



Matlab-based GSE - Generic Stream Encapsulation- Packet Generation User Interface UI: DVB-S2/S2X IP Communications

Sid Ahmed Chouakri

Telecommunications and Digital Signal Processing Laboratory,
University Djillali Liabes of Sidi Bel Abbes, Algeria

Abstract

The present work constitutes an educational and technological solution based on the interaction between DVB-S2 protocol (Digital Video Broadcasting via Satellite 2 and generation) and IP (Internet Protocol), for the generation of GSE (generic stream encapsulation) packets. This work includes a graphical user interface (GUI) based on MATLAB that allows users to interactively generate GSE packets designed for DVB-S2 IP transmission. This tool enables the creation of packets compliant with the ETSI TS 102 606 standard "Digital Video Broadcasting (DVB); Generic Stream Encapsulation (GSE) Protocol" for satellite IP transmission. The user interface facilitates the configuration of multiple parameters, such as: - Packet type selection: choose between full packet data unit (PDU) packets or fragmented packets. - Addressing options to select between 3-byte or 6-byte MAC addresses. - Fragmentation parameters: fields to specify padding bytes, fragment identifiers, and fragmentation lengths. The tool includes a user-friendly interface for data entry and displays the resulting GSE packets along with their lengths. It is compatible with MATLAB R2024b and later versions. The code is available on MATLAB File Exchange GSE -Generic Stream Encapsulation- Packet Generation V.1.1.0 - File Exchange - MATLAB Central. The user interface consists of 3 drop-down lists, which would be executed in series, namely: the main one "PADDING, PDU, RESET", "NO-FRAGM (meaning complete PDU packet), FRGMNT (to generate GSE packets from a fragmented PDU packet)", and "MAC 6 BYTES, MAC 3 BYTES" (to choose a 6- or 3-byte NPA addresses).

Keywords: Matlab, User Interface UI, GSE -Generic Stream Encapsulation-, DVB-S2, IP Communications

1. Introduction

To provide a quality experience to consumers, the founders of DVB-S introduced DVB-IP (Digital Video Broadcasting-Internet Protocol) for television broadcasting and video streaming (DVB, 2025). DVB-IP (ETSI, 2007) is a technology for transmitting video signals over an Internet Protocol (IP) network. It allows users to enjoy clear, noise-free video over networks

not specifically designed for video streaming. DVB-IP (Lanigan, 2021) allows video to be delivered over a wired, wireless, or IP network (Levil & Moulin, 2023). The main advantages of DVB-IP are its compatibility, low cost, ability to transmit digital data, and ability to integrate video and audio streams into existing networks (ETSI, 1997, A). DVB-IP is now widely used to deliver audio and video content over IP networks, and its applications are expanding rapidly (ETSI, 1997, B). The first-generation DVB system (DVB-S/T/C) also allows the transmission of variable-size audiovisual (A/V) data packets and packet data units (PDUs) encapsulated in fixed-size 188-byte "MPEG-TS" packets (ETSI, 1998). Two encapsulation methods are used in the various DVB-S, DVB-T and DVB-C transmission systems (Fairhurst & Collini-Nocker, 2005): MPEG-encoded and compressed A/V data are usually transmitted as packet elementary streams (PES) via TS, but can also be transmitted as IP datagrams using the real-time protocol, encapsulated by multiprotocol encapsulation (MPE) or unidirectional lightweight encapsulation (ULE) via TS (Collini-Nocker & Fairhurst, 2004). Second generation DVB broadcast networks are part of various packet transmission networks such as MPEG-X and its different versions (2, 4, 7, 21...) as well as with the arrival of a new link layer adaptation protocol (Cantillo et al., 2008), the GSE protocol, a new competitor to the MPE and ULE protocols (Ewald-Arostegui et al., 2010), which allows the transmission of IP and MPLS packets (Fairhurst & Collini-Nocker, 2008).

In 2011, the European Telecommunications Standards Institute (ETSI) defined a new standard: GSE (Generic Stream Encapsulation) is a link layer protocol for data encapsulation that has gradually replaced the MPE and ULE protocols in the field of IP (Internet Protocol) data broadcasting over satellite broadcast networks, such as DVB-S2 (ETSI, 2014). Unlike MPE/ULE designed for video broadcasting only, GSE allows the transport of any type of IP traffic (voice, data, video, etc.) bidirectionally. It standardizes the encapsulation of any type of traffic over broadcast networks, maximizing opportunities for new interactive services. Encapsulation with a GSE frame allows data packets to be efficiently grouped, minimizing encapsulation overhead and optimizing the use of available bandwidth. This encapsulation also facilitates data flow management and provides flexibility in packet transmission over different types of networks, including satellite communication networks. The PDU can be encapsulated in a single GSE packet or divided into PDU segments and then encapsulated in multiple GSE packets. GSE packets are typically of variable length to handle incoming IP traffic with minimal overhead.

2. Motivation

Faced with the lack of testing and simulation software dedicated to a particular standard/system, the proposed user interface was developed. This development was also motivated by the need of students and/or software developers to implement, test and evaluate their algorithms and applications intended for a particular standard. The developed interface is mainly intended to assist developers of DVB (Digital Video Broadcasting) over IP systems. It allows users to interactively generate the GSE (Generic Stream Encapsulation) packets designed for DVB-S2 (DVB Visa Satellite 2nd generation) IP transmission.

The user interface allows you to easily configure several settings, such as:

- Packet type selection: Choose between full Packet Data Unit (PDU) packets or fragmented packets.
- Addressing options to choose between 3- or 6-byte MAC addresses.
- Fragmentation parameters: fields for specifying padding bytes, fragment IDs and fragmentation lengths.

The tool includes a user-friendly interface for data entry and displays the resulting GSE packets along with their lengths.

The code is available as a freely downloadable software package on the MATLAB File Exchange (Chouakri, 2024a). It is compatible with MATLAB R2024b and later versions.

Furthermore, the proposed tool is, to our knowledge, the only MATLAB application with a friendly user interface dedicated to the generation of GSE packages distributed via a mainstream academic platform, such as MATLAB File Exchange (Chouakri, 2026).

3. Methods

Two complementary algorithms for GSE packet generation are implemented, that are: CHOUAKRI_KERNEL_GSE_PCKT for complete PDUs and CHOUAKRI_FRAGMT_GSE for fragmented PDUs. These algorithms follow ETSI TS 102 606 specifications and are integrated into a MATLAB-based user interface. The interface allows users to configure packet type, addressing scheme (3- or 6-byte MAC), and fragmentation parameters.

3.1 GSE DVB-S2 IP-based Encapsulation

Generic Stream Encapsulation (GSE) protocol was designed as an efficient, lightweight, and extensible encapsulation mechanism for GS transmission over DVB-S2 broadcast links.

Figure 1 illustrates the overall flowchart of GSE DVB-S2 IP-based encapsulation according to Figure 2.

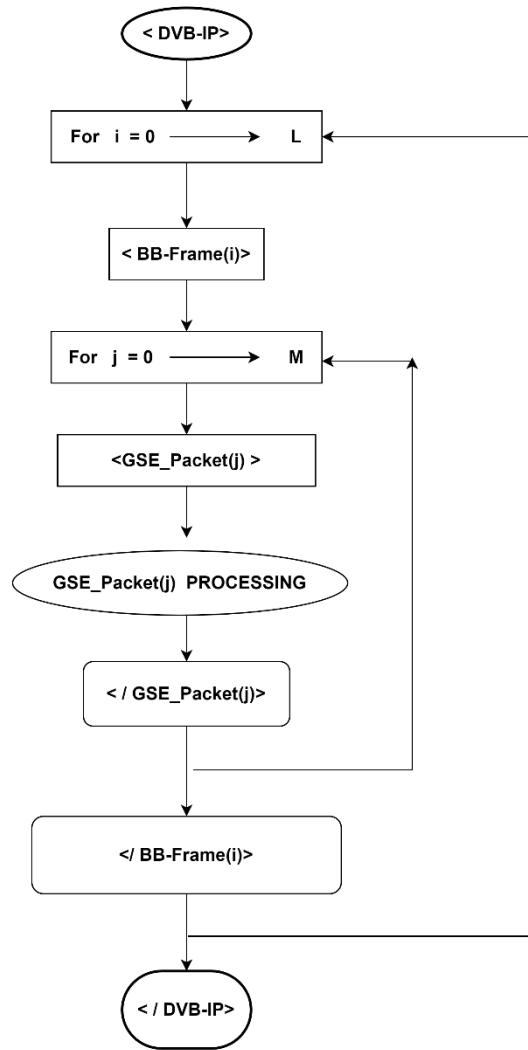


Figure 1: General flowchart of GSE IP-based DVB-S2 encapsulation

Figure 1 summarizes a general DVB-IP flow scenario which consists of a sequence of BB-Frame baseband frames into each of which a number of GSE M packets (numbered L and M respectively in the flowchart) are multiplexed and allocated.

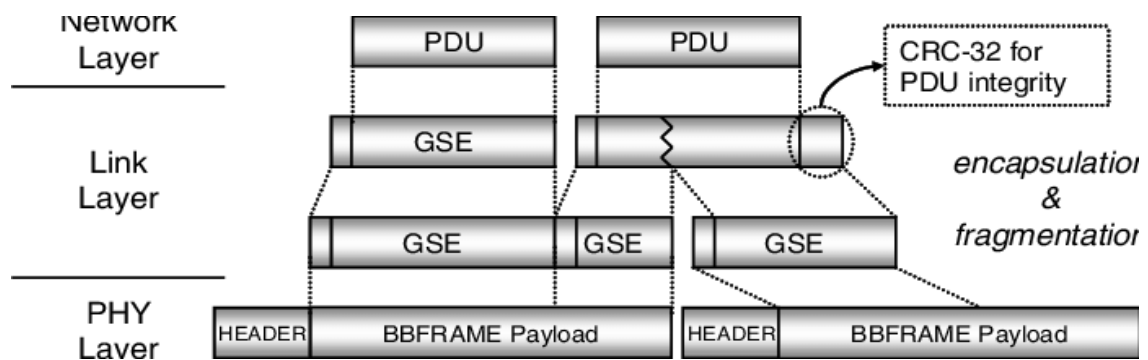


Figure 2: Illustration of the protocol stack for the GSE layer PADDING VS, PACKING (Papaleo et al., 2008)

3.2 GSE Encapsulation Flowchart

In this context, we developed a GSE packet generation algorithm for the case of “Type 2: EtherType compatible type fields” (See Figure 3).

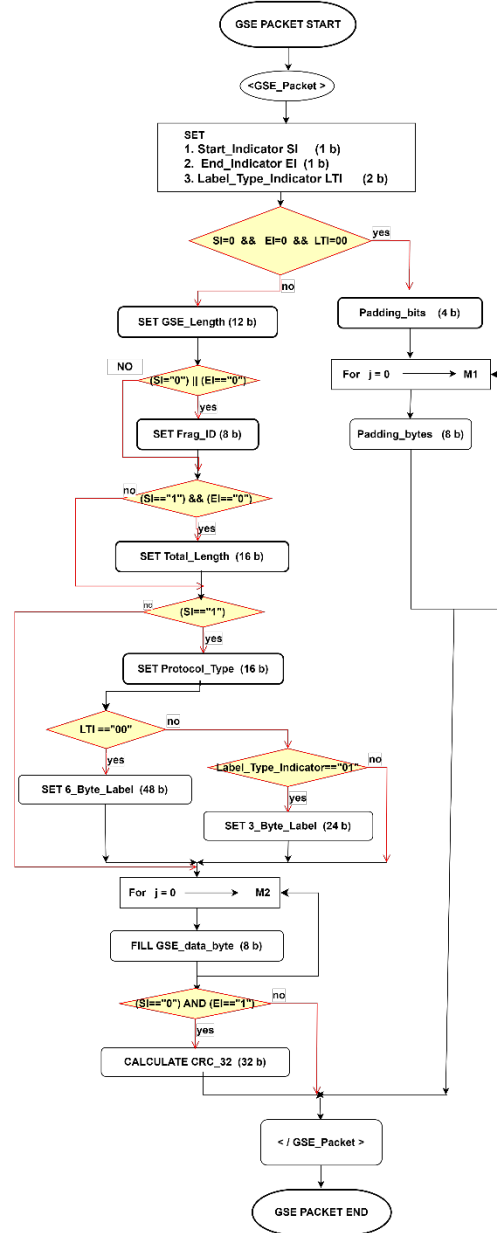


Figure 3: Flowchart of GSE syntax

3.3 The Two Complementary GSE Encapsulation Algorithms

Our basic approach consisted in developing two complementary procedures: the first one called 'CHOUAKRI_KERNEL_GSE_PCKT' (Chouakri, 2024d) which creates the specified GSE package for each encapsulation case as illustrated in Figure 4; while the second one handles the fragmentation case by calling the procedure 'CHOUAKRI_FRAGMT_GSE' for each fragmented case (Chouakri, 2024c): first, intermediate and last packet (See Figure 4).

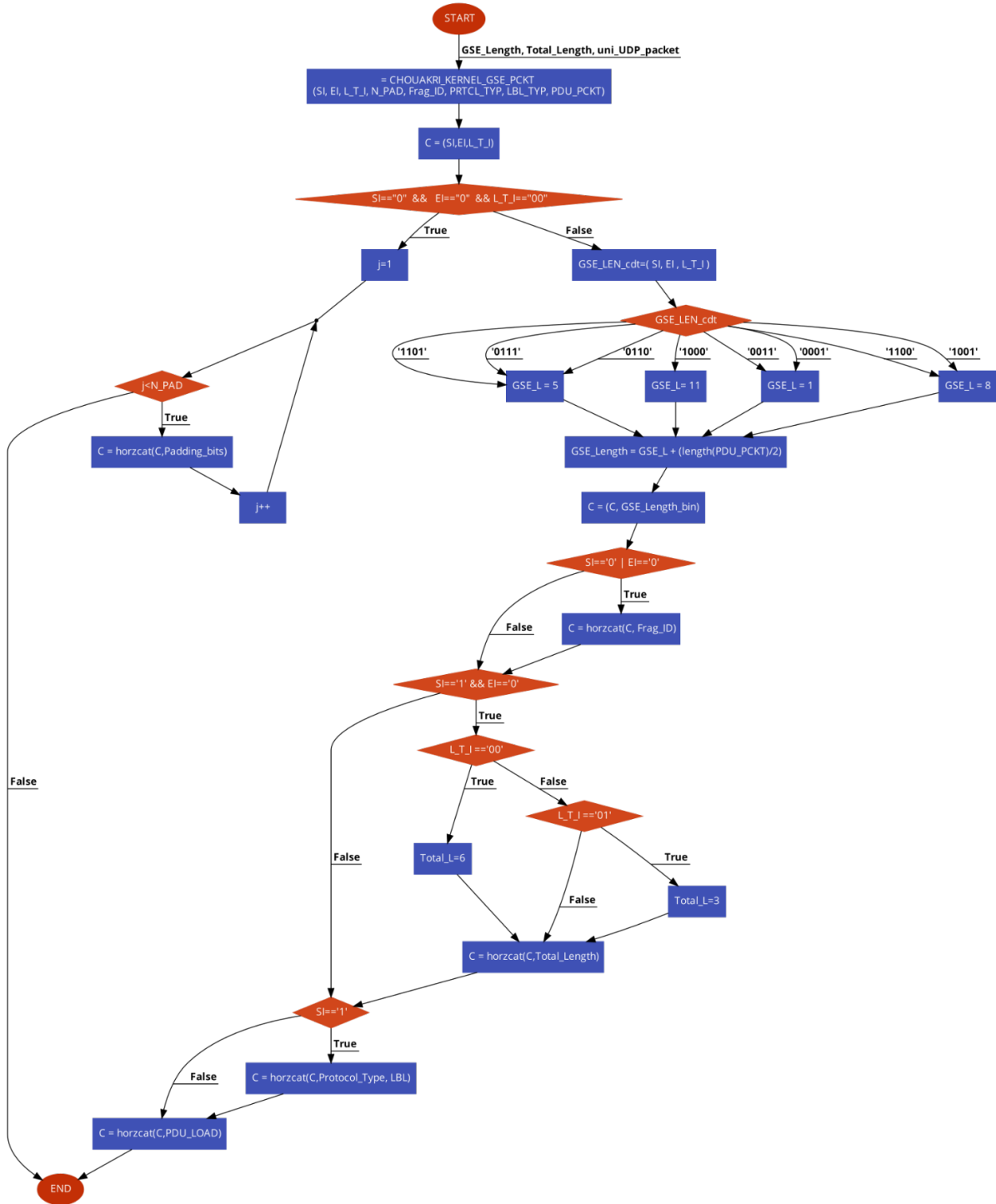


Figure 4: Flowchart for generating "CHOUAKRI_KERNEL_GSE_PCKT"

Figure 4 illustrates the flowchart of the procedure 'CHOUAKRI_KERNEL_GSE_PCKT'. The procedure mainly reads the start and end indexes and the label type indicator L_T_I , and even arbitrarily, if not required, the number of padding bytes, fragmentation identification, protocol type, label type, and PDU packet (SI , EI , L_T_I , N_PAD , $Frag_ID$, $PRTCL_TYP$, LBL_TYP , PDU_PCKT , respectively, listed in the flowchart). The procedure generates the GSE packet as well as the GSE packet length and the total GSE length. The length of the GSE packet (in bytes) is equal to the sum of the length of the input PDU packet (in bytes) and a parameter, called "GSE_L" in the flowchart, which denotes the distance (in bytes) between the "GSE Length" and "PDU Data" fields. The "GSE_L" parameter can take four different values:

1, 5, 8 and 11, depending on the SI, EI and L_T_I bits, as shown in Figure 5, which shows examples of the types of GSE packets generated.

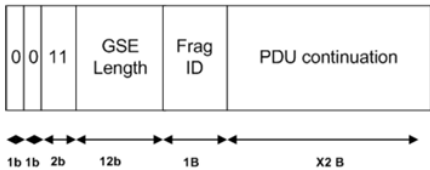
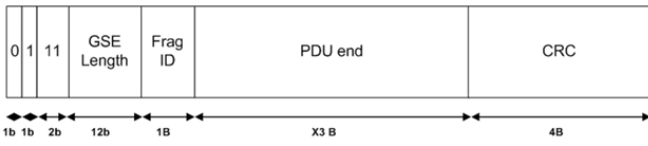
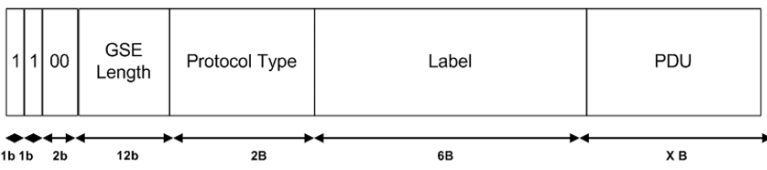
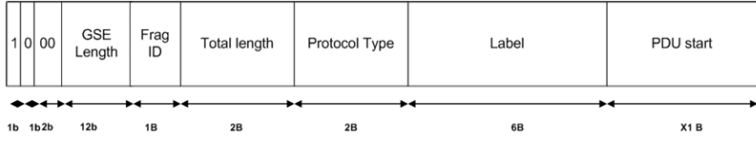
Example of generated GSE packet type	'GSE L'
	1
	5
	8
	11

Figure 5: GSE packet type depending on the "GSE_L" value

Then, if at least SI or EI is 0, the PDU packet is considered fragmented and the "Frag_ID" is inserted into the GSE packet. Otherwise, the complete PDU is encapsulated into the GSE packet. Depending on the 2-bit value "L_T_I" ("00" or "01"), a 3-byte or 6-byte NPA address is considered respectively. It is important to note that the total length of the GSE packet "Total_Length" is calculated and inserted in the unique case of the beginning of the fragmented PDU packet. It is also worth noting that, since this "GSE packet generation kernel" procedure is mainly aimed at generating the complete, unfragmented PDU packet, the CRC-32 field is not calculated.

The 'CHOUAKRI_FRAGMT_GSE' flowchart reads, in addition to those of the 'GSE packet generation kernel', the lengths of the first PDU fragments - the PDU fragment length will be deduced from the total length of the original PDU packet and generates the fragmented GSE packets including the start, middle and last packets.

First, the procedure calls the procedure 'CHOUAKRI_KERNEL_GSE_PCKT' with SI='1' and EI='0' referring to the first GSE packet of the fragmented PDU packet. Then, to generate the intermediate GSE packets of the original PDU packet, the procedure 'GSE packet generation kernel', with SI='0', EI='0' and 'L_T_I'='11' referring to the 'reuse' operation, is called for each fragment with its own length. Finally, for the final GSE packet, the procedure calls the procedure 'GSE packet generation kernel' with SI='0' and EI='1' and always 'L_T_I'='11'. At this phase, the CRC-32 checksum is generated and added to the end of the last GSE packet.

The 'CHOUAKRI_FRAGMT_GSE' flowchart reads, in addition to those of the 'GSE packet generation kernel', the lengths of the first PDU fragments - the PDU fragment length will be

deduced from the total length of the original PDU packet and generates the fragmented GSE packets including the start, middle and last packets.

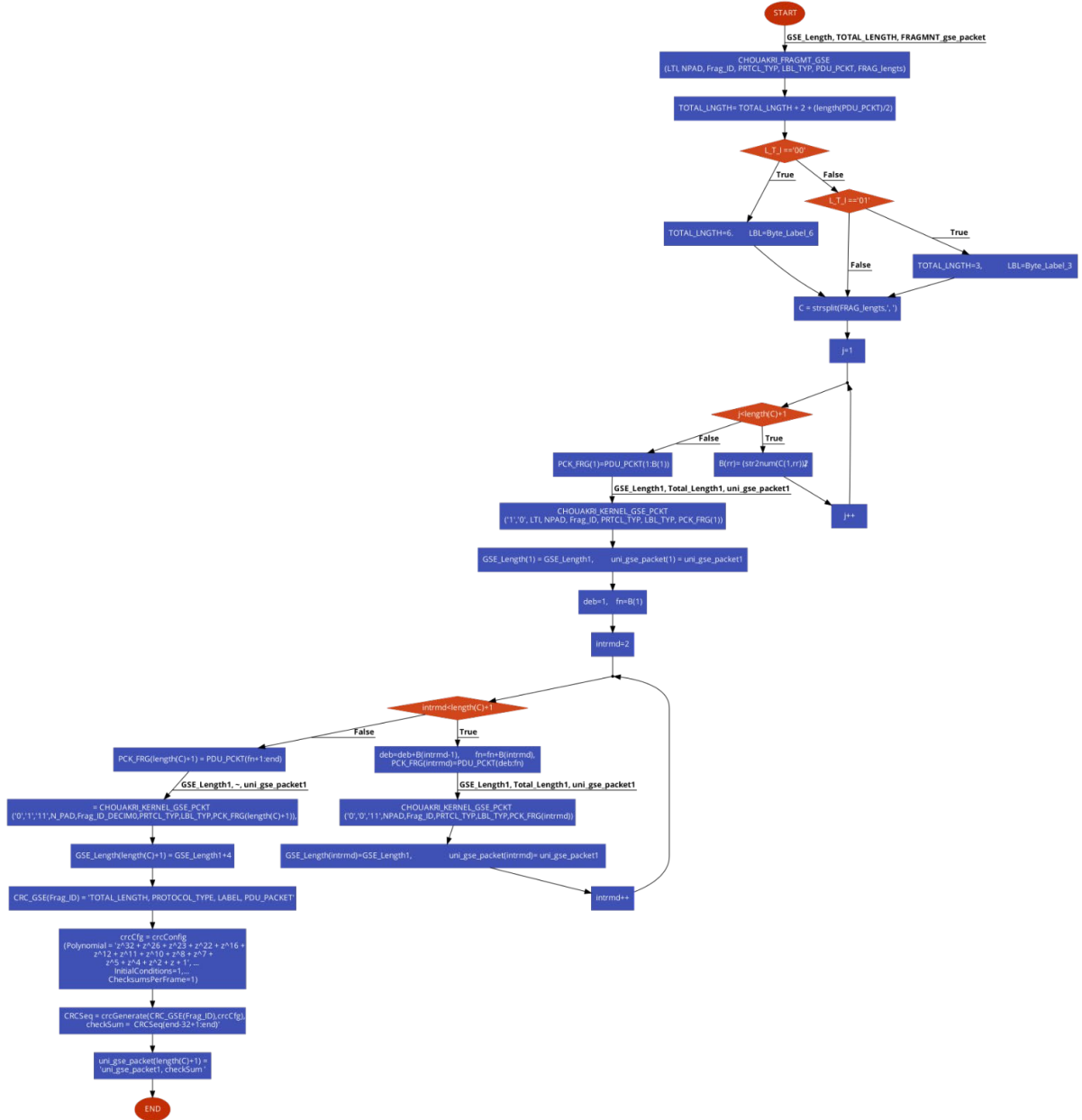


Figure 6: “CHOUAKRI_FRAGMT_GSE” organization chart

3.4 GSE MATLAB-based UI

Once the 'CHOUAKRI_KERNEL_GSE_PCKT' and 'CHOUAKRI_FRAGMT_GSE' procedures were created, we developed a MATLAB-based user interface (Chouakri, 2024b), allowing the whole set of GSE packet generation cases for 'Type 2: EtherType- compatible type fields' as shown in Figure 7. The user interface consists of 3 drop-down menus, which would be executed in series, each mentioned by the corresponding label, which are:

- The main “PADDING-PDU-RESET”: which indicates the possibility of executing one of the 3 actions mentioned by the title of the label;

- 'GSE PACKET TYPE': which indicates the possibility of performing one of the 2 actions which are: 'NO-FRAGM' (for the complete PDU packet), or 'FRGMNT' (to generate the GSE packets from a fragmented PDU packet)"
- 'LABEL TYPE': which indicates the possibility of executing one of the 2 actions which are: 'MAC 6 BYTES' or 'MAC 3 BYTES' (to choose respectively an NPA address of 6 or 3 bytes).

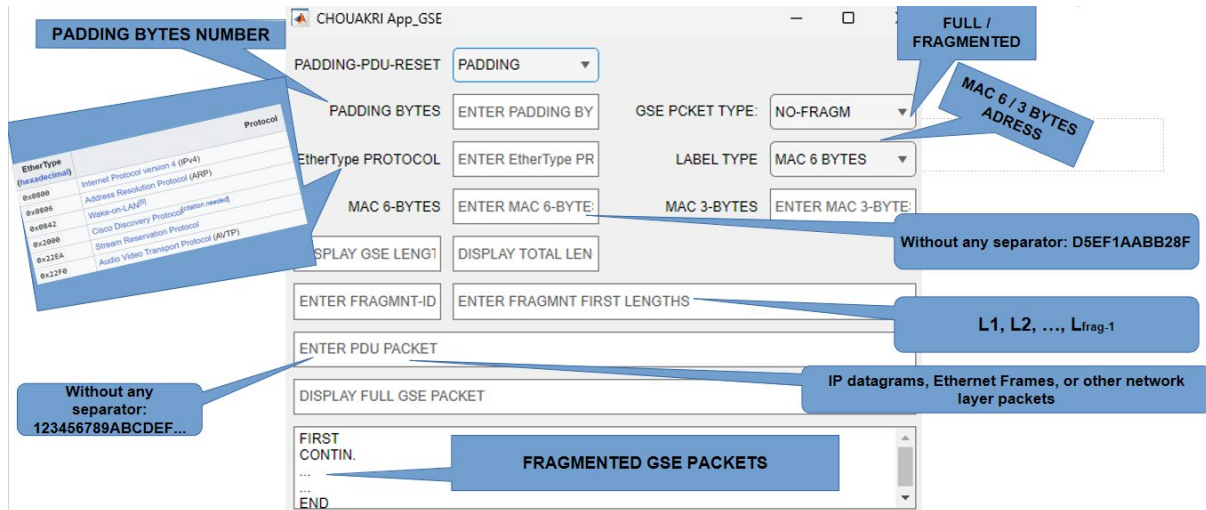


Figure 7: GSE Packet Generation User Interface Graphics

Additionally, the user interface includes a set of text editor fields to enter the required data for each specific generated GSE packet type referred to by their corresponding labels, namely:

- 'PADDING BYTES': This indicates the number of 0 padding bytes;
- 'EtherType PROTOCOL': to enter in the corresponding text editor the EtherType compatible Type fields between 0x600 (1536 decimal) and 0xFFFF;
- A 6- or 3-byte NPA address;

A set of 7 text editors is available, without any corresponding labels; whose role is as follows:

- 3 of them are intended to capture, each, the required information as follows:
 - 'ENTER FRAGMNT-ID': to enter the fragment ID;
 - 'ENTER FRAGMNT FIRST LENGTHS': to enter the fragmentation lengths excluding the last one (which would be deduced by the algorithm) expressed in decimal notation;
 - "ENTER PDU PACKET": To enter the original full PDU packet expressed in hexadecimal notation;
- 4 of them are intended to display the resulting data, as follows:
 - "SHOW GSE LENGTH": To display one or more GSE packet lengths for a full or fragmented GSE packet respectively;
 - "SHOW TOTAL LENGTH": to display the total length of the GSE packet(s);

- “SHOW FULL GSE PACKAGE”: to display the generated full GSE package;
- And the last one allows you to display the generated fragmented GSE packets, each on a new line.

4. Results

First, we note that to display, and subsequently make ready for use, the GSE - Generic Stream Encapsulation- -User Interface- packet generation user interface, shown in Figure 7, the user must run the Matlab file ' CHOUAKRI_App_GSE_PACKET.m ' (Chouakri, 2024b). The developed GSE Generic Stream Encapsulation UI is based on the main drop-down list indicated by the label "PADDING-PDU-RESET". This allows you to select and activate one of the following three actions:

A- RESET:

It is important to note that the "RESET" action must be executed each time to switch between actions, for example, "PADDING" and "PDU". Figure 8 illustrates the result of the "RESET" action when no data is displayed in the four specified text editors and the previously entered data is retained.

The screenshot shows the CHOUAKRI App_GSE window. At the top, there's a diagram of a GSE packet structure with fields: 1b, 1b, 00, GSE Length, Protocol Type, Label, and PDU. Below this, the main interface has a dropdown menu labeled 'PADDING-PDU-RESET' with 'RESET' selected. To the right of the dropdown is a 'GSE PKET TYPE' dropdown set to 'NO-FRAGM'. Below the dropdown are several input fields: 'PADDING BYTES' (3), 'EtherType PROTOCOL' (800), 'MAC 6-BYTES' (A1B2C3D4E5F6), and 'MAC 3-BYTES' (ENTER MAC 3-BYTE). There are also 'GSE LENGTH' and 'TOTAL LENGTH' fields. The 'GSE LENGTH' field contains '1', and the 'TOTAL LENGTH' field contains '11, 15, 19'. At the bottom, there's a hex string '123456789ABCDEF123456789ABCDEF123456789ABCDEF123456789ABCDEF' and two buttons: 'FULL GSE PACKET' and 'FRAGMENT GSE PACKET'.

Figure 8: Performing GSE Packet Generation Reset

B- PADDING

The "PADDING" action only takes into account the value entered in the "PADDING BYTES" text editor, 3 in this example, as illustrated in Figure 9. The generated GSE packet is composed of 3 bytes with the value "0", as illustrated by a blue highlight.

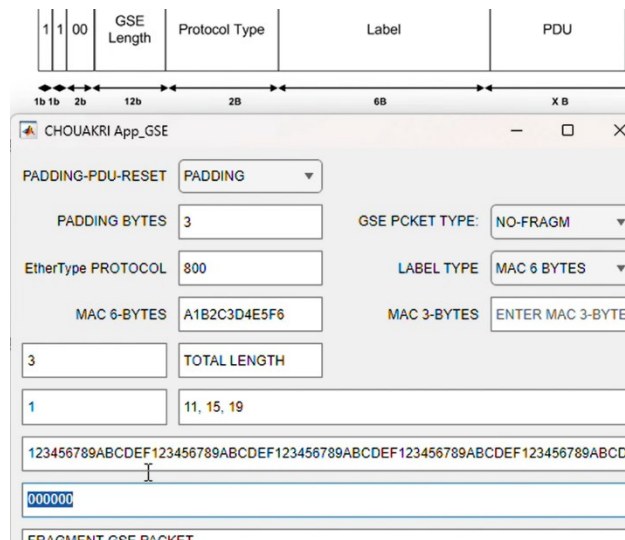


Figure 9: GSE Packet Generation “Padding” Execution

C- 'PDU'

The "PDU" action generates a GSE packet containing at least its mandatory "fixed" header and its data. The "PDU" action takes into account all the information entered. It first determines the type of GSE packet by testing "NO-FRAGM"/"FRGMNT".

C-1 “FRAGM-FREE”

Figure 10 shows the result of GSE packet generation with the "NO-FRAGM" drop-down menu. In this example, the x800 EtherType protocol, "MAC 6 BYTES", was chosen with an arbitrary address "A1B2C3D4E5F6". This data is visible, highlighted in blue, in the corresponding GSE packet generation text editor, with the required header and data fields.

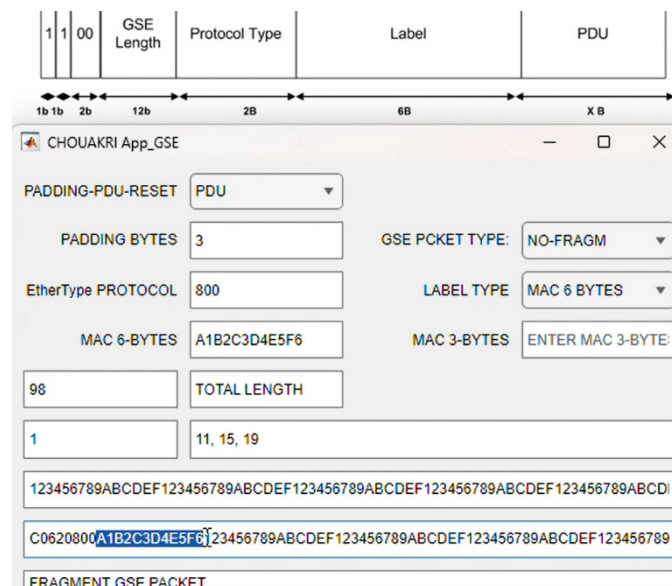
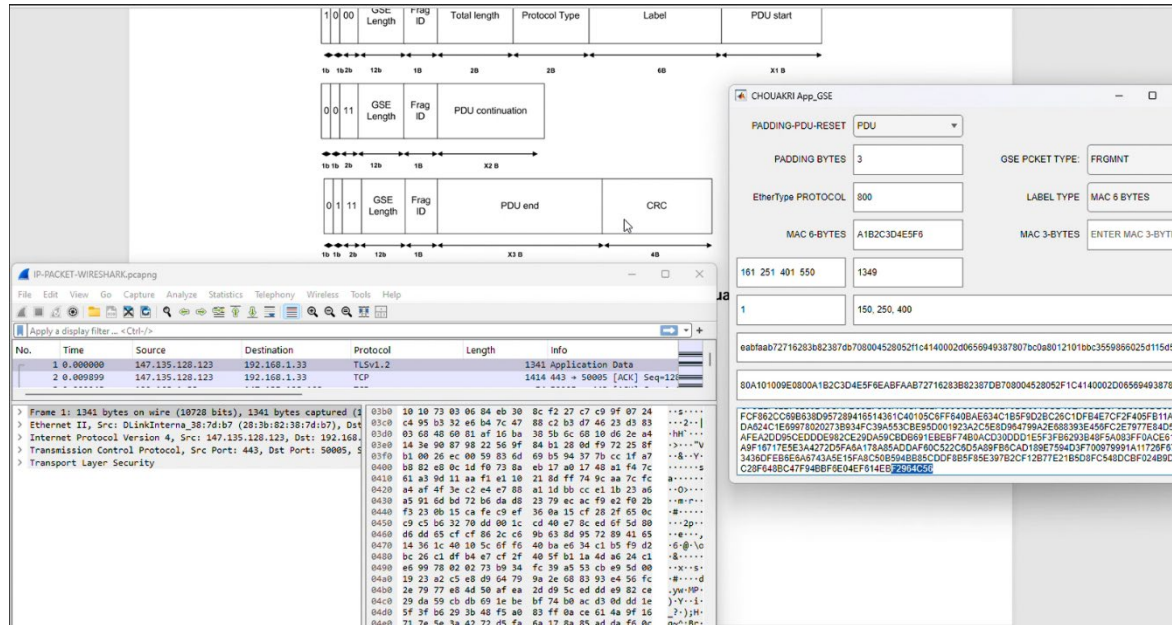


Figure 10: “NO-FRAGM” execution of GSE packet generation

C-2 'FRGMNT'

The 'FRGMNT' drop-down list is enabled if the PDU length exceeds 4 KB. In this case, in addition to the information required in the 'NO-FRAGM' case, the fragmentation identifier as well as the length of the first GSE packets are taken into account. Figure 11 illustrates a realistic scene based on a vivo Ethernet packet captured by the Wireshark application (Wireshark,

2026). In this example, as shown in Figure 11, a set of {150, 250, 400} bytes of first length is given arbitrarily where the lengths of 4 generated fragmented GSE packets are, respectively, 161, 251, 401 and 550, provided that the original PDU length is 1341 bytes. In Figure 11, the final part of the generated GSE packet is shown where the calculated CRC32 value on 4 bytes is highlighted in blue.



of operation of GSE packet generation in its different types. This makes it easier for users in their first sessions without intensive training.

In terms of functionality, emphasis is placed on multi-platform compatibility: the software could work on PC, Mac, Linux, and is adaptable to tablets and smartphones when relevant, as well as short loading times, smooth interactions, and real-time updates of results and visualizations.

In sum, the project offers educational impact and uses, as follows:

- For students and developers: a testing and learning tool that transforms complex standards into an interactive and accessible experience.
- For teachers and laboratories: a platform for demonstrating and validating DVB-S2 IP broadcasting algorithms, with concrete examples and reproducible scenarios.

5.2 User Interaction

The developed user interface is itself interactive, producing instant results based on queries based on input data. This is justified by the dynamic loading of results after each action (generation, fragmentation, testing).

Thus, a system of notes/assessments is planned for the scenarios or exercises accompanied by challenges and scenarios proposed to encourage discovery and experimentation.

Overall, the interface is described as user-friendly, with drop-down menus for key options (packet type, fragmentation, MAC address, etc.), and editors for entering data and displaying results.

5.3 Deployment

The project is built on a modular and well-documented architecture, facilitating understanding and adoption by developers and students.

The fact that the project is free to download provides availability, at hand, to an unlimited number of users while strengthening its international reach. The feedback provided by the '*MathWorks File Exchange*' platform, which hosts the project, allows for necessary improvements to the project to demonstrate its educational usefulness.

Thus, it is possible to use the project as a teaching aid for modules on network architectures, broadcast protocols and IP flow management.

Finally, we can say that the user interface promotes reproducibility between students and researchers and to replicate experiments with the same hypotheses. Also, the user interface and the associated files are designed to be reused in an educational setting (studying GSE flows according to ETSI TS 102 606).

Its impact is imminent on all students to acquire theoretical and practical knowledge related to the subject of IP encapsulation and GSE - Generic Stream Encapsulation - in particular. It thus allows members of the academic world, even industry, to simulate, generate and analyze GSE packets without worrying about the additional costs for such experiments. Thus, the project allows telecommunications researchers to develop and test new algorithms by having a reliable test bench compliant with the ETSI TS 102 606 standard. The tool targets educational scenarios and use in test benches, which is useful for universities and research centers. Moreover, as the application runs on MATLAB R2024b and later versions, this guarantees wide adoption in the academic and industrial world.

5.4 Open-Source GSE Tools

Several implementations of Generic Stream Encapsulation (GSE) packets generation exist in the open-source ecosystem, but they differ greatly in terms of target audience, functionality, and distribution (See Table 1).

Table 1: Open-Source GSE Tools Table description Chouakri, 2026)

Project	Language / Environment	Main Function	Target Users
<i>Libgse (ETSI, 2026)</i>	C (Linux library)	GSE encapsulation / decapsulation	System developers, R&D engineers
<i>dvb-gse (Rust) (Estévez, 2025)</i>	Rust (CLI)	GSE stream decoding and manipulation	SDR developers, advanced users
<i>GSEextract (Pavur et al., 2020)</i>	Python	Extraction of GSE packets into PCAP files	DVB analysts, SDR users
<i>Python GSE tools (Open Research Institute, 2021)</i>	Python / Jupyter	Experimental GSE processing	Amateur radio and SDR community
<i>MATLAB Interface (Chouakri, 2024)</i>	MATLAB (GUI)	GSE packet generation and analysis	Students, educators, researchers

The proposed tool is positioned as an educational solution for the generation and analysis of GSE packets in an academic environment. It constitutes a technological solution based on generic stream encapsulation (GSE) and on the interaction between DVB-S2 protocol (Digital Video Broadcasting via Satellite 2 and generation) and IP (Internet Protocol), for the generation and validation of GSE packets according to the standards of the reference document ETSI TS 102 606-1 V1.2.1 (2014-07) "Digital Video Broadcasting (DVB); Generic Stream Encapsulation (GSE) Protocol TS 102 606-1 - V1.2.1 - Digital Video Broadcasting (DVB); Generic Stream Encapsulation (GSE); Part 1: Protocol (ETSI, 2014). The work is based on an integrated software approach accessible to students and developers to test, generate and evaluate DVB-IP streams in simulated conditions, while supporting real scenarios of packet encapsulation and fragmentation (Type 2: compatible EtherType fields). **Conclusion**

The developed GSE packet generation user interface project is comprehensive. It allows students and developers to generate GSE packets intended for DVB-S2 IP communications, serving as a testbed for this task, and strictly adheres to the ETSI TS 102 606 standard: "Digital Video Broadcasting (DVB); Generic Stream Encapsulation (GSE)". Protocol". Nevertheless, the project can optionally take into account the Type 1: Next header type in the Protocol_Type field. This can be used to identify link-specific protocols and/or to indicate the presence of extension headers containing additional optional protocol fields (e.g., bridging encapsulation). Furthermore, the project can be extended to pass IP packet information up to the DVB-S2 physical layer.

Acknowledgment

The author would like to express his acknowledgment to **Dr. Meryem Romaiassa DJELLOULI** for her support throughout the development of this work.

References

- Cantillo, J., Collini-Nocker, B., Bie, U. D., Rio, O. D., Fairhurst, G., Jahn, A., & Rinaldo, R. (2008). GSE: A flexible, yet efficient, encapsulation for IP over DVB-S2 continuous generic streams. *International Journal of Satellite Communications and Networking*, 26(5), 437–457. <https://doi.org/10.1002/sat.915>
- Chouakri, S. A. (2024a). *GSE - Generic Stream Encapsulation - Packet Generation* (Version 1.1.0) [MATLAB software]. MATLAB Central File Exchange. <https://www.mathworks.com/matlabcentral/fileexchange/175538>
- Chouakri, S. A. (2024b). *GSE UI Matlab code file* [Computer software]. MATLAB Central File Exchange. https://www.mathworks.com/matlabcentral/fileexchange/175538/files/CHOUAKRI_App_GSE_PACKET.m
- Chouakri, S. A. (2024c). *GSE Fragmented Packet Matlab code file* [Computer software]. MATLAB Central File Exchange. https://www.mathworks.com/matlabcentral/fileexchange/175538/files/CHOUAKRI_FRAGMT_GSE.m
- Chouakri, S. A. (2024d). *GSE Kernel Matlab code file* [Computer software]. MATLAB Central File Exchange. https://www.mathworks.com/matlabcentral/fileexchange/175538/files/CHOUAKRI_KERNEL_GSE_PCKT.m
- Chouakri, S. A. (2024e). *Full GSE Packet Generation video illustration* [Video]. Google Drive. https://drive.google.com/file/d/1NvtU7Qz3wAahfj32I_1SLuumbhKntGMU/view
- Chouakri, S. A. (2024f). *Fragmented GSE Packet Generation video illustration* [Video]. Google Drive. https://drive.google.com/file/d/1EU3HY-Rauyg3JyQ2bTiHF8YQ5Ga8Z_Hu/view
- Chouakri, S. A. (2026). Comparative note: Open-source GSE tools and the MATLAB interface. https://chatgpt.com/s/t_6999b37c65bc81918d9599ea75e5e58b
- Collini-Nocker, B., & Fairhurst, G. (2004). ULE versus MPE as an IP over DVB encapsulation. In **Proceedings of HET-NETs 2004**.
- DVB. (2025). *Implementation guidelines for DVB-I* (DVB Document A184r2). DVB Project.
- Estévez, D. (2025). *Rust implementation of DVB-GSE* [Source code]. GitHub. <https://github.com/daniestevez/dvb-gse>
- ETSI. (1997a). **Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12-GHz satellite services** (ETSI EN 300 421). European Telecommunications Standards Institute.
- ETSI. (1997b). *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television* (ETSI EN 300 744). European Telecommunications Standards Institute.
- ETSI. (1998). *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for cable systems* (ETSI EN 300 429). European Telecommunications Standards Institute.
- ETSI. (2007). **Digital Video Broadcasting (DVB); IP Datacast over DVB-H: Set of specifications for Phase 1** (ETSI TS 102 468 V1.1.1). European Telecommunications Standards Institute.

- ETSI. (2014). *Digital Video Broadcasting (DVB); Generic Stream Encapsulation (GSE); Part 1: Protocol* (ETSI TS 102 606-1 V1.2.1). European Telecommunications Standards Institute.
- ETSI. (2026). *Open Source library implementing the Generic Stream Encapsulation (GSE)* [Source code]. GitHub. <https://github.com/nsat/libgse>
- Ewald-Arostegui, N. L., Fairhurst, G., & Yun-Garcia, A. (2010). A framework for an IP-based DVB transmission network. *International Journal of Digital Multimedia Broadcasting*. <https://doi.org/10.1155/2010/394965>
- Fairhurst, G., & Collini-Nocker, B. (2005). *Unidirectional lightweight encapsulation (ULE) for transmission of IP datagrams over an MPEG-2 transport stream (TS)* (IETF RFC 4326). IETF. <https://doi.org/10.17487/rfc4326>
- Fairhurst, G., & Collini-Nocker, B. (2008). *Extension formats for unidirectional lightweight encapsulation (ULE) and the generic stream encapsulation (GSE)* (IETF RFC 5163). IETF. <https://doi.org/10.17487/rfc5163>
- Lanigan, P. (2021). *DVB content delivery technologies: DVB-I and DVB-TA overview* [PowerPoint slides]. HbbTV Association. <https://www.hbbtv.org/wp-content/uploads/2021/02/DVB-I-introduction-HbbTV-February-2021.pdf>
- Levil, L., & Moulin, R. (2023). Leveraging broadcast networks to broaden the reach of OTT services: An introduction to DVB Native IP [Conference presentation]. IUT Workshop, Bangalore, India. https://www.itu.int/en/ITU-T/Workshops-and-Seminars/2023/0511/Documents/1_Regis%20Moulin_.pdf
- Open Research Institute. (2021). *DVB Generic Stream Encapsulation implementation* [Source code]. GitHub. <https://github.com/OpenResearchInstitute/GSE>
- Papaleo, M., Firrincieli, R., Cioni, S., Vanelli-Coralli, A., Corazza, G. E., Lee, H. J., & Kim, P. S. (2008). Link layer FEC in DVB-RCS: Performance evaluation in nLoS conditions. In *Proceedings of IEEE Vehicular Technology Conference (VTC Spring 2008)* (pp. 2972–2976). IEEE. <https://doi.org/10.1109/VETECS.2008.314>
- Pavur, J., Moser, D., Strohmeier, M., Lenders, V., & Martinovic, I. (2020). A tale of sea and sky: On the security of maritime VSAT communications. In *IEEE Symposium on Security and Privacy (SP)* (pp. 1384–1400). IEEE. <https://doi.org/10.1109/SP40000.2020.00056>
- Wireshark. (2026). *Download Wireshark* [Software]. <https://www.wireshark.org/download.html>