext page
OSD Information Structure	header: transparent short only message: transparent long only
Message Information Structure	message: all types specified in subclause 11.12.1, except reference to a whole Teletext page

Annex F (normative): Ratings

F.1 Purpose, ranges and use of ratings

In this ETS two kinds of ratings are defined, editorial and parental ratings. Both serve as indicators for the degree of recommendation for the programme they are attached. Obviously recommendations are individual and (more or less) emotional decisions of their originator.

Editorial rating is a 3 bits field in PI, so its range is from 0 to 7. Where 0 means "not recommended", 7 means "maximum recommendation". On selection of a certain level of editorial rating, the decoders shall display all programmes whose editorial rating is equal or greater than that level. The value 0 shall also be used if no rating is to be attached.

Parental rating is a 4 bits field in PI, so its range is from 0 to 15. It serves as an indicator for a recommended minimum age of the viewer. The following table gives meanings for the values. On selection of a certain level of parental rating, the decoders shall display all programmes whose parental rating is equal or less than that level. The value 0 shall also be used if no rating is to be attached.

Table 54: Semantics for parental_rating

parental_rating	meaning
0	no rating or recommended for viewers of any age
1	recommended for viewers of 2 years or elder
2	recommended for viewers of 4 years or elder
3	recommended for viewers of 6 years or elder
4	recommended for viewers of 8 years or elder
5	recommended for viewers of 10 years or elder
6	recommended for viewers of 12 years or elder
7	recommended for viewers of 14 years or elder
8	recommended for viewers of 16 years or elder
9	recommended for viewers of 18 years or elder
10	recommended for viewers of 20 years or elder
11	recommended for viewers of 22 years or elder
12, 13, 14, 15	recommended for viewers of 24 years or elder, with increasing restrictions

Annex G (informative): EPG and TV information from normal Teletext service

G.1 The use of TV information from normal Teletext service within EPGs

An EPG decoder may default to display information from a normal Teletext service if none of EPG data is available.

If the Teletext service includes a Magazine Inventory Page (MIP), the page numbers used for the now / next and TV schedule pages can easily be identified.

Annex H (informative): Relationship to Teletext (Level 1.5 / 2.5, PDC, VPS, VPT)

H.1 WST and enhancement levels

The primary application in WST is the transmission and display of information. The information as transmitted is to be read directly by the viewer. Enhancements on WST have been created in the form of TOP / FLOF (better navigation) and Level 2.5 Teletext (additional display characteristics). Because the application has remained the same the relationship between the information provider, network operator and setmaker remains unchanged.

The Teletext system is now very well established in Europe, the graphic enhancements are starting to be introduced.

H.2 VPS and PDC

VPS and PDC are quite unique forms of data transmission (in the case of PDC Teletext Data Transmission). Here data is sent to a machine (usually a Video Recorder or VCR) and is never designed to be read by man. The data is interpreted by the VCR to allow accurate recordings to take place. Currently the data is used by TVs as well as VCRs to perform an automatic installation function (e.g. ATS). This is a good example of how the quality of service of a product can be increased for the user by supplying machine readable data.

The use of VPS has now become a de facto standard in the German speaking parts of Europe. PDC (an enhanced version of VPS) is now gaining widespread acceptance inside Europe. It is quite reasonable to expect it to become a de facto requirement in Video Cassette Recorders (VCRs).

H.3 Video Programming by Teletext (VPT)

VPT was an attempt to marry the traditional uses of viewer readable Teletext and the programming function of a VCR. It involves laying out or supplying specific information that allows the VCR to identify the start of the title of a programme on a normal visible Teletext and also to determine the necessary recording information to allow the VCR to accurately record the programme. The use of VPT has been a problem. The reasons are many and varied but basically can be summarised by a few observations:

- the page layout was compromised and was not attractive to the normal viewer;

- as a result many network operators did not provide it - waiting to implement PDC which provides the information in extension packets (ghost rows);

- it did not give the viewer the functionality required to make VCR programming easy and reliable.

Currently the most popular VCR programming systems make use of LCD remote controls and OSD in the VCR.

Annex I (informative): Teletext capacity required

I.1 Calculation of the required Teletext capacity of a possible EPG

There are some decoders which can only receive data from a stream which obeys the 20 ms rule. In order to deliver data to a maximum number of decoders it is recommended to put as much information as possible into that stream.

Minimum requirements are:

- near information for a "This Channel EPG";
- near information of at least 4 channels in a "Multiple Channels EPG".

It seems to be reasonable to spend 60 % of the total transmission capacity which is used for EPG for the stream which obeys the 20 ms rule, because this stream transports all near data with a relatively high repetition rate.

Assume a datarate of 16 kbits/s (the equivalent of 8 VBI lines per field) for the whole Teletext service, 10 % of this capacity for, 20 seconds cycle time for near information and 20 minutes for far information, then:

- the stream with 20 ms rule can transport 16 kbits/s $\times$ 0,1 $\times$ 0,6 $\times$ 20 s = 19,2 kbits;
- the stream without 20 ms rule can transport 16 kbits/s $\times$ 0,1 $\times$ 0,4 $\times$ 1 200 s = 768 kbits.

In this example the stream obeying the 20 ms rule can transport a Multiple Channels Near EPG with 4 networks worth of information (4 kbits data per channel) and 2 kbits for additional control structures.

Annex J (informative): Teletext transmission

J.1 Enumeration of PI blocks by block_no

As stated in subclause 11.3, PI blocks are numbered by ascending start_time. Every network's programmes allocate a continuous range of block_no. The ranges of different networks are allowed to overlap. So the PIs need not be re-numbered (after finishing a programme and after moving from one day to the next), only the border values in AI change. In case of programme insertion or cancellation the block_no of programmes later than the last unchanged one have to be adjusted.

J.2 Sequence of transmission of PI blocks

PI blocks should be fed into Teletext transmission in the sequence of ascending block_no. Exemptions are allowed for single PI blocks to be updated and for sets of PI blocks with different repetition rates.

J.3 Why two streams for EPG

The reason for allowing service providers or network operators to put parts of the EPG information into a stream which obeys the 20 ms rule and other parts into a stream without 20 ms rule is found in the different transmission capacity of these streams. The 20 ms rule places an unpleasant limitation on the frequency a Teletext page can be put into an available number of VBI lines for transmission. A rule for dividing the EPG information is given in clause 7.

Annex K (informative): Examples

K.1 Generation of a PI block

Suppose the following programme to be part of an EPG, the EPG itself free of CA and free of copyright protection, the title "Title", starting on Jan. 26, 1996, at 9:00 a.m., end at 9:30 a.m., the short info "Short Info", no long info, one user defined theme, no editorial rating, parental rating for ages 7 and up, one descriptor for a set of languages.

The following table is partly copied from the syntax table. In the rightmost column fields are separated by commas. Read this column from right to left and top down (i.e. from LSB to MSB).

Table 55: Coding a sample PI block

Field	Bits	Value	Nibbles / Bytes (binary)
application_id block_size	5 11	0x01 0x48	0001 000,0 1001 ,0000
CA_mode	2	0x0	1,0,00
_copyright	1	0x0	
control_block_size	9	0x01B	1101 0000
checksum	8	see below	0001 0100
datatype_id	6	0x02	0010 10,00
block_no	16	0x0012	0100 0000 0000 11,00
netwop_no	8	0x0B	0010 00,00
start_time	32	0x5A930900	0000 0100 0010 1100 0100 1010 0110 00,01
stop_time	16	0x0930	1100 0100 0010 00,00
escape_sequences	see subclause 11.12.1	0x00 (8 bits)	0000 01,00
title_length	8	0x05	0001 00,00
_pil	20	0xD0A40	0000 1001 0010 0100 01,11

(continued)

Table 55 (continued): Coding a sample PI block

Field	Bits	Value	Nibbles / Bytes (binary)
no_themes	3	0x1	000,0
no_sortcrit	3	0x0	
editorial_rating	3	0x0	1,000
parental_rating	4	0x3	1,001
feature_flags	8	0x01	0000 0,000
background_reuse	1	0x0	
descriptor_looplength	6	0x01	0001 11,00
for (k=0;k<no_themes;k++) {			
theme	8	0x4F	0011 01,01
}			
for (k=0;k<no_sortcrit;k++) {			
sortcrit	8	-	
}			
for (k=0;k<descriptor_looplength;k++) {			
descriptor_type	6	0x0D	0011
descriptor_id	6	0x1F	1111 00,01
descriptor_eval	8	0x00	0000 00,00
}			
if (background_reuse)			
background_ref	16	-	
else {			
escape_sequences	see subclause 11.12.1	0x00 (8 bits)	0000 10,00
shortinfo_length	8	0x0A	0010 00,00
longinfo_stringtype	3	0x0	000,0
if (longinfo_stringtype==0) {			
escape_sequences	see subclause 11.12.1	0x00 (8 bits)	0000 000,0
longinfo_length	8	0x00	0000 000,0
}			
else if (longinfo_stringtype==1) {			
escape_sequences	see subclause 11.12.1	-	
longinfo_length	10	-	
}			
else if (longinfo_stringtype==2) {			
page_reference	24	-	
row_spec	5	-	
col_spec	6	-	
length	6	-	
}			
else if (longinfo_stringtype==3) {			
page_reference	24	-	
row_spec	5	-	
col_spec	6	-	
row_spec	5	-	
col_spec	6	-	
}			

(continued)

Table 55 (concluded): Coding a sample PI block

Field	Bits	Value	Nibbles / Bytes (binary)
else if (longinfo_stringtype==4)			
page_reference	24	-	
}			
fill_up	0 ... 7	0x0 (3 bits)	
for (k=0;k<title_length;k++) {			
text_char	8	'Title'	01010100, 01101001, 01110100, 01101100, 01100101,
}			
for (k=0;k<shortinfo_length;k++) {			
text_char	8	'Short Info'	01010101, 01101000, 01101111, 01110010, 01110100, 00100000, 01001001, 01101110, 01100110, 01101111,
}			
if (longinfo_stringtype==0 || longinfo_stringtype==1)			
for (k=0;k<title_length;k++) {			
text_char	8	-	
}			

Adding up all nibbles from control data (4th column of the above table) yields checksum:

checksum = 0x100 - (0001b + 0000b + 1001b + 0000b + 1000b + 1101b + 0000b + 0010b + 1000b + 0100b + 0000b + 0000b + 1100b + 0010b + 0000b + 0000b + 0100b + 0010b + 1100b + 0100b + 1010b + 0110b + 0001b + 1100b + 0100b + 0010b + 0000b + 0000b + 0100b + 0001b + 0000b + 0000b + 1001b + 0010b + 0100b + 0111b + 0000b + 1000b + 1001b + 0000b + 0000b + 0001b + 1100b + 0011b + 0101b + 0011b + 1111b + 0001b + 0000b + 0000b + 0000b + 1000b + 0010b + 0000b + 0000b + 0000b + 0000b + 0000b + 0000b) % 0x100 = 0x2E

NOTE: In the equation for the checksum the suffix "b" indicates that the preceding number has to be interpreted as a binary value, e.g. 0110b = 0x6.

Now encoded data for this block is (all in hexadecimal notation) - already including Block Type and Block Length as defined in ETS 300 708 [1] is:

02, 15, C7, 15, D0, B6, 15, 49, FD, 49, D0, 64, 15, 15, A1, 49, 15, 15, 64, 49, A1, 64, 8C, 38, 02, A1, 64, 49, 15, 15, 64, 02, 15, 15, C7, 49, 64, 02F, 15, D0, C7, 15, 15, 02, A1, 5E, 73, 5E, EA, 02, 15, 15, 15, D0, 49, 15, 15, 15, 15, 15, 15, 54, E9, F4, EC, E5, D3, 68, EF, F2, F4, 20, 49, 6E, E6, EF.

K.2 String examples

K.2.1 Encoding 'El Niño' as transparent short string

This coding example is based on the syntax tables from subclause 11.12.1. Merging the concerned tables yields the following syntax table, which is applicable to transparent short strings only:

Table 56: Coding a sample string

string	No. of Bits	Category	Value
string_type	3	C	0x0
no_of_escapes	8	C	0x01
for (i=0;i<no_of_escapes;i++) {			
insert_pos	8	C	0x05
escape_mode	6	C	0x04
escape_data	depending on escape_mode	C	0x6E (7 bits)
}			
_length	8	C	0x07
for (j=0;j<_length;j++){			
text_char	7	S	'El Nino'
}			
NOTE: The 'ñ' is defined in the escape sequence, in string data the fallback character is defined 'n'.			

K.2.2 Teletext references

Suppose a Teletext page:

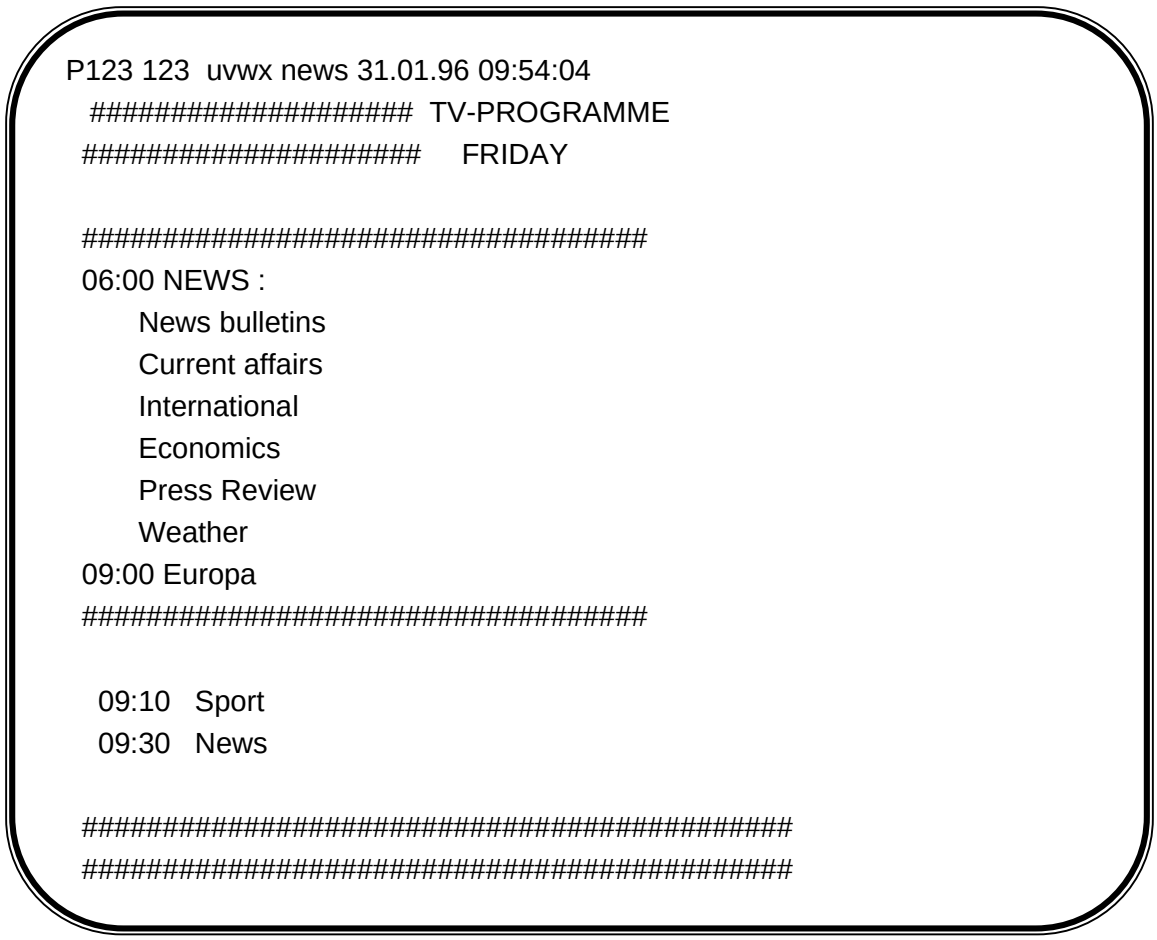


Figure 24: Illustration for a Teletext Reference

The string 'Current affairs' may be referenced to as:

- reference to a Teletext string (string type 2):

Table 57: Coding a Teletext Reference (string_type == 2)

string_data	No. of Bits	Category	Value
string_type	3	C	0x2
page_reference	24	C	0x32F7F3
row_spec	5	C	0x07
col_spec	6	C	0x09
length	6	C	0x0F

- or reference to a Teletext rectangle(string type 3):

Table 58: Coding a Teletext Reference (string_type == 3)

string_data	No. of Bits	Category	Value
string_type	3	C	0x2
page_reference	24	C	0x32F7F3
row_spec	5	C	0x07
col_spec	6	C	0x09
row_spec	5	C	0x07
col_spec	6	C	0x17
NOTE: References do not include escape sequences, since text has to be inserted as it would read in Teletext - this does include any applicable packet 26.			

K.3 Examples for attributes in a menu organisation

Consider the menu organisation illustrated in subclause 5.5.1. It is constructed of three navigation levels, the first level selection is "Themes", then "Sport" and at last "Golf". The fourth level in the example is a list of programmes. As defined, these first three levels are represented by NI blocks, and the fourth one by an OI block.

Attributes appear in the second and third level. The second level introduces a "Sport" attribute, the third level introduces the "Golf" attribute. Only the third level is immediately followed by a programme list (OI). At the end of the navigation path the decoder searches for all programmes, which have a theme attribute equal "Golf".

Supposed the third level includes not only the "Golf", but also a "Cricket" attribute. The corresponding NI block then in the field event_attribute comprises number_unit_attributes=2 and a loop with two kind_of_attribute fields with values 0x20 and 0x21 (if theme is used). Since there are two attributes of the same kind (theme), the decoder responds by finding out all programmes, which have a theme attribute equal "Golf" (logical) OR equal "Cricket". So the programme list at the end of the navigation tree will contain Golf programmes as well as Cricket programmes.

Suppose a different selection in a menu comprising a language attribute "English" and an editorial rating attribute of e.g. 5. Now there are different kinds of attributes, so the decoder will find programmes, which have an editorial rating greater or equal 5 (logically) AND a language attribute equal "English". The programme list in this case shows all programmes with English sound, whose editorial rating is at least 5.

NOTE 1: A language attribute (in NI) refers to any language descriptor including that language. A subtitle_language attribute (in NI) refers to any (sub-) title descriptor including that language.

NOTE 2: Combined attributes, like in the rating and language example, may be established by transmission (the service provider offers that selection), or it may be created by user interaction in the decoder only - although this would require a rather complicated user interface.

K.4 Model of a coding procedure

The following steps show how a data transmission can be constructed from data blocks. A decoder, of course, has to follow the reverse procedure:

- The control bits are concatenated into a single sequence. The LSB of the first field provides the LSB of this sequence, and the MSB of the last field provides the MSB;

- If required, fill_up bits get added, starting after the MSB of this sequence, to make the total number of bits in the sequence a multiple of 8;

- This control bit sequence is partitioned into 4 bits nibbles from which 8 bits bytes are formed by setting the 4 MSBs to 0;

- The checksum is calculated by summing the control bytes and string bytes (modulo 256) and subtracting the result from 256;

- The 8 checksum bits are inserted within the control bit sequence. This is partitioned into 4 bits nibbles and Hamming 8/4 encoded, yielding 8 bits bytes ready for transmission;

- Each character of a string is odd parity encoded. The parity bit is added as the MSB to form an 8 bits byte ready for transmission;

- The bytes are mapped to Teletext and transmitted as described in ETS 300 708 [1].

Annex L (informative): Implementation of an EPG prototype

L.1 General

A prototype of the Electronic Program Guide implementation may be represented by instances of four separate entities.

L1.1 Generator system

The generator system which maintains the database of EPG information. This system knows which TV programmes are to be transmitted and when, it knows which type of EPG it is to produce, it knows which data to transmit on a stream obeying the 20 ms rule and which to transmit on a stream without the 20 ms rule, and it incorporates a regular update strategy.

The generator system needs to generate application data according to this ETS, and it communicates with the transmission system using a protocol defined in subclause L.2 of this annex.

L1.2 Transmission system

The transmission system receives a bundle of data from the generator system, formats it for transmission, adds error detection / correction and transmits it. The data may be multiplexed with a conventional Teletext service. The transmission system need not have any knowledge of the meaning of the data it is transmitting beyond basic format knowledge (the knowledge of the structure as defined in subclause 9.1, annex A, and in ETS 300 708 [1]).

The transmission system communicates with the generator system using a protocol defined in subclause L.2 of this annex, and it generates Teletext pages.

L1.3 Receiver system

The receiver system receives Teletext and decodes a data bundle. It corrects errors if necessary (and if possible) and passes the data on to the next stage. It need not have any knowledge of the meaning or structure of the data beyond basic format knowledge (the knowledge of the structure as defined in subclause 9.1, annex A, and in ETS 300 708 [1]).

The receiver system receives Teletext pages and passes data to the application system according to a protocol defined in this annex.

L1.4 Application system

The application system handles the data received from the receiver system and displays it under control of the viewer. The application shall not expect to receive all data correctly, it shall expect errors in the data and formulate the best display it can.

The application system receives data according to a protocol defined in this annex and processes the contents of the data according to this ETS.

NOTE: This is an idealised view of the system. Domestic EPG equipment is unlikely to be implemented in this manner because of cost / capability constraints. It is the responsibility of the developers of this equipment to ensure that the receiver / application interface (however implemented) permits this view.

L.2 Protocol for data communication

This subclause describes one implementation of a serial communications protocol between a generator system and a transmitter system. It may also be possible to use the same protocol between the generator system and an application system for purposes of demonstration and validation where the application can be simulated on a general purpose computer such as a PC.

Moreover, this subclause includes an extension to define the method of creating data files which allow generation of the EPG from within a Softel transmission system to allow testing of the transmission path (transmission system to receiver system) and (transmission system to receiver system to application system).

In the real world there may be more than one of each of the entities described. Transmission systems will be broadcasting to a large number of receivers, receivers may receive data from a number of transmission systems. Generally there will be a one-to-one correspondence between a generator system and a transmission system and a one-to-one correspondence between a receiver system and an application system.

Communications will be via RS-232 into a serial port.

The following parameters can be defined by the transmission system operator :

- Line speed 9 600 or 19 200 bits/s;

- Number of start / stop bits;

- Hardware (RTS / CTS) or software (XON / XOF) flow control from transmission system to receiver system.

The following parameters are fixed :

- Eight bits data - no parity;

- No flow control from receiver system to transmission system (the transmission system is always free to send).

  NOTE: The RS-232 link is assumed to be local and error-free, in the case of a poor line a different protocol should be used.

Data is transferred from the generator system to the transmission system in blocks. Each block begins with a synchronisation DLE (0X10) character followed by an input control character.

- The input control character has the following structure;

- The 7 LSBs of the input control character contain the operation required, the MSB contains a stream indicator - if set to 0 the data is relevant to stream 1, otherwise it is for stream 2.

Any value less than 32 (0X20) of the 7 LSBs is an application_id value, and the data for an EPG block follows. A value of @ 0X40 causes the contents of the current page (if any) to be immediately queued for transmission.

Application blocks (within this protocol) begin with a block synchronisation DLE character, followed by an application_id with stream number (5 LSBs b + 1 MSB).

The block length (as defined in ETS 300 708 [1]) follows, this is the length (in bytes) of the remainder of the structure. It is sent as two bytes of 8 bits and 3 bits in that order.

The next two bytes are a compound item, the first two bits of the first byte contain the access control bits, the next bit contains the copyright bit, the remaining 5 bits in the first byte contain the 5 LSBs of the control block size, the first 4 bits of the next byte contain the 4 MSBs of the control block size. The 4 MSBs of this byte should be set to the first 4 bits of the control_block_data (in the EPG application this will be the 4 LSBs of the datatype_id).

The next group of bytes contain the remainder of the control block data. This is data which will be coded for transmission as 4 / 8 bits Hamming, for this protocol they are transmitted as pairs of nibbles within each byte, the first nibble is in the 4 LSBs of the byte. The 4 MSBs of the last byte will always be discarded. Any other unused bits in the last two bytes should be set to 0.

The last group of bytes is the string data which contains the string characters to be transmitted. Each character is sent as the 7 LSBs of a byte, the MSB is set to 0. The transmission software will add odd parity to each character.

Syntax of block protocol

Table 59: The Serial Protocol

Field
Synchronisation DLE char. (8 bits)
Application_id b + stream no. (5 bits b + 1 bit)
Block_size (8 LSBs)
Block_size (3 MSBs) b + 5 bits set to zero.
Copyright, Access_control and Control_block_size
Access_control (2 bits)
Copyright (1 bit)
Control_block_size (5 bits)
Control_block_size (4 bits) b + First 4 bits of Control_block_data.
Control_block_data remainder (4 bits b + 4 bits). 4 MSBs of last byte discarded.
String_data (7 bits b + 1 bit set to 0).

The contents and structure of the Control_block_data and String_data are as defined in the main body of this document and are not material to this protocol.

Responses

The Generator software should at all times be ready to receive and process the following data :

- XON (for software flow control - if enabled);
- XOF (for software flow control - if enabled);
- BEL - implies that flow control has been ignored - data has been lost (Perhaps the transmission software is not running);
- ACK - a correctly structured block has been received and will be processed for transmission;
- NAK - an incorrectly structured block has been received:

    - DLE not present when expected;
    - Unrecognised input control character;
    - Invalid block_size;
    - Invalid control_block_size.

- ESC - a correctly block has been received, however it was sent to stream 2 and this system is not enabled for stream 2 transmission. It will be sent on stream 1;

- Any character received other than those above implies that the transmission software is not running or that the port characteristics are not compatible.

The generator software need not wait for an ACK or NAK character before commencing the send of the next structure. In a correctly running system there will be as many ACK characters as structures sent, however in the case of errors there may be as many as one NAK per character sent as the transmission system attempts to synchronise.

Data files

For prototyping purposes it will be possible to generate data locally on the transmission system, this will be done by generating ASCII based data files rather than feeding data into an RS-232 port. The files generated by this process should contain only text information which will allow the contents to be edited using a conventional text editor, thus testing can be easily repeated and modified. Non-visible characters such as <LF> or <CR> are ignored and should be used to make the file more readable. Data files may contain comments to allow a description of the test data to be carried with the data itself.

If a character which is not a normal text character is to be inserted into the data then it is input as two hexadecimal digits preceded by a "#" character. Any 8 bits value can be input in this manner for example #00 or #FF.

When the end of the file is encountered, the software implicitly adds a DLE @ block to cause the contents of the current page to be transmitted (if any).

The maximum length of any one line of text is 128 characters, this does not restrict the contents of any output as the <CR><LF> characters will be ignored and processing begins at the start of the next text line. Structures may be split over two or more lines, however input of 8 bits values #dd form cannot.

Any data from an exclamation character "!" to the end of line (including any "!" characters) is considered as a comment and will be discarded. An exclamation character "!" can be input as #21, a "#" character can be input as #23.

The transmission system protocol will add odd parity to all characters in the string block which are not input using the #dd construct. Note that this applies only to data file input.

L.3 Example of a Bundle Information Block in the Serial Protocol

Table 60: A Sample Bundle Information Block in the Serial Protocol

Data	Explanation
DLE	Synchronisation
0X00	Application 0 - Stream 1
0X0A	Block_size 8 LSBs
0X00	Block_size 3 MSBs, 5 bits set to 0
0X50	Access_info b + Copyright_info b + Control_block_size 5 LSBs
0XF0	Control_block_size 4 MSBs b + 4 bits checksum
0X10	Checksum
0X00	No of applications b + application 1 type (4 LSBs)
0X00	Application 1 type (mid 8 bits)
0X10	Application 1 type (4 MSBs) b + app 2 type (4 LSBs)
0X00	Application 2 type (mid 8 bits)
0X20	Application 2 type (4 MSBs) b + app 3 type (4 LSBs)
0X00	Application 3 type (mid 8 bits)
0X00	Application 3 type (4 MSBs) b + filler 4 bits

Total number of bytes sent = Block_size b + 4.

Number of bytes in control block = Control_block_size:

- Calculated as:
  - $(((4^{th}$ byte & 0XFC) >> 2) | $((5^{th}$ byte & 0X0F) << 5)) = 10

NOTE: In this example there is no string data, if there were the number of characters in the string data would be (Block_size - Control_block_size).

Data file example

The data file to match the above example would be:-

#10#00#0A#00#50#F0#10#00#00#10#00#20#00#00#10@

L.4 Conversion of PI start_times to local times

The time of a programme to start - in terms of a broadcasting station's local time - is calculated by adding the network's LTO (from AI) to start_time (from PI).

A viewer living in a different time zone than a broadcasting station will like to have programme start times in terms of his local time. An EPG decoder may replace AI's LTO so as to display programme start and stop times in terms of the viewers local time.

If the viewer's local time offset with respect to UTC is known, the EPG decoder may use this time offset instead of LTO.

The LTO replacement can be discovered automatically by investigation of the Teletext service of a network operating in the time zone of the searched broadcasting station and investigation of the Teletext service of a network broadcasting in the receiver's time zone. Both Teletext services are assumed to convey their local time in the Teletext page header. Now the difference of these times shall be added to LTO so as to yield the viewer's time offset with respect to UTC.

History

Document history			
June 1996	Public Enquiry	PE 107:	1996-06-03 to 1996-09-27