FEDERAL UNIVERSITY OF SANTA CATARINA
TECHNOLOGY CENTER
AUTOMATION AND SYSTEMS DEPARTMENT
UNDERGRADUATE COURSE IN CONTROL AND AUTOMATION ENGINEERING

Guilherme Prudente da Silva

Design and Execution of Customer Integration Test of Subsea Valve Actuator Regarding the Communication and Electric Interface

Florianópolis
2023

Guilherme Prudente da Silva

Design and Execution of Customer Integration Test of Subsea Valve Actuator Regarding the Communication and Electric Interface

Final report of the subject DAS5511 (Course Final Project) as a Concluding Dissertation of the Undergraduate Course in Control and Automation Engineering of the Federal University of Santa Catarina.
Supervisor: Prof. Hector Bessa Silveira, Dr.
Co-supervisor: Ricardo José Rabelo

Florianópolis
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Silva, Guilherme Prudente da
   Design and Execution of Customer Integration Test of Subsea Valve Actuator Regarding the Communication and Electric Interface / Guilherme Prudente da Silva ; orientador, Hector Bessa Silveira, 2023.
   78 p.

   Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Engenharia de Controle e Automação, Florianópolis, 2023.

   Inclui referências.

   1. Engenharia de Controle e Automação. 2. Atuador de válvula submarina.. 3. Nível de Prontidão Tecnológica.. 4. Automação.. 5. Validação.. I. Silveira, Hector Bessa. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Controle e Automação. III. Título.

Guilherme Prudente da Silva

Design and Execution of Customer Integration Test of Subsea Valve Actuator Regarding the Communication and Electric Interface

This dissertation was evaluated in the context of the subject DAS5511 (Course Final Project) and approved in its final form by the Undergraduate Course in Control and Automation Engineering

Florianópolis, May 16th, 2023.

Dedico este trabalho aos meus pais Marcia de Souza Prudente da Silva e Rodrigo Bianchini da Silva, à minha irmã Carolina Prudente da Silva, à minha avó Maria do Carmo de Souza (in memorian), à minha namorada Taisa Pereira Salgueiro e aos meus amigos Jessé Westphal Schwambach e Pedro Henrique Cândido Lenzi.

ACKNOWLEDGEMENTS

Primeiramente gostaria de agradecer aos meus pais Rodrigo Bianchini da Silva e Marcia de Souza Prudente da Silva, à minha irmã Carolina Prudente da Silva e à minha avó Maria do Carmo de Souza (in memorian) pelo apoio constante e incondicional que sempre me foi dado e que me possibilitou chegar onde cheguei. Aos demais familiares por sempre torcerem por mim e me acompanharem. À Roseli Terezinha Brick, por sempre visar o meu melhor. À minha namorada Taisa Pereira Salgueiro pelo carinho e acolhimento ao longo da minha trajetória no exterior. Aos meus amigos de graduação Jessé Westphal Schwambach e Pedro Henrique Cândido Lenzi por todo o companheirismo e auxílio que foram fundamentais em tantos momentos. Aos demais amigos que graduacao me trouxe - Gabriel Martins do Rosário, Gustavo Albino, Renan Amorin e tantos outros - pelos aprendizados e vivencias que deixaram minha experiência, tanto no curso quanto fora, muito mais leve e proveitosa. Aos meus amigos que tem me acompanhado nessa jornada, especialmente a Deborah Dotta Corrêa, Renan Phelippe Eggers, Francisco de Paula Lemos, Julia Nakyama Silveira, Lucas Khoury, Theo Gomes Oliveira, Mariana Rocha Silva, Gabriela Zetehaku, Ana Carolini de Souza, Guilherme Barreto Malischeski e inúmeros outros que foram essenciais em tantos momentos da minha vida. Aos meus amigos de Aachen em especial Julia Galhardo Serafim, João Marcelo Neves Búrigo, Rhanna Auler, Maurício Salvador Losso, Rodrigo Neves Soares, Nicolas Antero Nunes, Leonardo Santiago e tantos outros por tornarem a minha experiencia na Alemanha infinitamente melhor.

Agradeço meu orientador Hector Bessa Silveira pela dedicação, atenção e suporte que me foi provido sempre que requisitado. Aos professores do DAS pelos incontáveis ensinamentos que me capacitaram para todos os desafios surgiram.

Ao Alexandre Orth e Amadeu Plácido Neto pela confiança e pela oportunidade de fazer parte desse projeto. Ao meu orientador na Rexroth João Pedro Duarte da Silva, por toda a assistência e suporte tanto na empresa quanto por fora. Ao meu mentor e colega de trabalho David Claus pela orientação, paciência e dedicação durante meu estágio. A todos os meus colegas da Bosch Rexroth especialmente ao Thomas Franz, Tim Strohmenger, Gottfried Hendrix, Juergen Schneider, Stein Berg Nilsen, Maximilian Kraus, Sebastian Amend e Markus Kubacki que fizeram com que o ambiente de trabalho fosse um local, além de sério e produtivo, lúdico e leve.

"Testing leads to failure,
and failure leads to understanding."
Burt Rutan (1943)

ABSTRACT

Before a technology is ready to be implemented in the market, looking for validation in most of its aspects is key. For that reason, the standard called Technology Readiness Level (TRL) was created, which consists of levels 1 up to 9, to evaluate the level of how prepared a product is to be on the market. The Project Subsea Automation System (PJ-SAS) in Bosch Rexroth, Germany, is using an adaptation of this standard - defined by the recommended practices API 17N - for their Subsea Valve Actuator (SVA). This adaptation consists of levels 0 to 7 (TRL 0 to TRL 7) and is used to validate if such a device is indeed ready to be implemented in a real scenario. Currently, the team is on the way to getting the TRL 4 (adapted) certification, and the next step is to evaluate if the technology meets the requirements of TRL 5 (adapted). In this context, this work has the objective of investigating if the SVA achieved indeed TRL 5 but only with respect to its software aspects. To do this, the requirements of TRL 5 were analyzed, a test procedure was designed to determine if the device achieved or not TRL 5, and a setup was built to execute them. More specifically, since the TRL 5 requests to evaluate the interfaces of the device and the software communication of all the subsea devices are done with the CANopen protocol, it was necessary to test the attributes from this protocol. Hence we used the test procedure elaborated by the Subsea Instrumentation Interface Standardization (SIIS) group and the CANopen Conformance Test Tool (CTT) from the CAN in Automation (CiA) group, so we were able to verify the operation of the main CANopen features. Regarding the evaluation methodology, we assessed the results of those tests based on the customer's point of view. It was developed also an automatic process to run the tests so that they would be easily reproducible and could be applied to other devices as well. The results obtained in this work show that, even though the SVA passed some of the referred tests, the device is not yet ready to fully achieve the TRL 5 from the software point of view, because we were able to identify critical errors in the current state of development of the SVA that had not been detected by the team before this work and hence need to be adjusted before the device is launched into the market. For this reason, the tests were a good addition to the PJ-SAS and, since the issues are easily fixable, we expect that they can be soon solved and the SVA can attain the TRL 5 in regards to the software communication.

Keywords: Subsea valve actuator. Technology Readiness Level. Automation. Validation.

RESUMO

Antes de uma tecnologia estar pronta para ser implementada no mercado, buscar a validação na maioria de seus aspectos é fundamental. Por essa razão, o padrão chamado de Nível de Prontidão Tecnológica (Technology readiness Level - TRL) foi criado, o qual consiste de níveis que vão de 1 até 9, para avaliar o quão preparado um produto está para o mercado. O Projeto Sistema de Automação Submarina (Project subsea Automation System - PJ-SAS) na Bosch Rexroth, Alemanha, está usando uma adaptação desse padrão - definido pelas práticas recomendadas API-17N - para o seu Atuador de Válvula Submarina (Subsea Valve Actuator - SVA). Essa adaptação consiste de níveis que vão de 0 até 7 (TRL 0 até TRL 7) e é usado para validar se o aparelho está realmente pronto para ser implementado em um cenário real. Atualmente, o time está a caminho de adquirir o certificado do TRL 4 (adaptado) e o próximo passo é avaliar se a tecnologia atende aos requisitos do TRL 5 (adaptado). Nesse contexto, esse trabalho tem o objetivo de investigar se o SVA realmente alcançou TRL 5 mas apenas em relação aos seus aspectos de software. Para fazer isso, os requisitos do TRL 5 foram analisados, um procedimento de testes foi produzido para determinar se o aparelho atingiu ou não TRL 5 e um setup foi construído para executá-los. Mais especificamente, dado que o TRL 5 requisitava avaliar a interface do aparelho e a comunicação de software de todos os aparelhos submarinos é feita com o protocolo CANopen, era necessário para testar os atributos deste protocolo. Portanto nós usamos o procedimento de testes elaborados pelo grupo Padronização da Interface de Instrumentação Submarina (Subsea Instrumentation Interface Standardization - SIIS) e a Ferramenta de Teste de Conformidade CANopen (CANopen Conformance Test Tool - CTT) do grupo CAN na Automação (CAN in Automation - CiA) para que nós possamos verificar o funcionamento dos principais atributos CANopen. Com relação a metodologia de avaliação, nós julgamos os resultados dos testes baseados no ponto de vista do cliente.Foi desenvolvido também um processo automático para executar os testes para que sejam facilmente reprodutíveis e possam ser aplicados também a outros dispositivos. Os resultados obtidos neste trabalho mostram que, apesar do SVA ter passado em alguns dos testes referidos, o aparelho não está pronto para alcançar totalmente o TRL 5 do ponto de vista do software, pois nós conseguimos identificar erros cruciais no estado atual de desenvolvimento do SVA que não tinham sido identificados antes desse trabalho e portanto precisam ser ajustados antes do equipamento ser lançado no mercado. Por essa razão, os testes foram uma boa adição para o PJ-SAS e, dado que os problemas são facilmente consertados, nós esperamos que eles possam ser brevemente solucionados e o SVA possa obter o TRL 5 no que se refere a comunicação de software.

Palavras-chave: Atuador de válvula submarina. Nível de Prontidão Tecnológica. Automação. Validação.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ACK	Acknowledgement
ACM	Actuator Control Module
API	American Petroleum Institute
CAN	Controller Area Network
CDM	CAN Device Monitor
CENELEC	European Committee for Electrotechnical Standardization
CiA	CAN in Automation
COB-ID	Communication Object Identifier
CRC	Cyclic Redundancy Checksum
CTT	Conformance Test Tool
DLC	Data Length Code
EDS	Electronic Data Sheet
EMCY	Emergency Service
EOF	End of Frame
FMECA	Failure Mode, Effects, and Criticality Analysis
HIL	Hardware In the Loop
IDE	Identifier Extention Bit
IEC	International Electrical Committee
ISAS	Surface Application Systems
ISD	Intelligent Seabed Devices
ISO	International Organization for Standardization
LSS	Layer Setting Services
NMT	Network Management
OD	Object Dictionary
PDO	Process Data Object
PJ-SAS	Project - Subsea Automation Systems team
PLC	Programmable Logic Controller
RPDO	Receive Process Data Objects
RTR	Remote Transmission Request
SCM	Subsea Control Module
SDO	Service Data Object
SEM	Subsea Electronics Module
SIIS	Subsea Instrumentation Interface Standardization
SIT	System Integration Testing
SOF	Start of Frame
SVA	Subsea Valve Actuator
SYNC	Synchronization Object

TPDO	Transmit Process Data Objects
TRL	Technology Readiness Level
XT	Xmas Tree

CONTENTS

1 INTRODUCTION

1.1 CONTEXTUALIZATION AND MOTIVATION

The gas and oil industry plays a big role in the global economy. The market size of global exploration and production of gas and oil has a value of 5 trillion dollars (IBISWORLD, 2022) making up a large portion of the global economy and it is only expected to grow (INVESTOPEDIA, 2022). Inside this scenario, there is the subsea sector, which has a huge value on the market, approximately 16.18 billion dollars (FUTURE, 2022).

One of the main results of this investment are offshore operations, which are very fundamental for draining oil and gas. Those processes are composed of many different components which need to cooperate simultaneously for the entire process to work. One of these components is the so-called *Christmas Tree* which is a device composed of valves and is attached to the wellhead to drain its resources (BAI, Y.; BAI, Q., 2010).

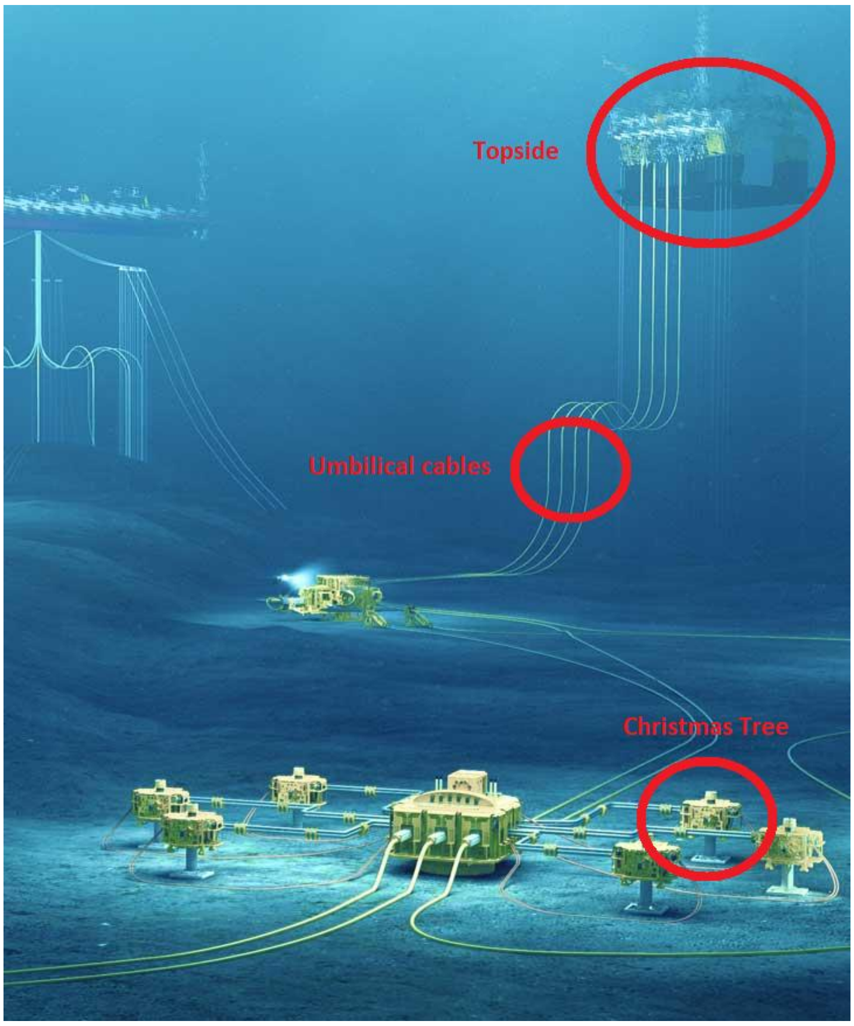
It is not possible to operate those devices manually, since they function at depths between 1830 to 3000 meters underwater (SILVA, 2019). To solve this problem, subsea actuators are used, so that the Christmas Tree can be controlled from the stations above the water, also known as *topside*. This connection between the subsea devices and the topside is made by an umbilical cable, which is responsible for not only sending commands to the equipment underwater but also providing everything that the devices may need, such as oil or energy. A general overview of this setup can be seen in Figure 1.

In relation to the subsea actuator, currently there are two main types: hydraulic actuators and electric actuators. Both of them have advantages and disadvantages in relation to the other. The hydraulic actuator has better fail-safe technology, which means that in case a problem occurs - such as a failure in the power or oil supply - the device is able to handle it. On the other hand, it requires constant insertion of oil through it and it is less environmentally friendly since it releases fluids into the ocean. The electric actuator doesn't require fluid entry, meaning that the umbilical cable for it is less thick and the device doesn't pollute the ocean. However, it has a more complex and costly fail-safe system, which leads to not being used as safety equipment (CLAUS, 2022; SILVA, 2019).

Given this scenario, the Project - Subsea Automation Systems team (PJ-SAS) from Bosch Rexroth saw this opportunity to come up with a solution, a device called Subsea Valve Actuator (SVA) which is a new type of subsea actuator. The innovation involved in this technology is that it is an electric actuator that has a simple fail-safe technology, which eliminates the mentioned disadvantages of both hydraulic and electric actuators. These features make this project coincide with the market trend since the technology and impacts on the oil and gas industry will be driven by both economic factors and sustainability issues (FUTURE, 2022). For that reason, this is a very interesting project for Bosch Rexroth as well.

But before any product can be released in the industry, it is crucial to seek validation of most of its aspects. This includes guaranteeing that the performance requirements will be

Figure 1 – Subsea production overview.



Source: Claus (2022).

met, that they will be ready to be implemented in real scenarios, and that the possibility of any failures that could be compromising will be minimized. Taking such issues into consideration, NASA created a methodology to estimate the technology maturity during its development phases, which is called Technology Readiness Level (TRL), that determines, from levels 1 to 9 level, how complete this product is for the real industry scenario, in which level 1 corresponds to the observation and report of the basic concept and level 9 to the finished system validated in an operational environment (TWI, 2021).

Nevertheless, even though the TRL qualification method was designed to be as far-reaching as possible, applying it to a subsea system isn't as easy as it sounds, given the extreme scenarios in which the devices are submitted. That's because of the high pressure that exists in the subsea environment, the necessity of isolating all the electronics involved from the water, the struggle to interact in the deep water, and other issues involved in operating in this scenario. Therefore, TRL doesn't match all the challenges that are encountered in the subsea environment.

For this reason, the project team at Rexroth follows a slightly different interpretation of the original TRL, which is provided by the API 17N – Recommended Practice on Subsea Production System Reliability, Technical Risk, and Integrity Managements. This norm considers the subsea scenario for measuring the TRL levels. In this adapted version, TRL levels are determined from 0 to 7, corresponding to levels 1 to 9 of the original TRL from NASA (INSTITUTE, 2017). Unless otherwise stated, from now on the term TRL will always refer to the original adapted version, and not the original one. Furthermore, for simplicity, levels 0 to 7 will be referred to as TRL 0 to TRL 7 respectively.

Recently, the SVA has attained TRL 4 - which requested that the device is tested in an environment equivalent to the intended scenario. This was obtained by operating the SVA in a chamber that had the pressure equivalent of a subsea environment. The next step is to achieve TRL 5 and, in order to attain it, all the interfaces of the technologies - hydraulic, mechanical, software, etc - should be ready to be integrated into the intended system. However, testing all the interfaces of a device isn't a simple task, due to involving the operation of many sectors of the project - like the mechanical area, software area, etc - so the team decided to approach the interfaces one at a time. The first area chosen to be tested was the software communication interface. This means that the technology should be ready to communicate with the system that it was originally planned for. In other words, the device should be able to connect with the setup that is used by the customers, so that it is possible to send and receive messages in a real-world scenario. Besides, since the communication of all the subsea devices is made with the CANopen protocol - which is a top-layer communication protocol for on-board systems - the tests need to be mainly focused on that. In the face of this scenario, the opportunity arose for the development of the present work. Unless otherwise stated, from now on, when the TRL 5 is mentioned, it will be from the software perspective even though the TRL 5 concerns many different areas of technology.

1.2 OBJECTIVES

The main goal of this work is to verify if the current state of development of the SVA meets or not the requirements of TRL 5. The specific objectives are as follows:

1. Determine the necessary requirements to achieve the TRL 5;
2. Design and develop test procedures related to TRL 5;
3. Create an environment with all the hardware and software necessary to run the CANopen tests;
4. Determine a methodology to evaluate if TRL 5 was attained by the SVA device;
5. Evaluate if TRL 5 was indeed obtained;
6. Automate the test procedure.

1.3 METHODOLOGY AND RESULTS

With regards to the methodology used to evaluate the results of the proposed tests, since TRL 5 classifies if the device is ready to be implemented in the intended scenario or not, it is necessary to check the communication interface between the SVA and the system of the client, which in this case are offshore drilling companies. For that reason, the methodology implemented in this work was a qualitative one, since it evaluated the tests from the client's perspective. This means that in order to evaluate if the TRL 5 was indeed attained, we used different weights on the individual tests according to what was considered more important for the client.

The results obtained in this work allowed us to conclude that the SVA did not attain the TRL 5 due to flaws that were then detected on some of the fundamental CANopen features of the SVA. Although we determined that the SVA technology wasn't at TRL 5 yet, the results here obtained had a very good impact on the PJ-SAS project of Bosch Rexroth, because we were able to identify critical issues in the current state of development of the SVA that had not been detected by the team before this work and hence need to be adjusted before the device is implemented in a real scenario. Besides, the critical errors that were identified by this work can be easily and quickly solved in such a way that the tests can be repeated and the expectation is that the product can be soon launched in the market.

1.4 DOCUMENT STRUCTURE

This document consists of the chapters described in the sequel:

1. **Technical Background:** we explain the main technical concepts and features regarding the devices, techniques, and methods used along this work.
2. **Problem Description:** we describe in further detail the problem here treated, that is, of evaluating TRL 5 for the SVA device.
3. **Proposed Solution and Used Methodologies:** we present the proposed solution as well as the applied methodologies.
4. **Development and Analysis of the Results Obtained:** we describe in detail the implementation of the proposed solution for the validation or not of TRL 5 for the SVA.
5. **Concluding Remarks and Future Perspectives:** we present some concluding remarks as well as suggestions for future works.

2 TECHNICAL BACKGROUND

In this chapter, we will describe the technical fundamentals necessary to properly understand the project and the issues underlying it. Section 2.1 gives an overview of the CAN network, Section 2.2 states what is this protocol and gives details of it, Section 2.3 lists the norms and standards that are used in this project, and Section 2.4 gives an overview to standard hydraulic and electric actuators.

2.1 CAN NETWORK

This section presents an overview of the CAN network. It was mainly based on references Pfeiffer, Ayre, and Keydel (2003) and Claus (2022).

A Controller Area Network (CAN) bus is a high-integrity serial bus system, developed by Robert Bosch GmbH in 1985. This bus was originally designed for automotive networks to allow sensors to report small values at a high frequency. For that reason, the CAN message frame has a size of only up to 8 bytes, but, on the other hand, the bus can handle many message frames per second.

Besides sending small messages, the CAN bus needs to be reliable, which is why a Cyclic Redundancy Checksum (CRC) is implemented in the bus. This CRC is an error-detecting code used to determine if a code line has been corrupted. Also, a CAN message can contain 50% of overhead or more, which is the additional memory required for supplemental information that allows a given message to be transmitted from one source to another destination, making the network very secure and reliable.

2.1.1 CAN Physical Layer

The CAN bus physical layer is composed of mainly two cables (labeled CAN High or CAN_H and CAN Low or CAN_L). The actual transmission of signals is done by measuring the difference between these two cables and converting it into data.

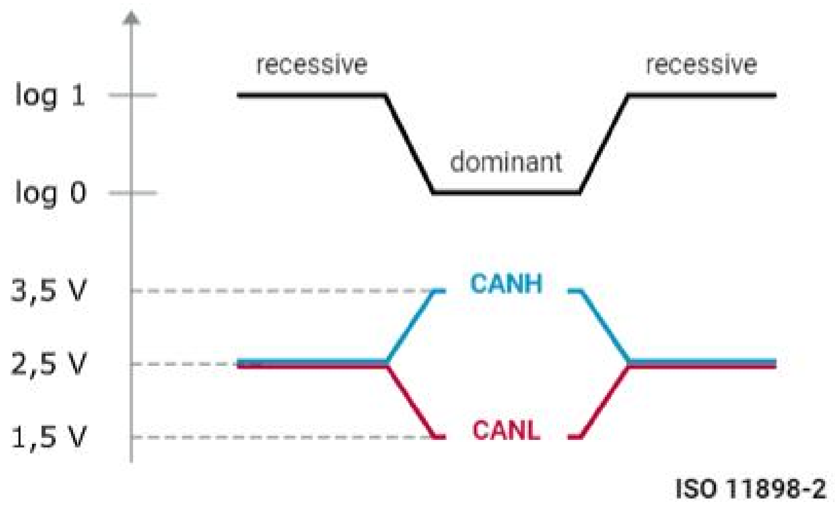
In case there is an overlay of information in the CAN bus, the logical 0 would be dominant to a logical 1. This means that if a node (a device connected to the bus) tries to send a 1 and another node is sending a 0 at the same time, the information from the second node will overwrite the first one. The reason for that is due to the circuit embedded in the bus and it is used to detect collisions between the nodes.

Also, it is important to clarify that there are two types of physical layers for the bus, the High-Speed CAN and the Low-Speed CAN or Fault-Tolerant CAN. They work in similar ways but with different features and voltage levels.

In High-Speed CAN, to send a logical 1, both cables stand with 2.5V, making the potential difference between them 0V. To transmit a logical 0, the CAN_H stand with 3,5V, and the CAN_L stands with 1,5V, making the difference between them 2V. It is possible to see a representation of the CAN signals in Figure 2. Moreover, this type of physical layer can work with a wide range of bit rates, more precisely from 10 kBits\s up to 1000 kBits\s.

With Fault-Tolerant CAN, also known as Low-Speed CAN, the voltage levels work differently than the High-Speed version. To send a logical 1, the CAN_L stands with 5V and the CAN_H stands with 0V, making the potential difference 5V. To send a logical 0, the CAN_L stays at 1,4V, and CAN_H stays at 3,6V making the difference between them 1,2V. Figure 3 shows a representation of this behavior. Also, one difference that makes this physical layer type unique is the fact that the data can still be sent even if one of both cables is ruptured, which makes it very robust to extreme scenarios. However, as its name already says, Low-Speed CAN cannot operate in such a wide range of baud rate as the High-Speed, it can only operate between 40 kBits\s up to 125 kBits\s.

Figure 2 – CAN_H and CAN_L signal levels in High-Speed CAN.

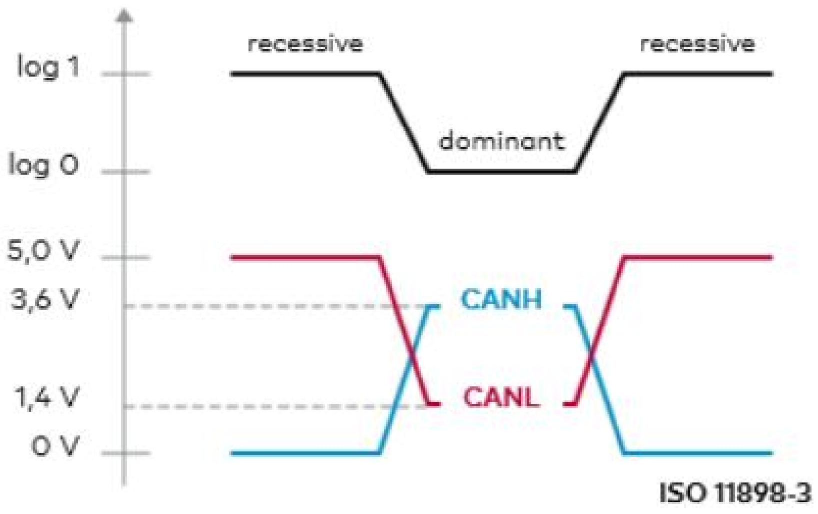


Source: Vector (2022).

Besides that, since the output signal of the CAN bus is relative, due to the potential difference between two voltages, this makes the bus more reliable against interference. For example, if something makes the voltage of the cables go down by 1V, the difference between the cables will remain the same.

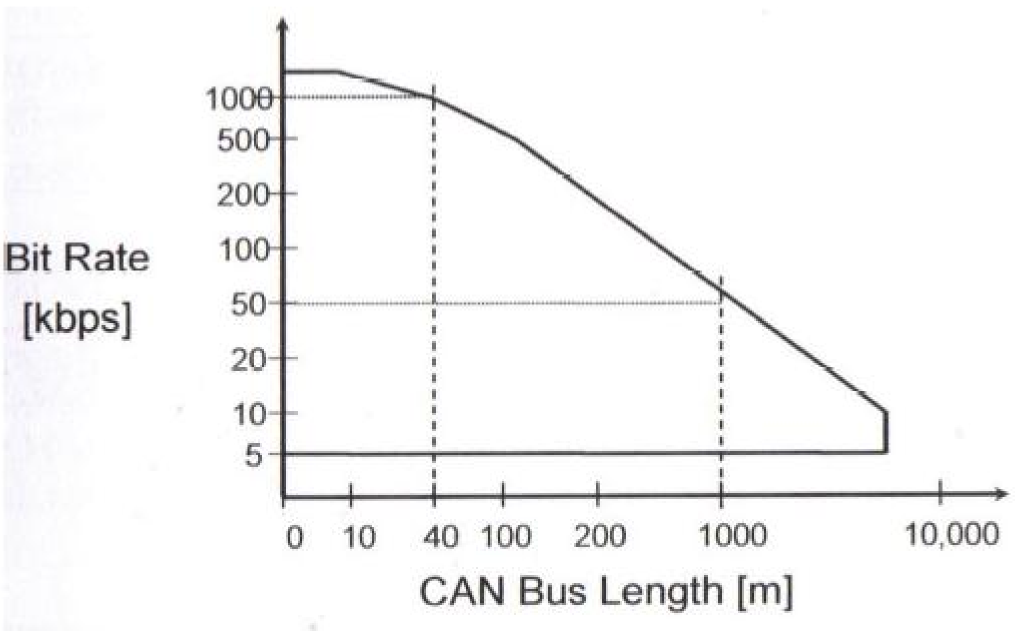
Another important factor from the physical layer, especially for subsea devices, due to operating with long cables, is the maximum bus length and maximum drop length. Although many factors are involved when calculating this value, the most important element is the bus speed. Figure 4 shows the influence that the bit rate has on the bus length and though it is possible to verify that, with long cables, it isn't possible to use fast bit rates.

Figure 3 – CAN_H and CAN_L signal levels in Fault-Tolerant CAN.



Source: Vector (2022).

Figure 4 – CAN Bit Rate versus Bus Length.



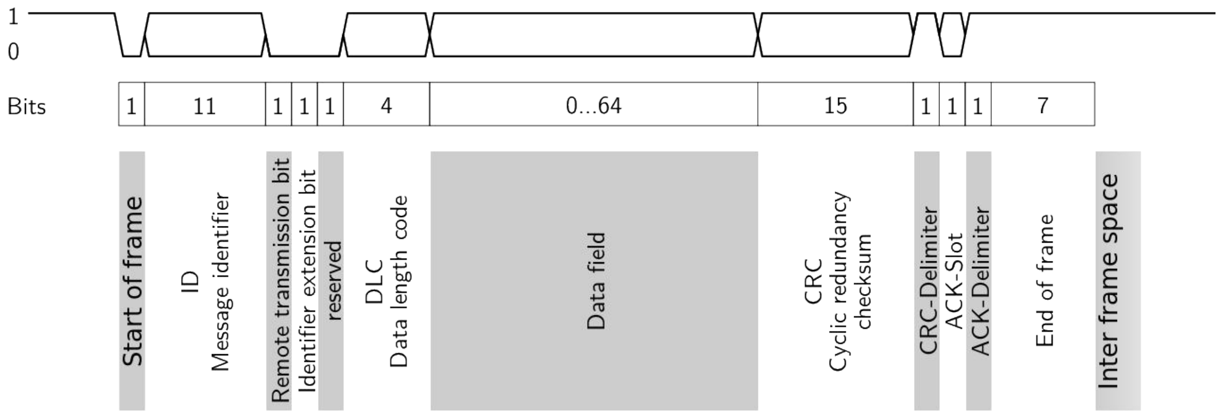
Source: Pfeiffer, Ayre, and Keydel (2003).

2.1.2 Data Frame

The CAN bus can offer various types of messages, but the one that is used the most often is the Data Framing containing process data, which is the normal data transmitted between elements in the bus. It is possible to see an illustration of the bus in Figure 5.

The different bit sections of the frame, from left to right, are:

Figure 5 – CAN message frame base format.



Source: Pfeiffer, Ayre, and Keydel (2003).

- **Start of Frame (SOF):** indication of frame start.
- **Message Identifier:** it is used to establish the destination of the message since in a bus you have a connection with many elements at the same time and the message may be exclusive to one of these elements. For that reason, it must be ensured that each message ID is unique in the network, so there are no conflicts of destiny.
- **Remote Transmission Request (RTR):** defines the frame type (data or remote frame).
- **Identifier Extention Bit (IDE):** used to enable a 29-bit identifier instead of a 11-bit.
- **Data Length Code (DLC):** specifies the number of bytes that are in this frame.
- **Data Field:** contain the data that is being transmitted.
- **CRC:** pattern calculated by a mathematical operation using the message to verify if the frame was sent correctly.
- **CRC-Delimiter:** used to give one bit time to the receiving node to compare the CRC calculated internally with the acquired in the message.
- **Acknowledgement (ACK)-Slot:** it gives a one bit time to acknowledge that they received the Data Frame.
- **ACK-Delimiter:** used to give one bit time to complete the CRC calculation.
- **End of Frame (EOF):** seven consecutive recessive bits to indicate that the message is over.
- **Inter frame space:** a period of 2 or 3 recessive bits before any other message is sent to avoid overlapping of frames.

2.2 CANOPEN

This chapter explains the concepts of CANopen that are related to this paper. It was based mainly on the Falch (2022), Pfeiffer, Ayre, and Keydel (2003) and Claus (2022)

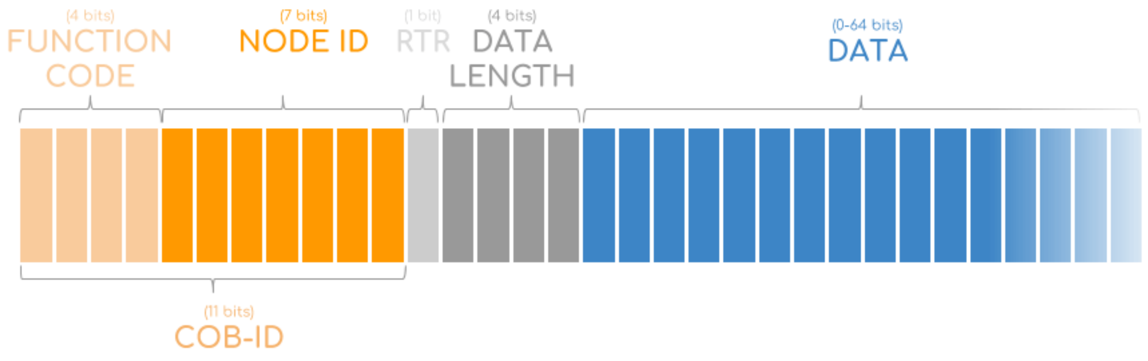
CANopen is a higher-layer communication protocol for embedded systems based on the CAN bus, that determines a variety of features for configuring devices and providing a well-established environment for the device's communication. It was initially designed for motion-oriented machine control systems, like handling systems. But currently, it is used in various application fields such as robotics, railway applications, building automation, and even subsea systems.

This protocol is called "open" due to three reasons. Firstly, the technology doesn't require payment of any license fee to be used. Second due to allowing both CAN and CANopen devices to be connected to the same network without conflicts, only being necessary that both do not interfere with the message identifiers from each other. At last, CANopen can easily be extended or customized toward a specific application. That's because the protocol only consists of a small set of mandatory functionality and a large number of optional features. Also, these mandatory features only need to be implemented in CANopen-compliant devices. In addition to all of this, CANopen is also expandable and tolerant to future functions, allowing the manufacturers to implement features that are not yet available.

2.2.1 CANopen frame

One of the differences between the normal CAN frame and the CANopen frame is the Communication Object Identifier (COB-ID). It is split into two sections, the first four bits are a function code, and the other seven bits contain the node ID. A representation of the CANopen frame can be seen in Figure 6.

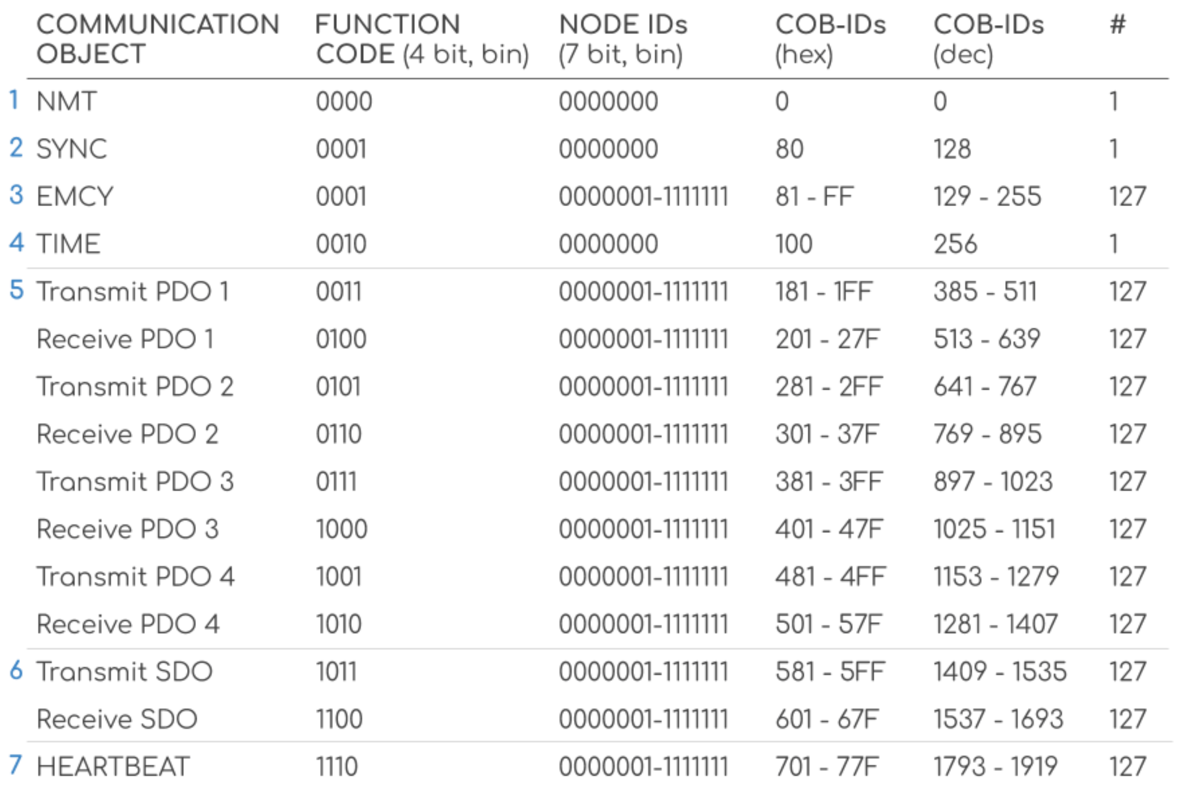
Figure 6 – CANopen frame.



Source: Falch (2022).

To understand the function code, it is necessary to take into consideration a pre-defined allocation table. This table can be seen in Figure 7. For this code, every combination of bits represents one of the CANopen services, which will be explained in future sections.

Figure 7 – Function code table.



	COMMUNICATION OBJECT	FUNCTION CODE (4 bit, bin)	NODE IDs (7 bit, bin)	COB-IDs (hex)	COB-IDs (dec)	#
1	NMT	0000	0000000	0	0	1
2	SYNC	0001	0000000	80	128	1
3	EMCY	0001	0000001-1111111	81 - FF	129 - 255	127
4	TIME	0010	0000000	100	256	1
5	Transmit PDO 1	0011	0000001-1111111	181 - 1FF	385 - 511	127
	Receive PDO 1	0100	0000001-1111111	201 - 27F	513 - 639	127
	Transmit PDO 2	0101	0000001-1111111	281 - 2FF	641 - 767	127
	Receive PDO 2	0110	0000001-1111111	301 - 37F	769 - 895	127
	Transmit PDO 3	0111	0000001-1111111	381 - 3FF	897 - 1023	127
	Receive PDO 3	1000	0000001-1111111	401 - 47F	1025 - 1151	127
	Transmit PDO 4	1001	0000001-1111111	481 - 4FF	1153 - 1279	127
	Receive PDO 4	1010	0000001-1111111	501 - 57F	1281 - 1407	127
6	Transmit SDO	1011	0000001-1111111	581 - 5FF	1409 - 1535	127
	Receive SDO	1100	0000001-1111111	601 - 67F	1537 - 1693	127
7	HEARTBEAT	1110	0000001-1111111	701 - 77F	1793 - 1919	127

Source: Falch (2022).

As for the node ID, it refers to the identification of every device in the network. This implies that every device must have a unique ID in the network, so it is possible to address the messages to the nodes without conflicts.

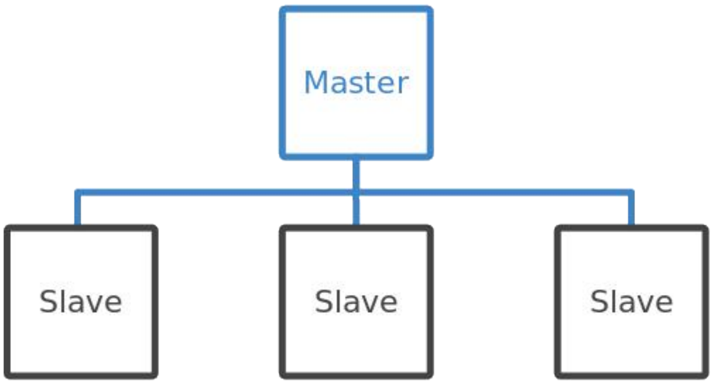
2.2.2 Communication Models

The protocol also supports three communication models between nodes. The possible models are:

1. **Master\Slave:** In this model, the communication is unidirectional, a device (master) controls one or more devices (slaves) and the information is broadcast in the network only from the slave to the master upon the master's request. Also in the CANopen protocol, only one master per slave is allowed to avoid configuration conflicts. A representation of the Master\Slave communication model can be seen in Figure 8.

2. **Client\Server:** Here, the information is held by the Server, and the Client has access to this data by a request. In this model, the information is more centralized, since one node

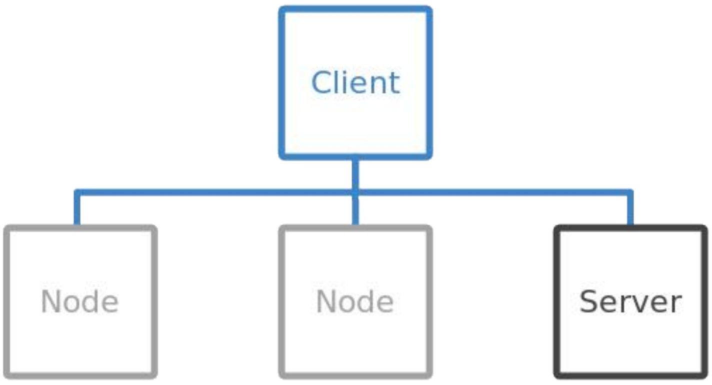
Figure 8 – Master\Slave communication scheme.



Source: Falch (2022).

contains all the information available. Another difference in this type of communication is that multiple nodes can be configured as a Server but only one Client may access the Server at a time. A representation of the Client\Server communication model can be seen in Figure 9.

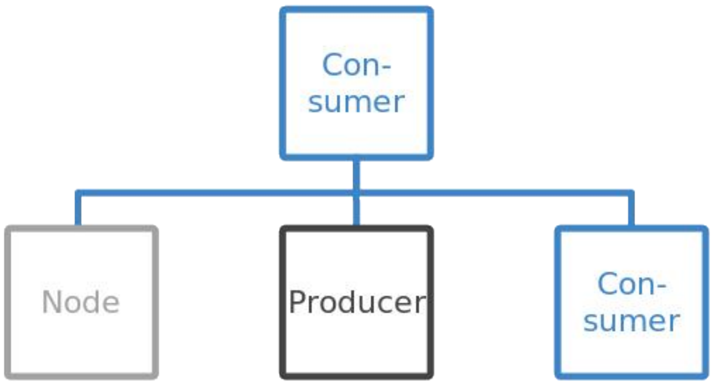
Figure 9 – Client\Server communication scheme.



Source: Falch (2022).

3. **Consumer\Producer:** In this communication model, the Producer broadcasts data independently of the Consumers. Multiple Producers and Consumers can be connected at the same time. The producer can also send data on request (pull model) or without a specific request (push model). A representation of the Consumer\Producer communication model can be seen in Figure 10.

Figure 10 – Consumer\Producer communication scheme.



Source: Falch (2022).

2.2.3 OD

The OD feature is a standardized structure that contains all variables related to a CANopen node. This means that every device has a unique OD that contains a description of its configuration, functionalities, and information. Also, some of the items may or may not be writable or readable by another CANopen device.

The OD works like a table, to have access to the data, it is necessary to send an index and a sub-index as the ODs entries. The index has a 16-bit size, giving a maximum of 65.536 entries, and the subindex has an 8-bit size, which translates to 256 subentries. Although the OD support a large number of entries, it is not necessary to implement all of them.

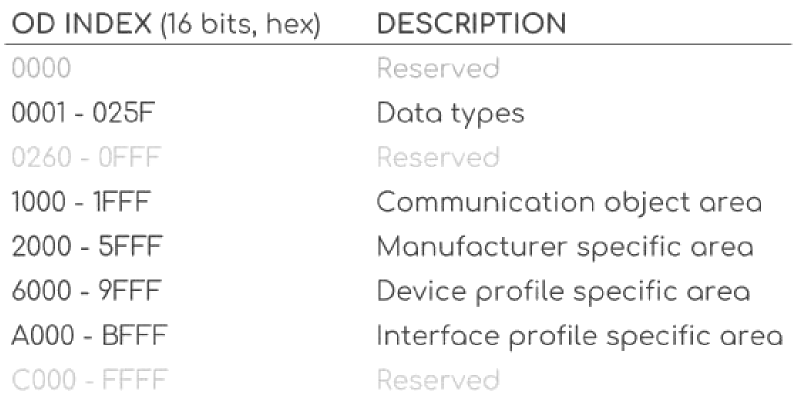
Although every OD is unique, all of them are organized in a standard way. Figure 11 shows how the indexes are organized. Also, when referring to indexes and sub-indexes, it is a common practice to use hexadecimal numbers instead of bits (0001h or 0x0001 for example).

2.2.3.1 0001h - 0FFFh: Data Types

The OD can store standard data types as well as data types set by manufacturers. These data types are all stored in this section of the OD. In addition, it is also specified in the CANopen protocol two basic classes of data types: Standard and Complex. Standard are pre-defined data types normally used in programming languages like Boolean or Integer. As for Complex data types, they are equivalent to the structure type of data used in C, where it is possible to hold a list of variables inside one structure. The internal division of the data types can be seen in Table 2.

- **Standard Data Types:** pre-defined data types like Boolean or Integer.
- **Pre-defined Complex Data Types:** predefined complex data types, similar to structures in the programming language C.

Figure 11 – Organization of the OD.



OD INDEX (16 bits, hex)	DESCRIPTION
0000	Reserved
0001 - 025F	Data types
0260 - 0FFF	Reserved
1000 - 1FFF	Communication object area
2000 - 5FFF	Manufacturer specific area
6000 - 9FFF	Device profile specific area
A000 - BFFF	Interface profile specific area
C000 - FFFF	Reserved

Source: Falch (2022).

Table 1 – Data Type Storage in the OD.

Index Range	Definition
0001h - 001Fh	Standard Data Types
0020h - 0023h	Pre-defined Complex Data Types
0024h - 003Fh	Reserved
0040h - 005Fh	Manufacturer Complex Data Types
0060h - 007Fh	Device Profile Standard Data Types
0080h - 009Fh	Device Profile Complex Data Types
00A0h - 025Fh	Multiple Device Modules Data Types
0260h - 0FFFh	Reserved

- **Manufacturer Complex Data Types:** section where the manufacturer can define custom data structures.
- **Device Profile Standard Data Types:** list of standard data types related to device drofile
- **Device Profile Complex Data Types:** list of complex data types related to the device profile
- **Multiple Device Modules Data Types:** list of data types in case the device applies to more than one profile

2.2.3.2 1000h - 1FFFh: Communication object area

This area describes the functionalities related to the communication of the node. Some of these objects are writable, allowing the configuration of the node by other nodes on the network. Among the entries in this area, there are some that are mandatory and these are the most important of them.

Table 2 – Mandatory objects in the Communication object area.

Index Range	Definition
1000h	Device Type
1301h	Error Register
1017h	Producer Heartbeat Time
1018h	Identify Object

- **Device Type:** describe basic information about this device.
- **Error Register:** indicates the errors that happened in the node.
- **Producer Heartbeat Time:** define the period in which a heartbeat message is sent.
- **Identify Object:** gives some information related to the ID of the vendor assigned by CAN in Automation (CiA).

2.2.3.3 2000h - 5FFFh: Manufacturer specific area

This field is totally linked to the data inserted by the manufacturer, which means that the CANopen specification has left it open for application use. Every parameter or function that must be implemented for the device can be added here.

2.2.3.4 6000h - 9FFFh: Device profile specific area

This section contains the objects defined by the device profile. This means it will include data fields and functionalities that are specific to this node.

2.2.4 CANopen services

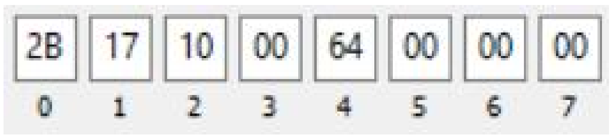
The CANopen protocol is composed of many different services, which provide a large number of facilities and unique features to the bus. In this subsection, it will be given a brief explanation of the main CANopen services.

2.2.4.1 SDO

The Service Data Object (SDO) is a simple method used to access the ODs through client/server communication. It enables the user to send read or write requests to individual indexes of the Object Dictionary. As an example, if it is required to read information from a device, such as its name or serial number, the user should search which node corresponds to this device and send an SDO read request to this node with the index and sub-index related to that information.

When it comes to the data, this type of message can hold up to four bytes of actual data, the rest being used as overhead, meaning that the first four bytes are utilized for index addressing and message specification. Figure 12 is an example of SDO data and below it, the data is explained.

Figure 12 – SDO data example.



2B	17	10	00	64	00	00	00
0	1	2	3	4	5	6	7

Source: Personal file.

- The first number of the byte 0 is the client command specifier, which indicates the type of data on the message.
  - **2:** write request
  - **4:** read request
  - **6:** success message
  - **8:** error message
- The second number in the byte 0 indicates the amount of not used bits on this message.
- bytes 1 and 2 hold the index value backward (0x1017 -> byte 1 = 0x17 and byte 2 = 0x10)
- byte 3 holds the sub-index value.
- bytes 4 to 7 hold the actual data that was read or that will be written.

2.2.4.2 PDO

This type of message differs itself due to not being transmitted upon request like the SDO. Also, it doesn't have four bytes of overhead, which means that the PDO can hold up to 8 bytes of actual data inside one message.

The idea behind this kind of message is that the data that goes into the 8 bytes that are transmitted in a CAN message is manageable. In other words, the user can define, for example, the in one PDO two parameter of 4 bytes, one parameter of 4 bytes and two with 2 bytes, or any combination of data as long as the total length of the data is less than 8 bytes.

When it comes to message transmission, the PDO uses the producer/consumer model and has three main options defined by a parameter called "transmission type": Event-driven, time-driven individual polling, and synchronized polling.

- **Event driven:** this method transmits a message if any of its data has changed. For example, if a PDO is configured to hold the reading of this sensor, every time that the value from this sensor is read, a message is transmitted.

- **Timer driven:** with this method, the messages are transmitted in a defined period of time.
- **Synchronized polling:** with this method, a PDO is sent every time a SYNC message, which will be explained in the Section 2.2.4.3, is transmitted.

At last, this type of message also uses two terminologies: Transmit Process Data Objects (TPDO) and Receive Process Data Objects (RPDO). Simplifying, when only one node is taken into consideration, Transmit PDOs are the process data that are transmitted from this device, and the Receive PDOs are the process data that are received by this node.

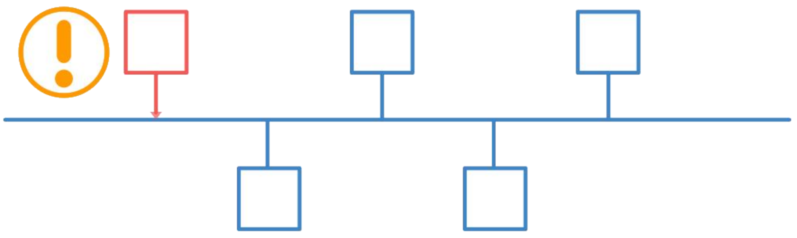
2.2.4.3 SYNC

This SYNC service resumes into messages sent by a node master in order to require a PDO message of the slave nodes that are in the "Synchronized polling" mode.

2.2.4.4 EMCY

The EMCY service is applied in case of a fatal error to the device and this will grant that the network will have knowledge of this problem. A scheme of the EMCY can be seen in Figure 13.

Figure 13 – EMCY scheme.



Source: Falch (2022).

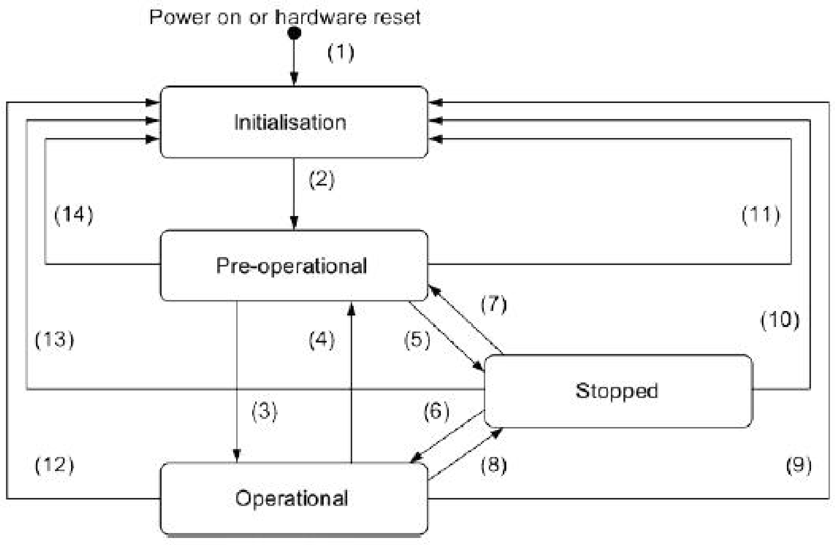
2.2.4.5 NMT

This service is linked to the management of the slave node's operational states. Basically, every CANopen node must implement the NMT state machine that is commanded by a node master and dictates which services are active in this node. Figure 14 it is shown the NMT state machine with a description of the possible transitions and Figure 15 shows which services are active in which NMT state.

2.2.4.6 Heartbeat

This service has the function of sending the information that this node is alive. Apart from notifying the master that the node is "alive", it sends also the current NMT state of this

Figure 14 – NMT state diagram.



(1)	At Power on the NMT state initialisation is entered autonomously
(2)	NMT state Initialisation finished - enter NMT state Pre-operational automatically
(3)	NMT service start remote node indication or by local control
(4),(7)	NMT service enter pre-operational indication
(5),(8)	NMT service stop remote node indication
(6)	NMT service start remote node indication
(9),(10),(11)	NMT service reset node indication
(12),(13),(14)	NMT service reset communication indication

Source: CIA... (2022).

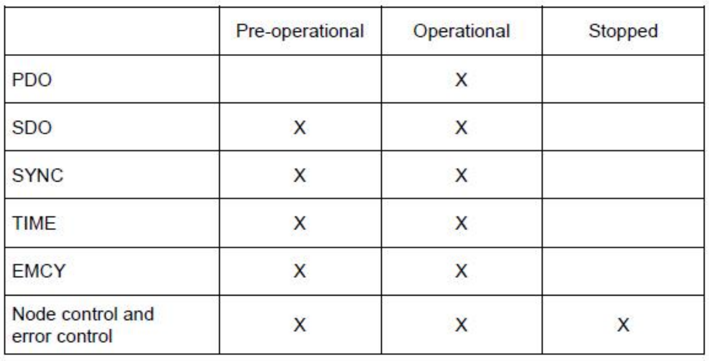
device. It executes both functions by regularly sending messages to another node (which could be a master or a slave) with a code that informs the node status. A scheme of the heartbeat can be seen in Figure 16.

2.2.5 EDS

The EDS file is a file format used to describe the OD of a specific node. It is an electronically readable file that enables tools, such as monitors, analyzers, and configuration tools, to have all the information related to the OD.

One example of these tools is the CiA Conformance Test. This tool is used to verify the general conformance of the device with the CANopen protocol and also test if the device implements all the objects defined in the EDS file. More details about this software will be explained later in this document.

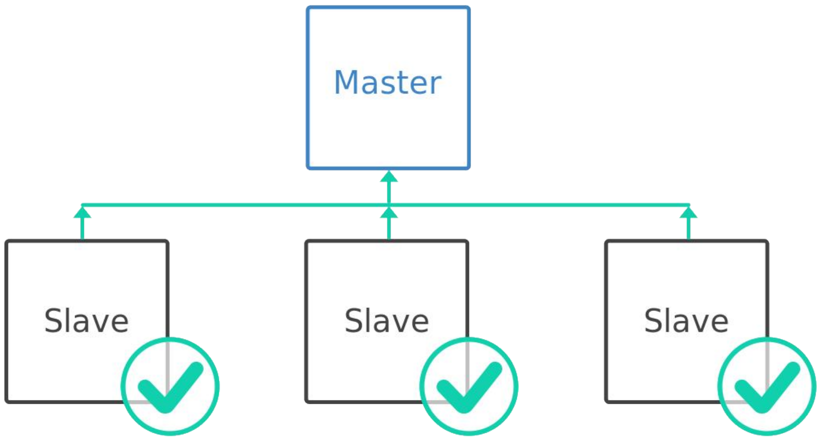
Figure 15 – NMT state definition.



	Pre-operational	Operational	Stopped
PDO		X	
SDO	X	X	
SYNC	X	X	
TIME	X	X	
EMCY	X	X	
Node control and error control	X	X	X

Source: Automation (2021).

Figure 16 – Heartbeat scheme.



Source: Falch (2022).

2.2.6 LSS

In short, the Layer Setting Services (LSS) allows you to enter information and change the parameters of a CANopen bus like the bus bit rate and set up the nodes configurations.

2.3 NORMS AND STANDARDS

In this section, we will explain the norms and standards used for the project and clarify the correlation of these documents with the project.

2.3.1 CiA

As it is clarified in CIA... (2022), CiA stands for CAN in Automation and is a group of users and manufacturers with the goal of providing a platform for CAN-related specifications and standards. This organization has support from International Organization for Standardization (ISO) and International Electrical Committee (IEC) and European Committee for Electrotechnical Standardization (CENELEC), which are three standardization organizations and committees. Together, they create documents that define standards for different types of markets, including industrial automation, road vehicles, medical devices, and more.

Among all the documents from this organization, the ones that are the most important for this project are:

- **CiA 301:** CANopen application layer and communication profile.
- **CiA 302:** CANopen additional application layer and communication profile.
- **CiA 303:** Device and network design.
- **CiA 305:** CANopen layer setting services (LSS) and protocols.
- **CiA 443:** CANopen profile for SIIS level-2-devices.

2.3.2 SIIS

The SIIS is a group established in 2003 by OTM Consulting with the objective of creating an open standard for the oil and gas industry. The group has generated recommended practice document that defines three levels of instrument interface protocols for communication between subsea control modules and subsea sensors (SIIS..., 2022):

- **Level 1 - Analog Devices:** These are sensors that work with 4-20mA and sensing devices with two wire loop powered analogue output
- **Level 2 - Digital Serial Devices:** These are complex sensors or instruments that have a digital connection with the main control system. The communication protocol used is CANopen (CiA 443) with an ISO 11898-3 (fault tolerant) physical layer.
- **Level 3 - Ethernet TCP / IP Devices:** These are devices that are used with Intelligent Seabed Devices (ISD). They require direct communication access between the ISD Surface Application Systems (ISAS) and the actual ISD.

2.3.3 API 17N

As it is described in Institute (2017), American Petroleum Institute (API) 17N is the short name for *API Recommended Practice 17N, Recommended Practice on Subsea Reliability,* *Technical Risk, and Integrity Management*. This document, as the name implies, states recommended practices for operators, contractors, and suppliers for guidance on the management and application of subsea projects.

The paper includes practices in many different aspects of a subsea project, such as document formats, project development methodologies, and evaluation procedures. The practices that are related to this report will be explained later in this document.

2.3.4 TRL

The TRL is a method created by NASA to estimate the maturity during the development phase of a technology, meaning that it evaluates how ready is the product for the real scenario. The NASA levels are classified from 1 to 9 being level 1 the observation and report of the basic concept and level 9 the finished system validated in an operational environment (TWI, 2021).

Nevertheless, even though the TRL qualification method was designed to be as wide as possible, applying it to a subsea system isn't as easy as it sounds, given the extreme scenarios in which the devices are submitted. That's because of the high pressure that exists in the subsea, the underwater context, and other factors that could be damaging to the instruments involved in the operation. Therefore, the original TRL doesn't match all the challenges that are encountered in the subsea environment.

For this reason, the team follows a slightly different interpretation of the original TRL, which is provided by the API 17N. It considers the subsea scenario for measuring the TRL levels. In this version, which is explained in Institute (2017), TRL levels are determined from 0 to 7, equivalent to levels 1 to 9 from NASA. A basic explanation of each TRL level from the API 17N can be seen right below.

- **TRL 0-Basic Research:** this level is related to the basic conception of the project idea, including the basic principles of the technology, fundamental requirements, review of similar technologies, etc.
- **TRL 1-Concept Selection:** here it is required evidence to prove that the concept can be used in the desired application. It should include a formulation of the concept and potential applications, functionality demonstration by analysis, preliminary assessment of fit, etc.
- **TRL 2-Concept Demonstration:** consists of demonstrating that the concept is valid and has the potential to function in its intended environment. It takes into account the confection of the initial Failure Mode, Effects, and Criticality Analysis (FMECA), demonstration of the technology functionality with physical models or lab "mock-ups", the performance of some material testing with the degradation mechanisms, etc.
- **TRL 3-Prototype Development:** at this stage a prototype should be built and subjected to basic functionality and reliability testing. The prototype should go through some functional and performance tests, highly accelerated life tests, and reliability tests.

- **TRL 4-Product Validation:** at this level, a prototype should be manufactured and assembled using the procedures defined for the actual production item. Also, the tests should be made in an environment equivalent to the environment in which the device was designed, even though the technology wasn't used in the actual intended operating environment.
- **TRL 5-System Integration Testing:** here it is required that the device is ready to be incorporated into its intended system. This includes performing tests while the device is connected to the wider system and addressing the technology interfaces (mechanical, software, electronic, etc).
- **TRL 6-System Installation and Commissioning:** the technology will be installed in its final operational environment and all the TRL 6 tests will be executed.
- **TRL 7-System Operation:** The device is installed in the intended environment for enough time to prove its reliability.

2.4 ACTUATORS

Since this project is related to subsea actuators, it is relevant to understand the characteristics of normal actuators. In the sequel, we discuss the basic functioning of hydraulic and electrical actuators as well as their features.

2.4.1 Hydraulic Actuators

As explained in SIT (1991) and in Yong Bai and Qiang Bai (2010), the function of hydraulic actuators is to convert hydraulic into mechanical power. Also, among their advantages, they can produce high values of force, and they can be easily controlled by an electronic system. But, when it comes to their disadvantages, one of the main problems with hydraulic actuators is that it is necessary to take care of its pollution due to leaking, and it is required purity maintenance of the operating oil.

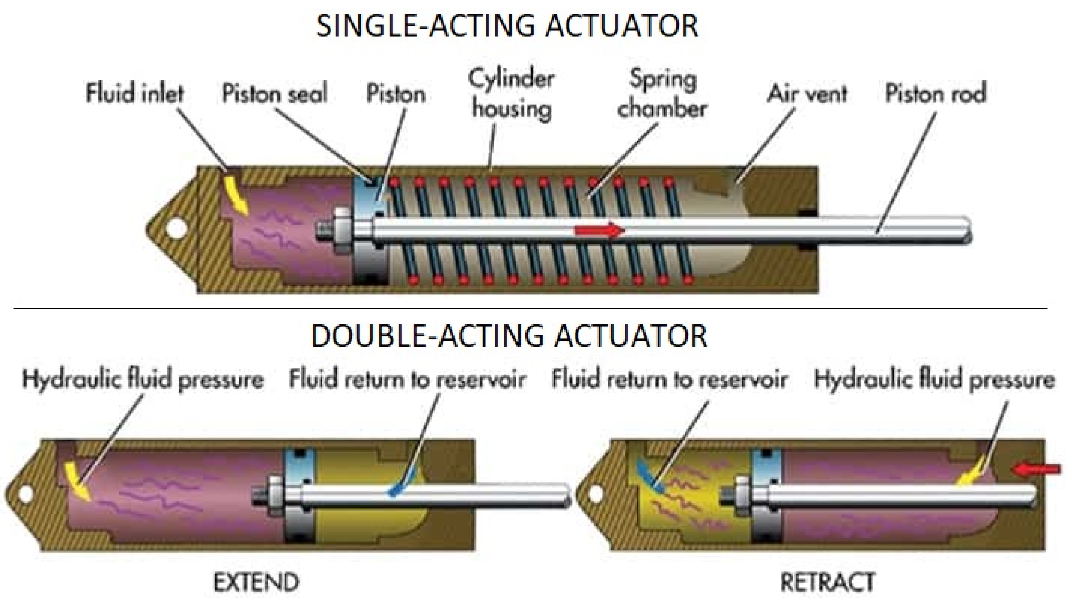
2.4.1.1 Hydraulic Linear Actuators

The single-acting linear actuator can also be called a "hydraulic cylinder". Among hydraulic linear actuators, there are two classifications of them: single-acting and double-acting. Both types of actuators have two chambers and inserting fluid into one of the compartments will create the movement.

When it comes to their functionality, it has one chamber with an inlet, that allows the fluid to come inside this compartment and moves the piston in a linear direction. In the other chamber, there is a spring, that will push the fluid out of its compartment when it is necessary. That's why it is called a 'single-acting' actuator because the only action made is to input fluid into its chamber.

In relation to the double-acting actuator, it uses both chambers to insert fluid. If the piston needs to move forward, the fluid is inserted into one side of the chamber and the oil or gas is expelled from the other chamber. But in case the piston needs to retract, the fluid is added on the other side of the chamber and the material is expelled from the first compartment. Given that for this actuator to work, it is necessary to act on both sides of it, it is called 'double-acting'. Figure17 illustrates the operation of both types of actuators.

Figure 17 – Operation scheme of single and double acting actuators.



Source: Thorat (2023).

2.4.1.2 Hydraulic Rotary Actuators

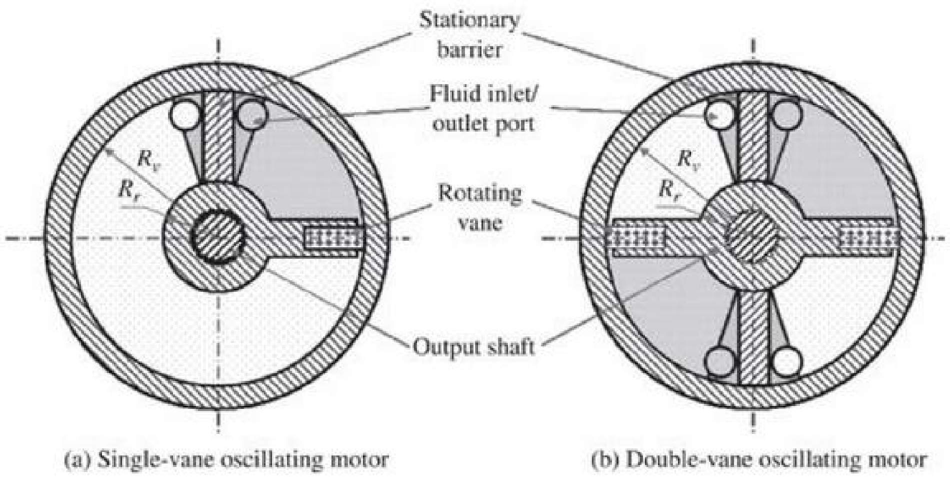
As it is detailed in Zahng (2018) This type of actuator works with the same principles as the linear actuator. This implies that its movement is created by inserting fluid into chambers, but instead of, single and double-acting options, this type of actuator works with single or double-vane. Figure 18 illustrates the configuration of both types.

In a single-vane actuator, there is an expendable chamber that expands when the fluid is inserted into its inlet achieving up to 280°. The double-vane actuator, which has two fluid chambers separated by two inlets, has a reduced range of less than 100°but it produces twice the torque.

2.4.2 Electric Actuators

An electric actuator, as well as a hydraulic one, converts power into motion, but instead of hydraulic power, it uses electric power.

Figure 18 – Illustration of the configuration and operation principle of a typical (a) single-vane and (b) double-vane type oscillating rotary motors.

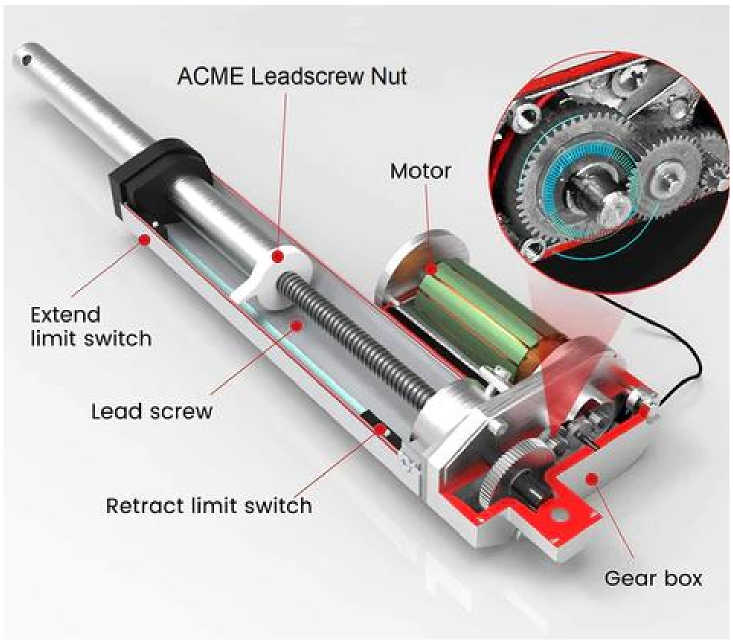


Source: Zahng (2018).

2.4.2.1 Electric Linear Actuators

As explained in Xiang (2021), its operation is based on an electric motor that is attached to it. As Figure 19 shows, the idea of this device is to transform the rotation of an AC or DC motor into linear motion. It does this with a series of gears that will rotate a lead screw and push the main rod in and out.

Figure 19 – Scheme of a electrical linear actuator.



Source: Dickson (2018).

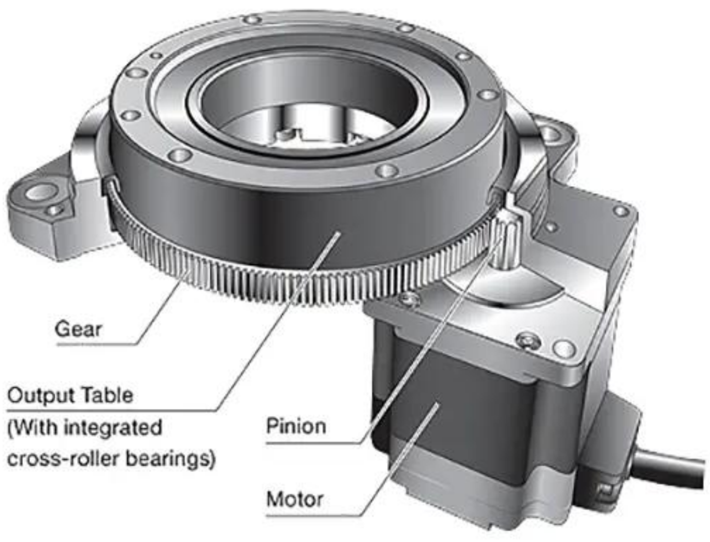
Among its features, stands out its preciseness, meaning that it has the ability to stop at any position with good accuracy and they tend to be small and require little maintenance. Their biggest disadvantages rely on their robustness, due to the possibility of overheating, and their high cost.

2.4.2.2 Electric Rotary Actuators

According to Tips (2021), electric rotary actuators share the same features as the electric linear actuator, implying that this variant is also more precise and requires less maintenance than the hydraulic but is also less robust, due to the overheating possibility.

The difference in its operation, when compared to the linear actuator, is that instead of rotating a lead screw and pushing the main rod, the rotary actuator will rotate a pinion gear coupled to a motor. Figure 20 illustrates the functioning of an electric rotary actuator.

Figure 20 – Scheme of an electric rotary actuator.



Source: Tips (2021).

3 PROBLEM DESCRIPTION

In the present chapter, we will describe the problem treated in this work as well as the associated technical requirements.. Section 3.1 describes the history of Bosch Rexroth up to the group in which this work was developed, Section 3.2 explains the subsea production system and its components, Section 3.3 describes the general context of the main issues approached in this work, Section 3.4 elaborates in more detail the problems that are involved in the main problem, and Section 3.5 outlines the technical requirements associated to it.

3.1 OVERVIEW OF BOSCH REXROTH

The Bosch company started in 1886 when Robert Bosch opened a Workshop for Precision Mechanics and Electrical Engineering in Stuttgart. One of the products that stood out at the time was a magnet ignition device that Bosch was the only supplier of. But it was their next solution, a high-voltage magnet ignition system, that made Bosch set on the track of being a world-leading automotive supplier.

Currently, Bosch is not only a reference in the automotive industry, but it has also become a leading global supplier of technology. Its operations are spread into four areas: Mobility Solutions, Industrial Technology, Consumer Goods, and Energy and Building Technology. The company is very strong in the area of smart homes, industry 4.0, and connected mobility.

Since the company has this massive size, there are a lot of subsidiaries that belong to the Bosch Group such as Rexroth, the company in which this project was made. Rexroth started as a foundry in the city Lohr am Main, Germany, and in 1950 it entered the hydraulics market. After many expansions, in 2001 this company became part of the Bosch Group and established itself as a global leader in Drive and Control solutions.

Nowadays, Rexroth technology can be found in many different places, from Factory Automation to Industrial Hydraulics, and the company focuses a lot on research to always come up with new technologies. To demonstrate how focues the company is in regards to research, in the year 2021, the firm spent €344 million in R&D (REXROTH, 2021).

Among the numerous research groups of Bosch Rexroth, there is the PJ-SAS, the group in which the project presented in this paper is developed. The abbreviation stands for *Project Subsea Automation System* and it is the only group approaching the subsea area in Rexroth. The PJ-SAS was created with the objective of developing the *Subsea Valve Actuator* or SVA (which will be explained in the Section 3.2.4).

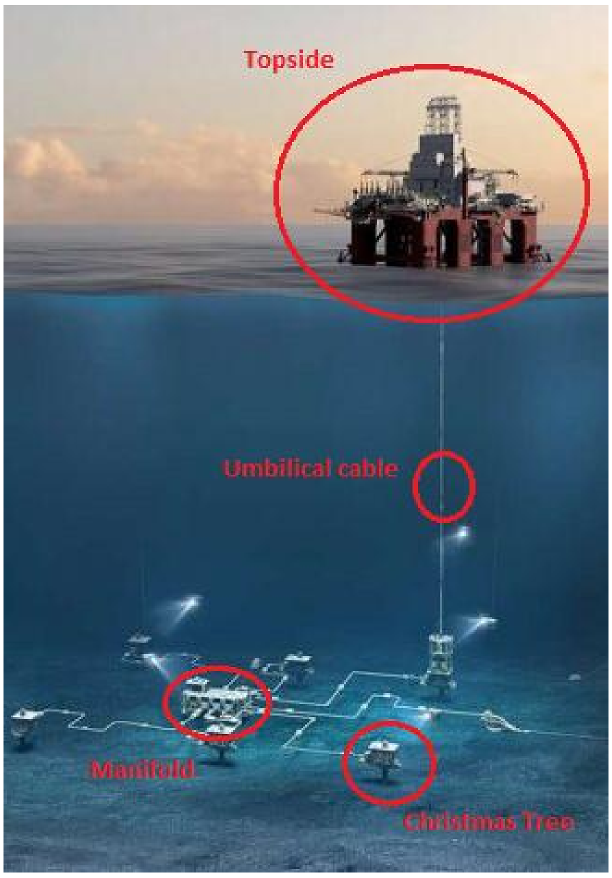
3.2 SUBSEA ACTUATORS AND SUBSEA PRODUCTION SYSTEM

As this project is based on an underwater environment, it is crucial to understand the components found in it. In this section, we will explore the main elements of the draining oil production system.

3.2.1 Subsea Production System

As Claus (2022) describes, draining oil from a wellhead is a task that requires the combination of different components working together. Basically, the oil path begins at the Christmas Tree, goes through the Manifold, and is sent to the Topside through an Umbilical cable. An overview of these systems can be seen in Figure 21, and they are explained in the sequel.

Figure 21 – Subsea production system.



Source: Jiang (2018).

3.2.2 Topside

The topside involves all the systems that are above water, meaning that it encompasses from the power units to the worker accommodations. All the components are designed to be modular and installed onto a floating or fixed structure. This part of the production system is responsible for controlling the entire offshore process and feeding all of its components (CLAUS, 2022). Figure 22 shows a picture of an offshore topside.

Figure 22 – Topside picture.

Source: SUBSEA… (2023).

3.2.2.1 Umbilical Cable

This part is responsible for making the connection between the topside and the subsea. It carries not only cables that power the equipment, but also hydraulic, chemical, and information cables, which makes this part of the system crucial for the proper functioning of the subsea elements. Regarding the information cables, they enable commands to be sent from the controller unit above water to the underwater equipment and enable also readings of sensors to be sent back to the topside (SILVA, 2019; CLAUS, 2022). Figure 23 shows the cross-section of an umbilical cable.

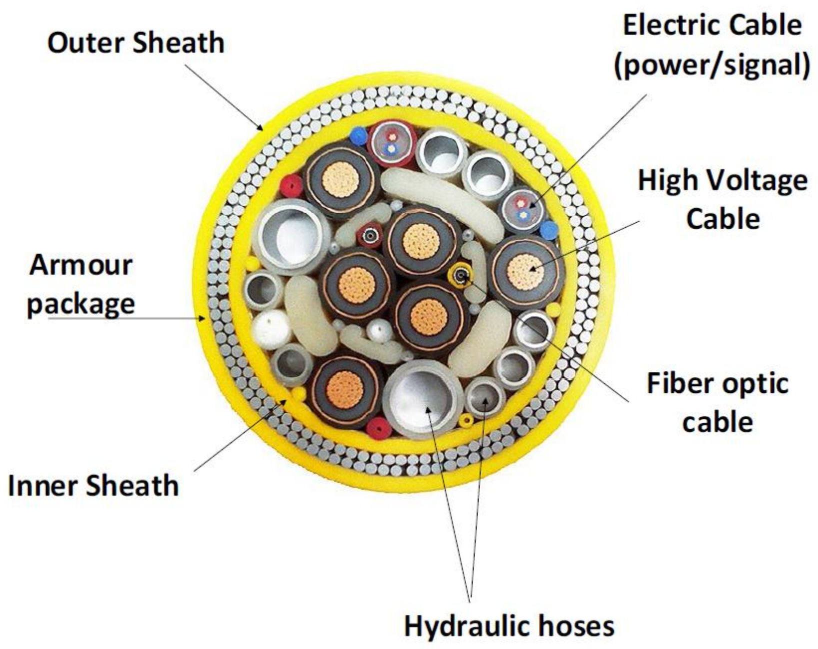
3.2.2.2 Manifold

This equipment is used to distribute the cables from the umbilical cable to the different components. They are important to minimize the use of subsea pipelines and optimize the flow of fluid in the system (SILVA, 2019). Figure 24 shows a Manifold connected to some "Christmas Trees".

3.2.2.3 Christmas Tree

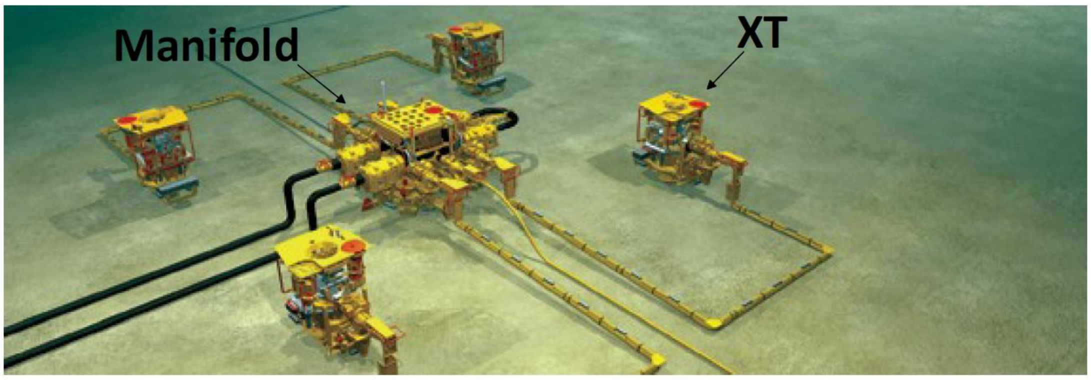
This is the device that is directly connected to the wellhead. It is called "Christmas trees" or Xmas Tree (XT) due to the resemblance between older models of this tool and the decorated tree at Christmas. It's composed of basically an assembly of fittings, spools, and valves, that are connected to the well and provide control over the drained fluid. An image of a subsea Christmas tree can be seen in Figure 25. It is not possible to operate them manually, as a result of needing to operate in Deep or Ultra-Deepwater (between 1830 and 3000 meters) under the sea. To solve

Figure 23 – Umbilical cross-section.



Source: Silva (2019).

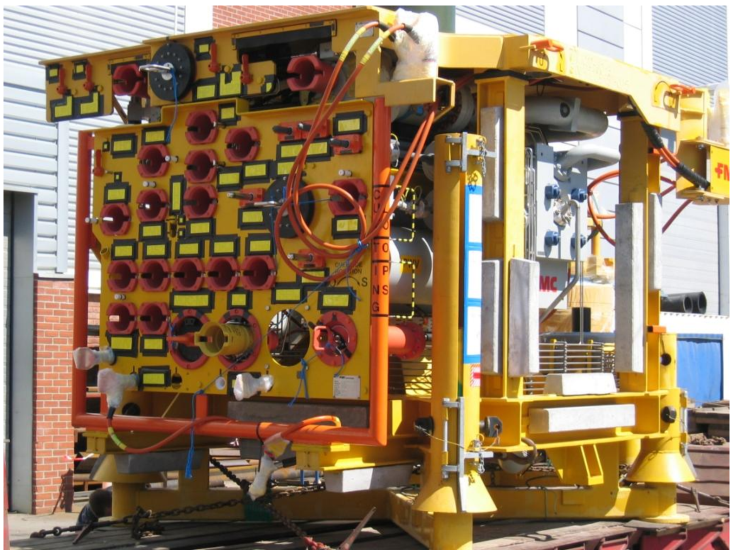
Figure 24 – Manifold .



Source: Panjaitan (2017).

this problem, each XT has Subsea Electronics Module (SEM) and a Subsea Control Module (SCM) installed on it. The first is responsible for communicating with the main control station at the topside and the second controls the well during the production phase of subsea oil and gas. It does so by sending commands to the subsea actuators that control the XTs valves (SILVA, 2019).

Figure 25 – Subsea Christmas tree.

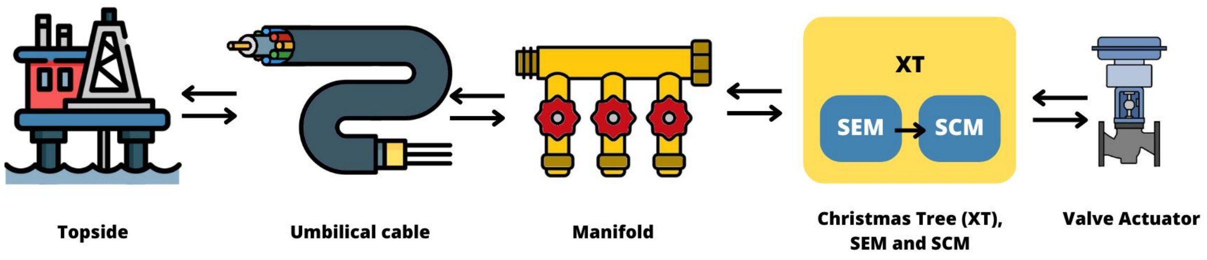


Source: WHY. . . (2022).

3.2.3 Subsea Actuators

As it was said previously, the main function of subsea actuators is to manage the XTs valves. The communication with these devices is made by sending the signal from the topside, passing through the umbilical cable, being handled by the SEM, forwarded to the SCM, and finally directed to the subsea actuator. Figure 26 shows a scheme of communication in the subsea production system.

Figure 26 – Subsea production system scheme.



Source: Personal file.

Besides the commands from the topside, the umbilical cable includes everything necessary for all the instruments underwater, for example, hydraulic lines, chemical fluid lines, and electric lines. Within these electrical lines, there are the communication cables - like industrial Ethernet or OPC-UA - which are responsible for sending commands to the SCM, and it forwards the command through the CAN cables to the actuator. With this connection, the user is able to send the commands to open or close the valves from the Christmas Tree and read the values from the actuator's sensors. These commands will be received by an embedded Programmable Logic Controller (PLC) and will be handled according to the type of actuator, to result in the rotation or the unidirectional stroke of the actuator.

Currently, in the subsea scenario, there are two main types of subsea actuators (CLAUS, 2022; SILVA, 2019): an All-Electric Control System and a Multiplexed Electro-Hydraulic Control System. Both of them have unique features but have the same principle, they are attached to a Christmas tree to manage its valves and they receive commands from the topside through the messages sent from the SCM.

- **Multiplexed Electro-Hydraulic Control System**

  This type of actuator is the most used in subsea scenarios. It requires the constant injection of oil and other fluids through the umbilical cable. Among its advantages, there is the short-time response, long-distance use possibility, and reliability due to having a more established technology for fail-safe (BAI, Y.; BAI, Q., 2010; SOTOODEH, 2019). As MacKenzie, Halvorsen, and Vedeld (2020) explains, a fail-safe mode is an ability to go to a safe state, that could be opened or closed, in case of any failure like loss of energy or loss of hydraulic power.

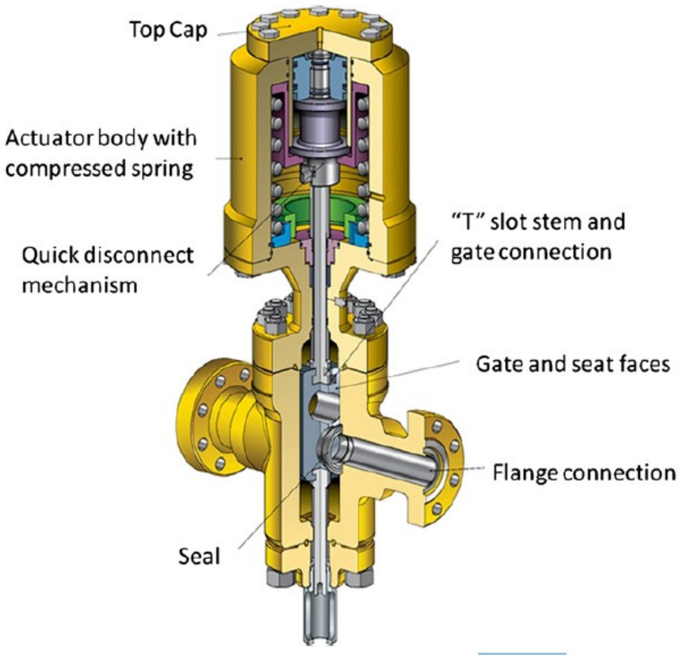
  However, this controller has a high level of complexity, it requires a high level of fluid cleanness and periodic fluid recharging since it commonly releases hydraulic fluid into the ocean (BAI, Y.; BAI, Q., 2010; SILVA, 2019). Figure 27 shows an example of a hydraulic actuator from Magnum.

- **All-Electric Control System**

  As its name says, this type of actuator works only with energy, not needing oil to operate. This indicates that it won't have to deal with leakage of fluids, the umbilical cable for this device will be smaller than the cable for a hydraulic actuator, as it is depicted in Figure 28, and it will be more environmentally friendly. Also, it is a simpler system in comparison to conventional actuators and is ideal for environments with high-pressure and high temperature, due to not needing hydraulic fluids.

  On the other hand, this type of device requires a mechanism to ensure the continuous open position of the production valve in which the actuator is idle. Besides this, in regards to its fail-safe functions, it is necessary a lot of extra equipments to realize these functions. Due to these and other problems, this type of actuator is generally not used as a well safety equipment, it is used more as a processing subsea equipment (ORTH; NEDO; GOTTFRIED, 2022; BAI, Y.; BAI, Q., 2010; SILVA, 2019). Figure 29 shows an example of an All-Electric Control System

Figure 27 – Hydraulic subsea actuator from Magnum.


Top Cap
Actuator body with compressed spring
"T" slot stem and gate connection
Quick disconnect mechanism
Gate and seat faces
Flange connection
Seal

Source: **subseaStrcturalEngineeringempty citation**.

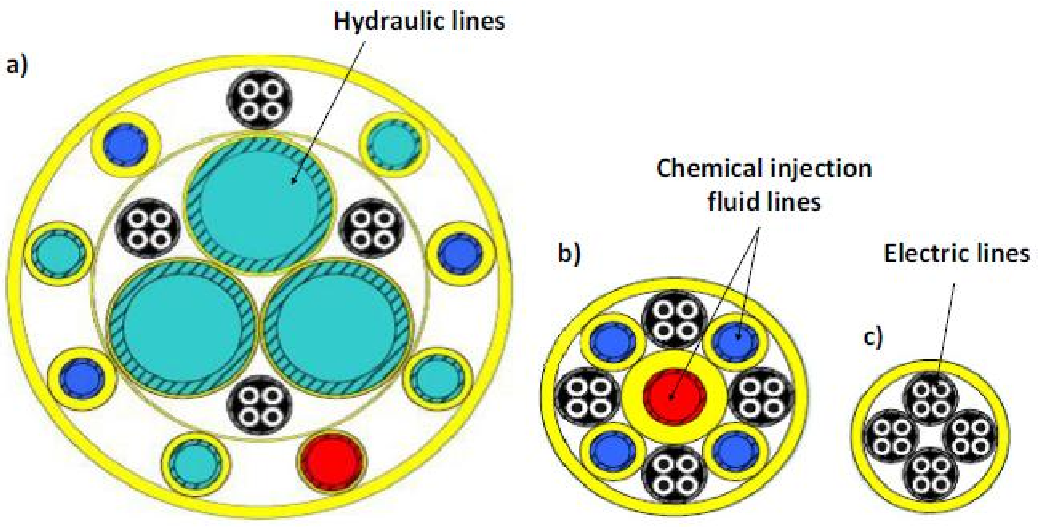
3.2.4 Subsea Valve Actuator (SVA)

SVA stands for *Subsea Valve Actuator* and it is a device that is being developed by the PJ-SAS Group from Bosch Rexroth. This is also a subsea actuator, but it is the world's first electric actuator with a safety spring, as compact as hydraulic cylinders (SVA-R2. . . , 2022). It means that this device has features from both hydraulic and electric actuators.

To be more specific, the instrument interface with the Christmas Tree is purely electric. This guarantees all the advantages of an All-Electric Control System, including the smaller umbilical cable, its simplicity, being environmentally friendly, etc. The difference is that this device contains internally a passive hydraulic system used for pressure compensation and to detect any possible leakage outside. Also, it contains a fail-safe system that is more compact than the normal electric fail-safe systems, as shown in Figure 31 where the gray area shows the equipment necessary for a fail-safe system in an electric actuator and the blue shows the fail-safe system implemented in the SVA (SILVA, 2019; CLAUS, 2022).

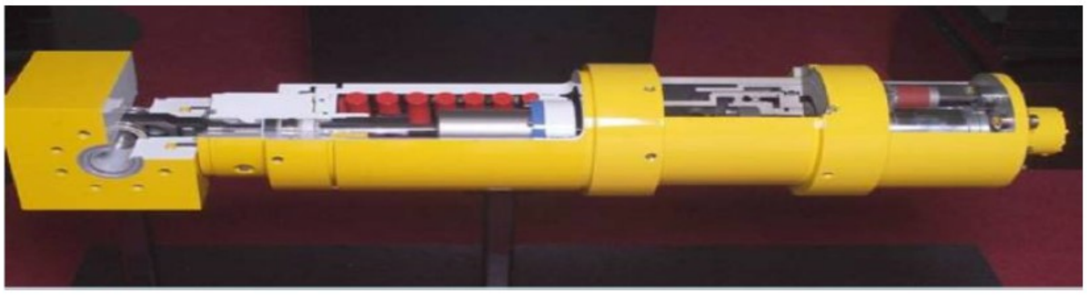
Currently, the actuator has two main versions, one that executes a unidirectional stroke (SVA-L2) and the other one that executes rotation movement (SVA-R2). The rotational version of the actuator will be the main focus of this report. A general look of the SVA-R2 can be seen in Figure 30.

Figure 28 – Illustrative comparison between the cross sections of traditional electro-hydraulic and all-electric umbilical cables.



Source: Silva (2019).

Figure 29 – Example of a All-Electric Control System.



Source: Berven (2013).

3.3 CONTEXT AND MAIN PROBLEM

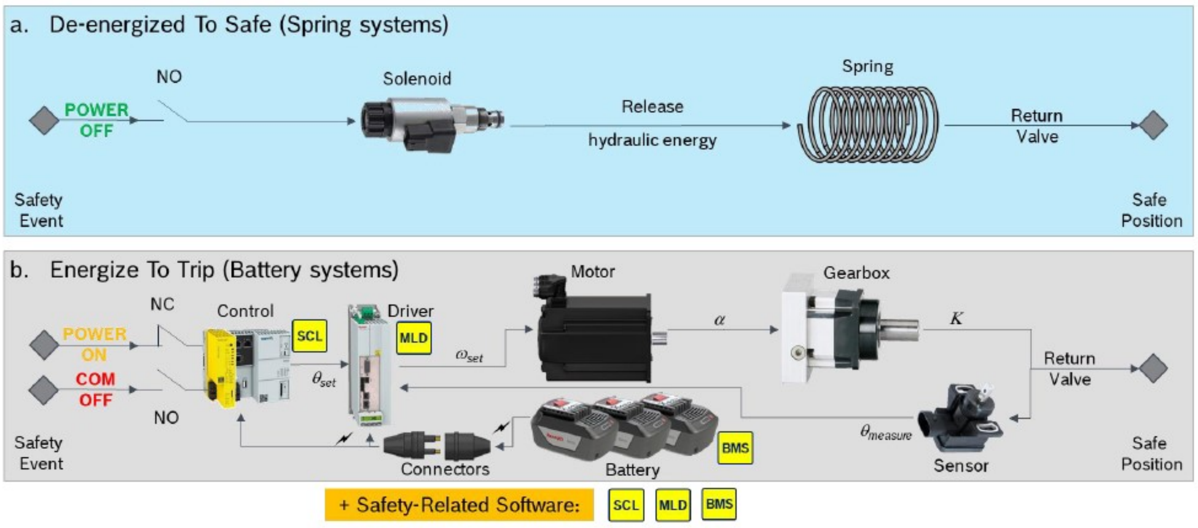
As explained in Section 3.2, for a gas and oil production system to work, it is necessary that many different components need to work together. One of these components is the "Christmas Trees" or XT for short, which is used for draining wellheads. These "Christmas Trees" are managed by internal valves, that are controlled by the subsea actuators. Currently, on the market, there are two main types of subsea actuators, hydraulic actuators, and electric actuators. Both of them have advantages and disadvantages over the other.

The hydraulic is more used for the subsea scenario, due to its already well-established technology and more reliable fail-safe functions. However, the use of oil brings some issues like the need to clean the oil and the requirement of constant oil and chemicals injection through the umbilical cable. This makes the cable very thick and liable for defects. In addition, another

Figure 30 – General look of the SVA-R2.

Source: SVA-R2... (2022).

Figure 31 – Fail-safe systems comparison (a) SVA (b) electric actuator.



Source: Orth, Nedo, and Gottfried (2022).

drawback of this instrument is the environmental damage caused by the discharge of the oil and chemicals into the ocean after being used.

A possible alternative to this controller is the electrical actuator. This device does not require the injection of oil and chemicals into its system, implying that it is more environmentally friendly and it is a simpler system. Nevertheless, one of the main issues of this device is that it isn't simple to implement fail-safe functions. This results in this device being rarely used as safety equipment.

Given this situation, the Bosch Rexroth group PJ-SAS is developing the *Subsea Valve Actuator*, or SVA, a device that has the best features from both types of actuators. It is an electric actuator, meaning that it has all the advantages of a standard electric actuator. The difference is that it contains a fail-safe technology ensured by springs, which is more compact than the normal fail-safe technologies for electric actuators, eliminating one of the biggest problems of this type of device.

The device is in the development process and it is not ready to be placed on the market yet. To be sure that the product is ready for commercialization, it is crucial to seek validation in most of its aspects. That includes guaranteeing that the performance requirements will be met, that they will be ready to be implemented in real scenarios and the possibility of any failures that could be compromising will be minimized.

For that reason, the project uses the TRL classification, to validate the progress of the development process. Remembering what was detailed in Section 2.3.4, this is a categorization standard created by NASA, that indicates from levels 1 to 9 how ready a project is to be released into the market. However, this classification standard didn't take into consideration the issues involving the subsea scenario, like the high pressure that exists underwater, the underwater context, etc. For this reason, API 17N defines a different version of the TRL, which uses levels from 0 to 7 instead of 1 to 9. Recapping the Section 2.3.3, API 17N is a document that contains many recommended practices for subsea projects, including project methodologies, document formats, and so forth.

Currently, the project is in the process of certificating the TRL 4, which stands for testing the product in an environment equivalent to the scenario that the technology was designed for. So the next main problem of the project consists of evaluating if the TRL 5 was achieved, for this, it is necessary to verify all the interfaces of the technology (mechanical, electrical, software, etc) with its intended system. However, testing all those interfaces is a complicated task, because it involves the operation of many different areas of the project that currently cannot afford to focus on this regard. For this reason, the team decided to check the interfaces one at a time. The first one that was chosen to be verified was the software communication interface and that is where this report comes in evaluating if the TRL 5 was achieved.

3.4 SPECIFIC OBJECTIVES

However, to approach the main problem, we consider the following specific objectives:

1. Determine the necessary requirements to achieve the TRL 5;
2. Develop the tests to obtain TRL 5;
3. Create an environment suitable to produce the tests;
4. Determine the methodology to evaluate if the TRL 5 was attained;

5. Evaluate if the TRL 5 was indeed obtained;
6. Automate the test procedure.

3.5 REQUIREMENTS

Besides coming up with a solution, the project needs to meet some requirements, which are described in the sequel.

3.5.1 TRL 5 requirements

As previously described in Section 2.3.4, the TRL has a main specification and some requirements for achieving its levels. The main requirement of this level is the final product to be developed should be incorporated into its intended system. Full interface and function testing should be completed before it is placed in its intended environment" (INSTITUTE, 2017). Regarding the other specifications, according to Institute (2017), they are:

1. perform interface FMECA;
2. perform function and performance tests when integrated with the wiser system, noting that System Integration Testing (SIT) activities are not generally performed subsea;
3. address mechanical, hydraulic, optical, electronic, software, ROV/tooling, and human interfaces;
4. confirm product SIT;
5. initiate performance/reliability data collection;
6. update system reliability assessment;
7. estimate reliability and residual technical risk and uncertainty.

3.5.2 SIIS requirements

As already explained in the Section 2.3.2 the SIIS is a group that establish standards regarding subsea technology and this means that the SVA needs to follow its standard as well. The group separates the interface communication in three levels, in which level 1 are analog devices, level 2 are digital serial devices and level 3 are Ethernet TCP devices. The one related to this work is the Level 2 and we will explain the requirements that comes with this level. Also, we will clarify the technical aspects involved in these requirements.

1. Use CAN as the network protocol.
2. Use CANopen protocol.

3. Use Low-Speed CAN as the CAN physical layer.

4. Operate with the bit rates 50 kBit/s and 125 kBit/s.

Recapping Section 2.1, CAN is a type of bus commonly used in automobiles due to its reliability. For being a very safe bus, SIIS obligates every subsea device to use this type of network.

As also explained in Section 2.1.1, the CAN bus has two physical layers, Low-Speed CAN (or Fault-Tolerant) and High-Speed CAN. The High-Speed CAN is used more often on the market but SIIS requires that every subsea device use Fault-Tolerant CAN, due to being more robust and, in subsea scenarios, this is very relevant since it is not trivial to make maintenance in a kilometer-long cable under the sea. This resilience is related to the possibility of working with only one of the two CAN cables - in case of any rupture.

As it was explained in Section 2.2, CANopen is a protocol that offers many features like specific services and memory organization, that provide a well-established environment for the device's communication.

4 PROPOSED SOLUTION AND USED METHODOLOGIES

Given the scenario and the problems explained in Chapter 3, the proposed solution to solve the issues and meet the API 17N requirements will now be described in this chapter. In Section 4.1 we explain all the steps and layers of the proposed solution, including the TRL 5 interpretation, the test elaboration, the automatic tests, the explanation of the results methodology, and summarizing with an illustration of the overview the proposed solution, in Section 4.2 we describe other methodologies used throughout the project.

4.1 GENERAL IDEA

This section will explore various aspects that were considered in the development of the proposed solution, from the interpretation of the TRL requirements to the solutions methodology.

4.1.1 TRL 5 Interpretation

As previously stated in Section 3.5.1, the main request of this level is that the device should be incorporated into the purposed system and its interfaces and functions should be tested before being incorporated into the real scenario. Keeping that into consideration, the strategy to fulfill the specifications rely primarily on testing the interface between the technology and the intended system. Besides this, even though this level requires all types of interfaces to be tested (mechanical, electrical, hydraulic, etc), this proposal is going to focus only on the software/communication interface. Also, the solution is not going to cover every small requirement listed in Section 3.5.1, given that they can be developed after the main goal is achieved, for example, the interface FMECA is made after the interface is finished.

After having the idea set, it was necessary to understand how the connection of subsea devices works. As it was said in Section 3.5.2, one of the requirements of this standard is to use CAN and CANopen of the communication protocol. This indicates that every subsea device in a real scenario uses these protocols for their communication, in other words, if it is necessary to evaluate the link of the device in a real scenario, the tests should be related to testing the CAN and CANopen performance of the technology.

4.1.2 Tests Elaboration

Now with that goal in mind, the next step was to elaborate the tests that could evaluate the CANopen interface of our device. For that, we used the SIIS Level 2 list of tests. Recapping Section 2.3.2, SIIS is a group that defines standards for technologies in the oil and gas industry. They classify technologies based on their communication method and the Level 2 are devices that use CANopen. But to get this classification, it is necessary that the device goes through a series of tests that verify all the main CANopen aspects of the technology. For that reason, we choose this test list as our main TRL 5 test list, thus we can test the key aspects of CANopen that could be used in a real scenario.

Besides the tests from SIIS, we used the CiAs CAN CTT, to give more confidence in the end result. Summarizing the Section 2.3.1, CiA is a group that provides specifications and standards for CAN and they disclose this tool that tests the communication interface from a device. Subsequently, we chose to add the *CAN Conformance test tool* to our procedure.

Although we had already a good set of tests, it would be a significant benefit if we had some input from someone that had already operated in a real-life setting, given that the project has no real-world exposure. Due to that, we planned a conversation with a company partner to our project, in this way, we could be closer to what the intended environment would expect.

4.1.3 Automatic Test Cabinet

Although our main problem does not concern the automation of those tests, the team thought that would be a great addition to the report if it was possible to do the tests automatically. This way, we could reproduce the tests easily and quickly in our SVA and in other devices as well. For this reason, we came up with the idea of a modular test setup, that is able to reproduce the SIIS tests in any device that is connected to it and generate a report with the test results. Since the CTT is a software for itself, the idea is to not include this part of the TRL 5 tests. Also, due to the project's limited amount of time, wouldn't be possible to automate all the tests in time, so the idea is to automate the most tests possible.

4.1.4 Results Evaluation Methodology

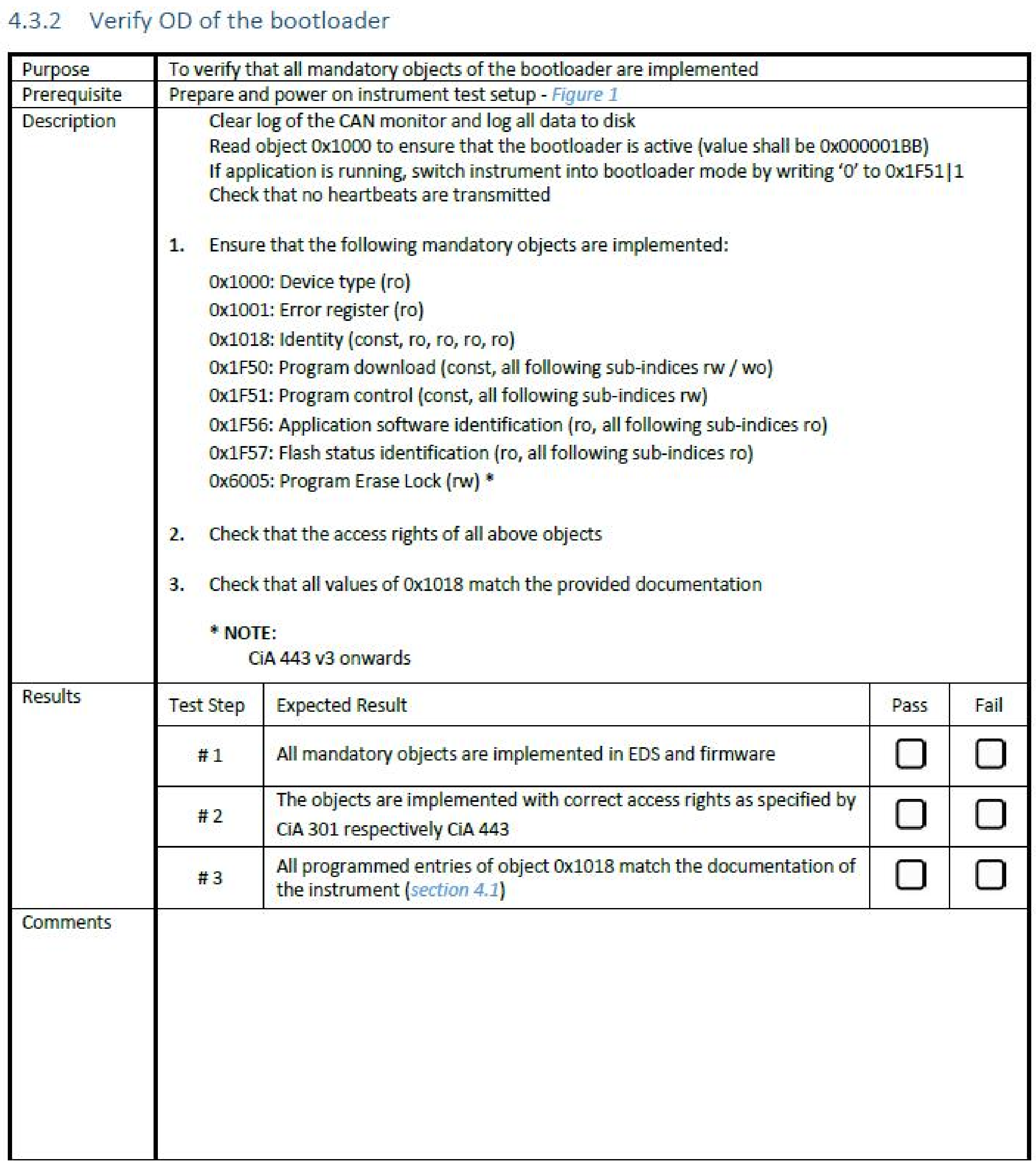
After defining the tests it is necessary to determine how the test results will be evaluated, in other words, how will be defined if the product is ready to be TRL 5 or not.

In relation to the result of the SIIS test list, it is already included the expected results for each test. If the device achieved the expected result, it passes, if the outcome is different than anticipated, it fails the test. An example of a test from the SIIS document can be seen in Figure 32. In regards to the CTT results, the output of this tool is a list of the tests that were a success, a list of the ones that failed, and a justification for why some of them failed.

Given these results, the used methodology was a qualitative method, where all the tests from both SIIS and CTT would be done, but the weight of each result will be defined based on the customer's point of view. This indicates that it is more valuable that the device performs a specific CAN test, which is important for the client than a more common CAN test, which isn't interesting for the customer. In this way, it is possible to verify if the system's interface is ready to perform the most important of the client's operations, which is the principle evaluated in the TRL 5. The evaluation of the importance of each test was elaborated based on the group's knowledge of the subsea scenario and the feedback from the partner company.

Regarding the automatic test results, the evaluation is going to be made by comparing the results from the automatic tests with the manual tests. This means that if the results from

Figure 32 – Example of a test from the SIIS level 2 document.



4.3.2 Verify OD of the bootloader

Purpose	To verify that all mandatory objects of the bootloader are implemented			
Prerequisite	Prepare and power on instrument test setup - *Figure 1*			
Description	Clear log of the CAN monitor and log all data to disk Read object 0x1000 to ensure that the bootloader is active (value shall be 0x000001BB) If application is running, switch instrument into bootloader mode by writing '0' to 0x1F51|1 Check that no heartbeats are transmitted 1. Ensure that the following mandatory objects are implemented: 0x1000: Device type (ro) 0x1001: Error register (ro) 0x1018: Identity (const, ro, ro, ro, ro) 0x1F50: Program download (const, all following sub-indices rw / wo) 0x1F51: Program control (const, all following sub-indices rw) 0x1F56: Application software identification (ro, all following sub-indices ro) 0x1F57: Flash status identification (ro, all following sub-indices ro) 0x6005: Program Erase Lock (rw) * 2. Check that the access rights of all above objects 3. Check that all values of 0x1018 match the provided documentation *** NOTE:** CiA 443 v3 onwards			
Results	Test Step	Expected Result	Pass	Fail
	# 1	All mandatory objects are implemented in EDS and firmware	☐	☐
	# 2	The objects are implemented with correct access rights as specified by CiA 301 respectively CiA 443	☐	☐
	# 3	All programmed entries of object 0x1018 match the documentation of the instrument (*section 4.1*)	☐	☐
Comments				

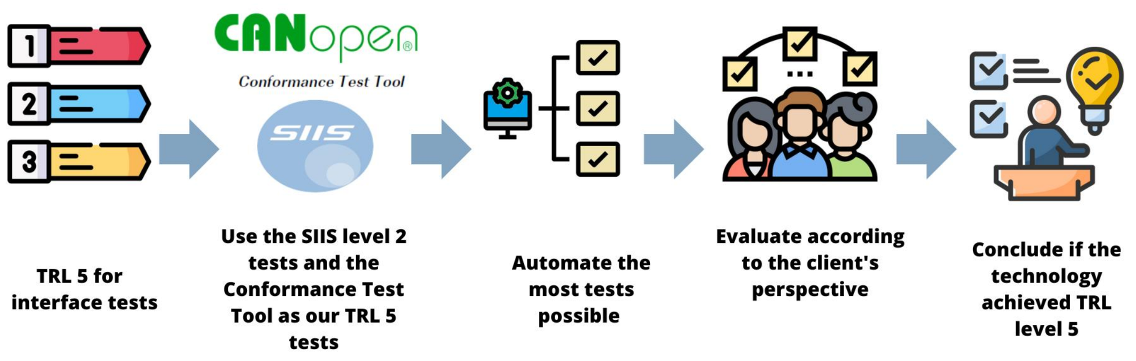
Source: SIIS (2016).

both approaches are the same, it means that the automatic tests are working properly.

4.1.5 Overview of the solution

To summarize, the interpretation of the TRL 5 principles led us to the conclusion that to achieve this level, it is necessary that the technology has the technology's interface ready to be used in a real scenario, in other words, by the customer. We chose to focus only on the software interface, meaning that the tests will be made to verify the CANopen performance. Taking this into consideration we established that the tests to be done are the SIIS Level 2 test list and the CTT tests because they both analyze the CANopen interface. Also, we decided to approach as well the automation of the SIIS tests, so that we can reproduce those tests easily and test other devices as well. Regarding the evaluation of the results, the method to ponder each test took into consideration the customer's point of view and was qualified according to their importance in a real-world scenario. After the evaluation, we concluded if the technology achieved the TRL 5 in the software area. A diagram explaining this overview can be seen in Figure 33.

Figure 33 – Overview of the proposed solution.


CANopen
Conformance Test Tool
SIIS
TRL 5 for interface tests
Use the SIIS level 2 tests and the Conformance Test Tool as our TRL 5 tests
Automate the most tests possible
Evaluate according to the client's perspective
Conclude if the technology achieved TRL level 5

Source: Personal file.

4.2 OTHER METHODOLOGIES APPLIED

In this section, we explain the methodologies used throughout this report's project and also in the PJ-SAS group.

4.2.1 PJ-SAS Mehodology

In relation to the methodology of the entire PJ-SAS group, it uses an agile methodology in which the group makes meetings regularly to verify the advances in the different areas. This procedure encourages the presentation of results and stimulates the members to have new progress every week. Additionally, the agile method is an organic way of making every member of the team aware of what is happening in the project as well as facilitating the interaction between different sectors of the team in case of need. Besides the meetings, the team also has another type of meeting called "demos", which are meetings with the same purpose but with longer presentations to show something more practical from your work.

4.2.2 Automatic Tests Programming Methodology

To develop the automatic tests, it was used a methodology similar to scrums. "Scrum is an agile project management framework that helps teams structure and manages their work through a set of values, principles, and practices" (DRUMOND, 2021). As it was described, scrum is a very complex tool that covers a lot of different areas of a software development process. However, the automatic tests aren't as complex as a software developed by a team. For that reason, just a few concepts of this framework were used.

Among these concepts, the main one is the *sprint* concept, which is a short period in which all the efforts are focused on completing a certain task. Applying this idea to the application of the automatic tests, every test from the SIIS document is a sprint, meaning that the tests were made one out of time and the next one would be only produced after the first one is completely developed and tested.

5 DEVELOPMENT AND ANALYSIS OF THE RESULTS OBTAINED

With the solution already in mind, it was necessary to implement it. In this chapter, we will describe how proposed the solution exposed in Chapter 4 was implemented, the obtained results, and our conclusion on it. In Section 5.1 we explain in detail how the solution was implemented, in Section 5.2 we analyze the results that were obtained and make a critical analysis of it, and in Section 5.3 we conclude if the SVA is indeed at TRL 5 from the software communication point of view.

5.1 SOLUTION IMPLEMENTATION

In this section, we describe how the solution was implemented, presenting the equipment used and the decisions that were made regarding the implementations. Also, we will separate description of the implementation into the manual tests and the automatic tests.

5.1.1 Manual Tests Setup

Since both tests were made with different setups and they are executed differently, they were subdivided into two subsections.

5.1.1.1 SIIS Tests

The SIIS Level 2 tests cover many different areas of the device. It consists of 6 different features:

- **Identification:** the user inserts the information regarding the device's name, the software version, the supported bit rates, etc.
- **Instrument Configuration:** the tests are related to the initial setting of the CAN bus, in other words, this part of the tests verifies how well the LSS service works, which, as explained in the 2.2.6, is responsible for configuring the bus and its devices.

  Summarizing the demands of these tests, it is asked to see if the device can make a power circle correctly (turn off and on the device), it is asked to configure the node ID -which is the identification of a device in a CAN bus- and it asked to change the bit rate of the bus
- **CANopen:** in here all the main CANopen services and functions are tested.

  Summarizing the requests of this section, it is asked to check the transition between NMT states, verify the OD from both boot-loader mode and application mode, it is asked to validate the information inserted in the last section of tests with the EDS file, check the performance of PDOs and SDOs and verify the access to the CANopen functions.

  Remembering what was clarified in previous chapters, Section2.2.4.5 explains that NMT is responsible for managing the operational states of the CAN bus devices. Section2.2.3

reports that the OD is a table that carries the addresses of a device's variables. Section2.2.5 describe that the EDS is a file that contains OD description. Section2.2.4.2 and Section2.2.4.1 explains that PDOs and SDOs are type of messages that are sent between nodes in a CAN bus.

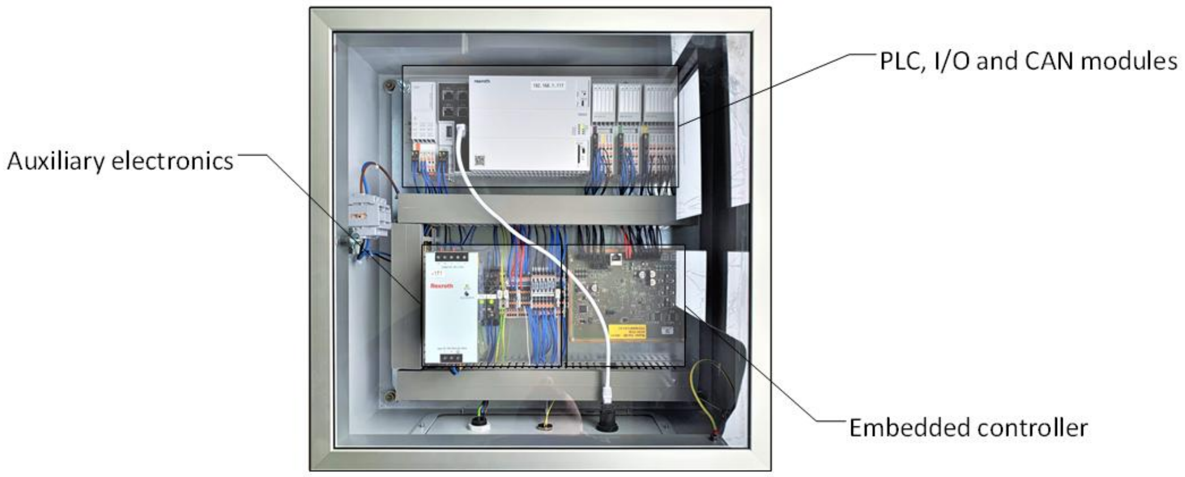
- **Performance Tests:** in here it is tested the performance of some CAN and CANopen features. This includes testing the bootup time, verifying the SDO response time, the accuracy of periodic messages, and the interval between PDO messages.
- **Hardware tests:** in here it is measured the power consumption of the device with different voltage supplies, measure the inrush current in a cold start -meaning that the current of the supply power should be measured when the device is started - and fault tolerance characteristics are tested as well.

  As it was explained in Section2.1.1, fault-tolerant is a type of physical layer from CAN, that is more robust than the normal High-Speed CAN and has some unique features like being able to maintain connection even without one of the bus cables.
- **Optional Tests:** in here it is asked to test the multiplex PDO, which is a type of PDO used when more precise data is transmitted.

With the tests in mind, we entered into contact with one partner company for our project to have some feedback on our approach and also our device. The feedback validated the way that the SVA already communicates, in other words, the manner in which our device communicates is already the same as in the market.

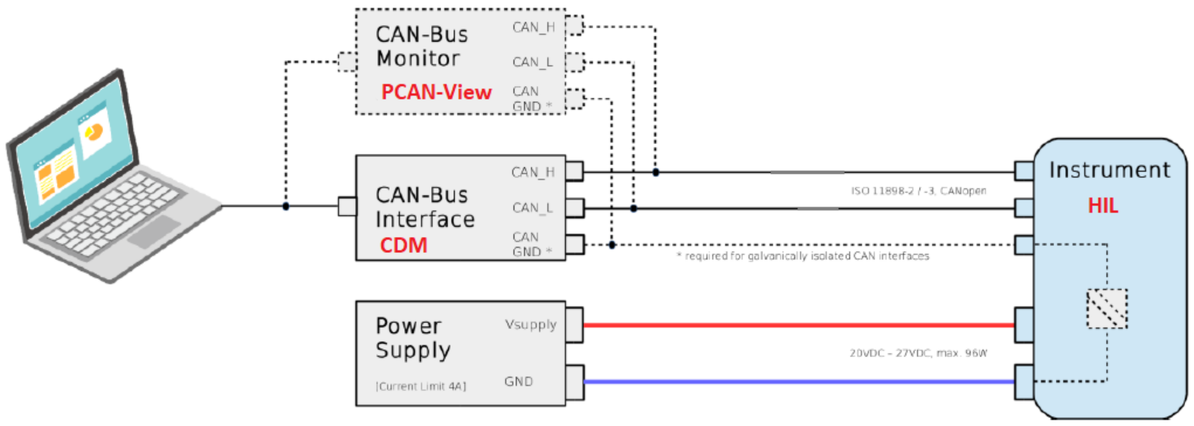
After this, we needed to make a setup that would be able to send CAN messages to the device and monitor the messages on the bus at the same time. For this reason, the setup was made using a computer connected directly to the device. The computer was responsible for sending the messages and another software was able to monitor the messages on the bus. The software that we used to send the messages was CAN Device Monitor (CDM), a software that is able to make use of most CANopen services, which, remembering from Section2.2.4, are unique features of CANopen-enabled devices and buses. To monitor the data sent through the bus, we used the *PCAN-View*, which is a software that is able to show every message sent or received from the computer, detailing the data contained in the message, length, and CAN-ID. In relation to the actual device, we used our HIL, which is a mock-up of the hardware and software of our SVA. Its original objective was to simplify the testing of new actualization in the controller's software - which we call Actuator Control Module (ACM). This signifies that when an actualization was ready to be implemented, the group could implement it into this mock-up instead of the real device without the mechanical aspects of it. Figure 34 shows the physical implementation of the HIL cabinet. But, for this test setup, it served as a perfect imitation of the SVA, due to having the same software and the same PLC embedded into it. Figure 35 shows a diagram of the setup and Figure 36 illustrates the data flow of this setup.

Figure 34 – HIL cabinet.



Source: SIIS (2016).

Figure 35 – Setup 1 for the SIIS tests.



Source: SIIS (2016).

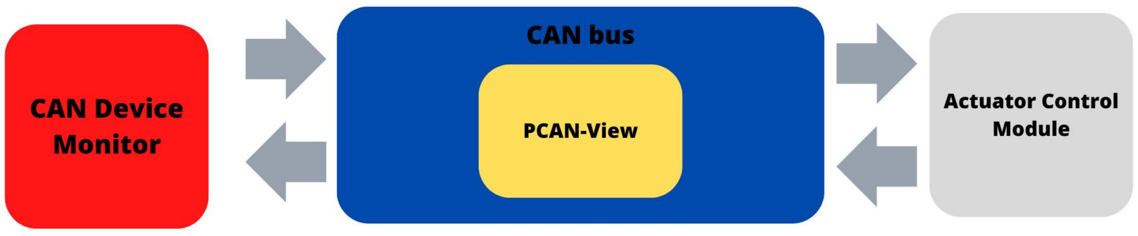
Regarding the hardware on this setup, we used the PEAK-USB-Adapter to connect the computer to the CAN bus and the bus converter PCAN-TJA1054 to convert the High-Speed CAN signals coming from the computer to the HILs Low-Speed CAN. The converter was added because the PEAK-USB-Adapter can only send messages in High-Speed CAN. Figure 37 shows the hardware setup for this part of the tests and Figure 38 shows the data flow of this test setup.

5.1.1.2 CTT Tests

Regarding the CTT tests, they were made using the same hardware setup shown in Figure 37, but instead of using the CDM and the PCAN-View, the only software used was the CTT.

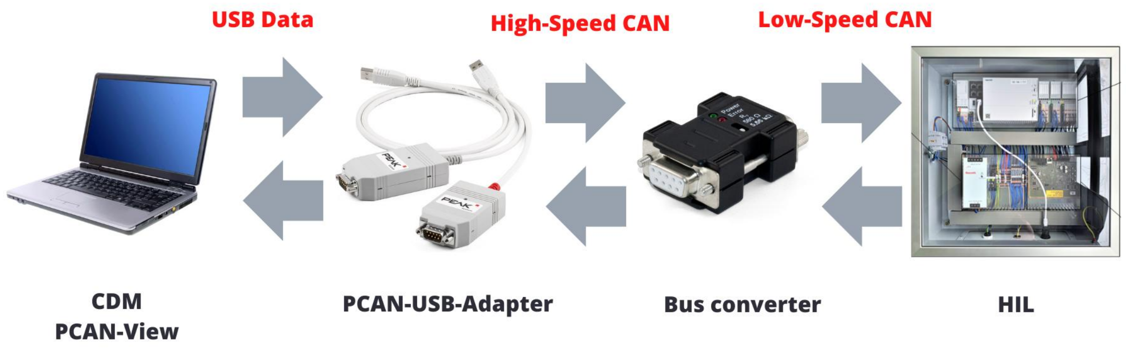
In relation to the tests made, CiA programmed a large number of tests to be made. These

Figure 36 – Data flow of the test setup.



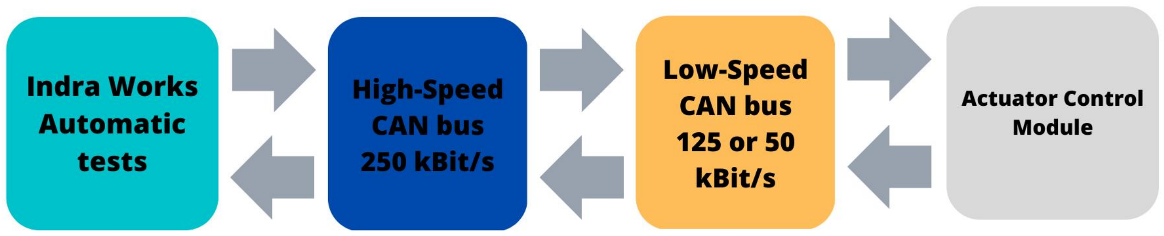
Source: Personal File.

Figure 37 – Hardware setup for the SIIS tests.



Source: Personal file.

Figure 38 – Data flow of the automatic test setup.


Indra Works Automatic tests
High-Speed CAN bus 250 kBit/s
Low-Speed CAN bus 125 or 50 kBit/s
Actuator Control Module

Source: Personal File.

tests are basically messages that are sent to the device and a response is expected, if this answer is achieved, the test is marked as ’success’, otherwise, it is marked as ’failed’.

The tests are divided into 10 sections, which correspond to the main CANopen services: EDS, SDO, PDO, OD, EMCY, SYNC, Error control, Testing other, NMT states and NMT transitions.

5.1.2 Automatic Tests Setup

To create the automatic tests we needed to come up with another setup. Instead of using the CDM and the PCAN-View, we needed to use a device that was able to send and receive CAN messages. To solve this problem we used the XM22 - a PLC designed by Rexroth - attached in a CAN module. This way, we were able to program the XM22 to send the messages. To program the XM22 we used Indra Work, which is an Integrated Development Environment (IDE) for embedded systems, that was created based on another evnironment called Codesys. Even though Indra Works offers a big amount of programming languages, the tests were programmed in Structured Text, a common language for embedded systems.

The problem with that XM CAN module is that it works only with 250 kBit/s and High-Speed CAN. Both are issues because, as it was explained in the Section3.5.2, SIIS says that is mandatory that the device operates in Low-Speed CAN and with the bit rates 50 and 125 kBit/s. To solve the first problem, we were able to use the same bus converter that was used in the last set of tests to transform the High-Speed CAN signals into Low-Speed. But to solve the second, we needed to add another device that would e able to operate in those bit rates and was able to change it automatically. For that reason, we added the Ixxat CAN Bridge 420NT to our setup. This device is a bridge that is able to link two busses with different baud rates and also offers a lot of configuring options. Among these options, there is the possibility of programming the device using LUA, which enabled us to implement a code that changes the bit rate upon receipt of a specific message. With those two devices, we were able to make the setup of the automatic tests, which an overview is shown in Figure 39.

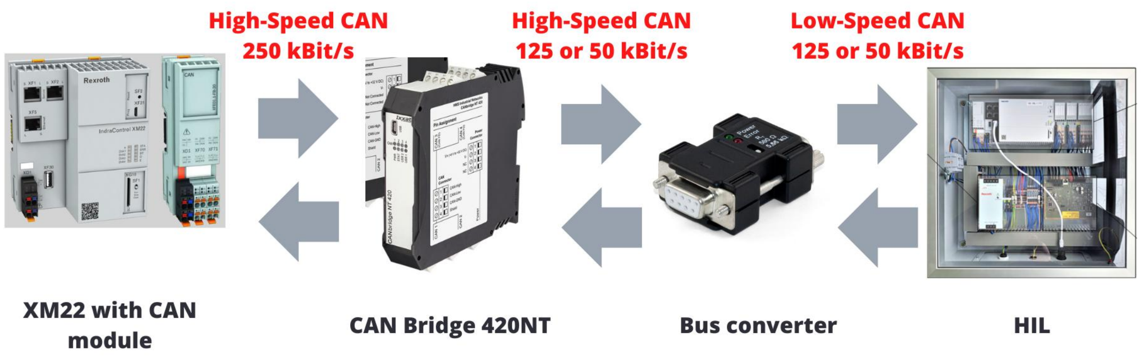
Although the idea was to use the HIL to develop the tests, we started the development using another controller that implements CANopen (SBL-2300), due to theHIL not being available at the time that we began the experiments.

5.2 RESULTS ANALYSIS

In this section, we present the obtained results and problems that occurred during the implementation of the solution.

Regarding a problem that occurred in all the tests, it wasn't possible to use the Low-Speed CAN layer with the HIL. This happened because the software embedded in the HILs controller wasn't configured to send the messages through this output. To get around this problem we used the High-Speed CAN output, which doesn't interfere with the test that doesn't regard the Fault Tolerant features since this problem is just in the physical layer and the information sent would be interpreted in the same way. However, this is a very relevant point when we took into consideration the outcomes that will be described in this section. That's because the implementation of the Low-Speed CAN in the physical layer is one of the obligatory SIIS requirements and every subsea production system uses this type of bus. As a result of this, both hardware setups shown in Figure 37 and Figure 39 needed to remove the bus converter

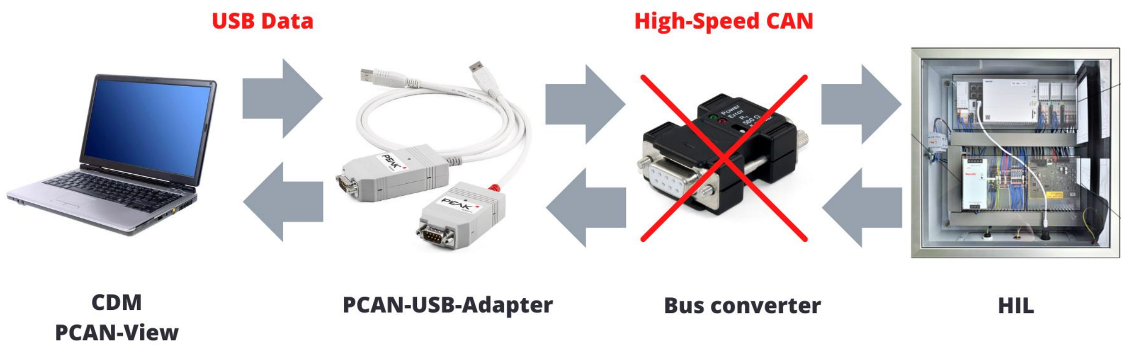
Figure 39 – Hardware setup for the Automatic tests.



Source: Personal file.

PCAN-TJA1054. The new hardware setups can be seen at Figure 40 and Figure 41.

Figure 40 – New hardware setup for the SIIS and CTT tests.



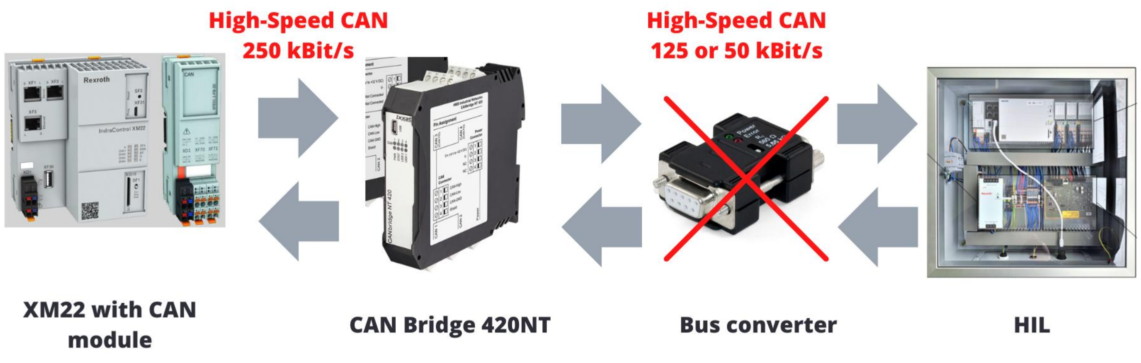
Source: Personal file.

5.2.1 SIIS Results

Here it will be given an overview of the main outcomes according to the test sections.

- **Instrument Configuration:** Test 4.2.2 asks to change the device's node ID. When this test was done we were able to configure it in the bootloader mode but not in the application

Figure 41 – Hardware setup for the Automatic tests.



Source: Personal file.

mode. This is a major problem since, in a real scenario, the bootloader mode is used only shortly after the device is started just to make the setup for the application mode, which means that it is important for the node ID to be changed in the application mode as well.

A similar problem occurred during Test 4.2.3, which requires configuring the device's bit rate. We were able to change it during the bootloader mode, but not during application mode. This is a big problem as well for the same reasons described in the last paragraph. Also, this test asked to change the bit rate to many other values - from 10 kBit/s to 1000 kBit/s - and it wasn't possible, because the HIL could only operate with 50 kBit/s and 125 kBit/s. Given that these are the mandatory bit rates and the others are optional, the result isn't very significant for the evaluation.

- **CANopen:** Test 4.3.1 evaluated the NMT states performance, sending NMT messages like "Start", "Stop", and "Reset" and verifying if the device went to the right state (NMT state machine can be seen in Figure 14). During this test, the ACM got 75% of the transitions right. But, the remaining 25% of the tests involved sending a "Reset" message to the entire network. This indicates that when a message is sent to all the equipment asking them to reset, our device doesn't work properly, which is a very considerable mistake.

  Test 4.3.2 and 4.3.3 assesses the implementation and access rights of the OD in both bootloader and application mode. In other words, it is verified if all the parameters that were supposed to be applied are in fact added and if they have the correct access right. In this test, some of the variables didn't have the correct access right but all the parameters were implemented correctly. This outcome isn't so problematic due to the fact that all the parameters are applied and having the wrong access right in some of them wouldn't compromise the device's operation.

  Test 4.3.4 check the PDO transmission, asking the user to set up the sending of PDO messages with different transmission types. In this test, we were able to send the PDO with all the transmission types, but we weren't able to change the mapping of the variables that would be transmitted by the PDO. This means that it is not possible to select which parameters will be sent in the message. This outcome is a very crucial issue, because, in a real-world scenario, the user may select the variables that he wants to read.

Test 4.3.5 verifies the PDO inhibit time - which is the minimum time between two messages. In this test, it wasn't possible to configure the inhibit time. This result is not very significant because it is possible to operate the device without a inhibit time.

Tests 4.3.6 through 4.3.10 check firmware updates. Basically, these tests ask to verify different aspects of the firmware update (upload a corrupt file, lock the application, upload a valid file). Through these tests, we noticed that it wasn't possible to change from the application mode to bootloader, only the other way around. Also, we had a problem after executing these tests. First, it wasn't possible anymore to change to application mode and second, we needed to reinstall the software into our controller. Another problem was that we weren't able to download a valid file into the device. In regard to the relevancy of these issues, they were very significant, because, since the device will be placed in Deep or Ultra-Deepwater under the sea, it is fundamental that we are able to update the software via these methods and we realized that the device is not yet ready for it. Besides this, the problem where it wasn't possible to change back to the application mode is a very serious problem, because it is critical that the gadget is able to go to this mode.

- **Performance Tests:** Tests 4.4.1 through 4.4.5 ask for performance details - the time between messages, the accuracy of periodic messages, etc. Since they are only tests that measure the times and delays in some situations, they weren't so relevant for a real situation.

- **Hardware tests:** Test 4.5.1 and 4.5.2 inquire about measurements and testing with different voltages and currents in the power supply. The issue was that it wouldn't make sense to make these tests with the HIL, for the reason that the HIL and the actual SVA use different components (sensors and actuators). This indicates that for this type of test, it would be necessary to use the real SVA so that we could certify that all the pieces of equipment would work. However, the actual SVA was about to be shown in a fair, so it wasn't available to be used at the time.

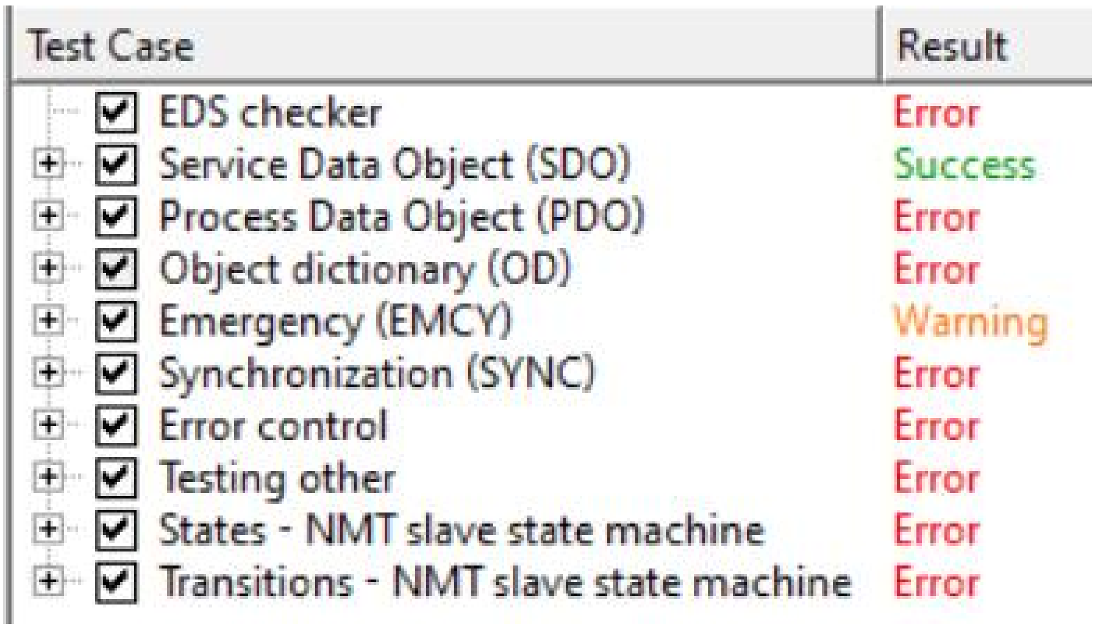
  Test 4.5.3 asked to verify the fault tolerance features of the bus. However, due to the problem stated at the beginning of this section, it wasn't possible to use a Fault-Tolerant CAN bus, so this test wasn't executed.

- **Optional Tests:** Test 4.6.1 - the only test of this section - wasn't executed. That is because the test asked to verify the operation of multiplex PDO, but the device doesn't implement them.

5.2.2 CTT Results

After executing the CTT, the software displays a log with the messages sent in the tests, the outcome and, in case of failure, the reason why it failed. The general results will be discussed according to the test sections. Figure 42 shows the overview of the test results divided into sections. Each section has a number of tests and the section's output will be the result of the Boolean operation *and* applied in every test, meaning that if there is one failed test, the section's outcome will be *error* and if all the tests pass then the outcome will be *success*. If a test is skipped, it means that this test isn't applied in this device and isn't considered in the operation.

Figure 42 – Overview of CTT results.



Test Case	Result
☑ EDS checker	Error
☑ Service Data Object (SDO)	Success
☑ Process Data Object (PDO)	Error
☑ Object dictionary (OD)	Error
☑ Emergency (EMCY)	Warning
☑ Synchronization (SYNC)	Error
☑ Error control	Error
☑ Testing other	Error
☑ States - NMT slave state machine	Error
☑ Transitions - NMT slave state machine	Error

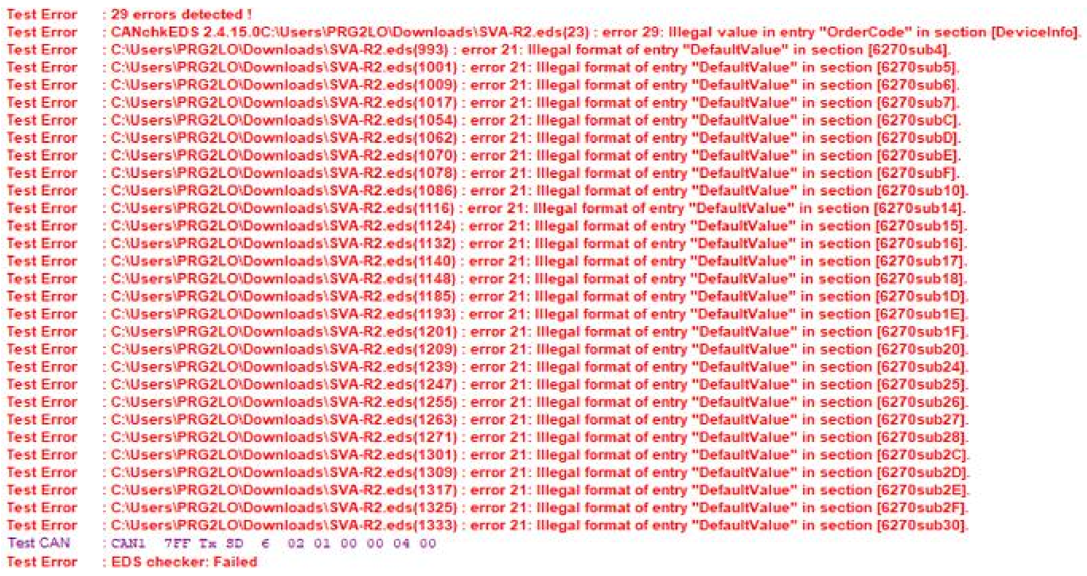
Source: Personal file.

- **EDS checker:** this section is a single test where the EDS file is checked to see if it is on the CiA standards. We were able to see that there were some problems with the declaration of some objects as shown in Figure 43. We weren't able to understand the reason behind these problems, but it had something to do with the declaration of the initial value of these parameters. This isn't a major issue because we were able to use the variables even with this problem.

- **SDO:** among the 30 tests, 16 succeeded and 14 were skipped. This outcome was a great achievement since SDOs are a critical part of every device's operation and this shows that there is no issue with the general operation of the SDOs.

- **PDO:** among 33 tests, 3 succeeded, 15 failed and 15 were skipped. The reason behind the failures was that it wasn't possible to read the COB-ID of the PDOs. Figure 44 shows

Figure 43 – EDS checker error log.

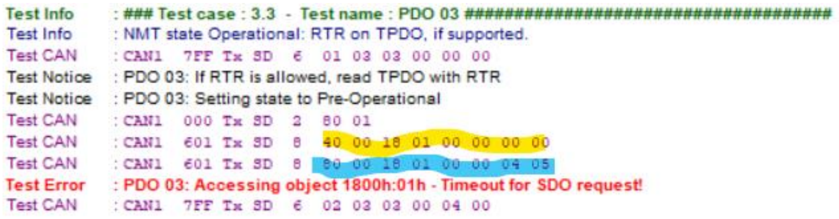


```
Test Error    : 29 errors detected !
Test Error    : CANchkEDS 2.4.15.0C:\Users\PRG2LO\Downloads\SVA-R2.eds(23) : error 29: Illegal value in entry "OrderCode" in section [DeviceInfo].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(993) : error 21: Illegal format of entry "DefaultValue" in section [6270sub4].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1001) : error 21: Illegal format of entry "DefaultValue" in section [6270sub5].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1009) : error 21: Illegal format of entry "DefaultValue" in section [6270sub6].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1017) : error 21: Illegal format of entry "DefaultValue" in section [6270sub7].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1054) : error 21: Illegal format of entry "DefaultValue" in section [6270subC].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1062) : error 21: Illegal format of entry "DefaultValue" in section [6270subD].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1070) : error 21: Illegal format of entry "DefaultValue" in section [6270subE].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1078) : error 21: Illegal format of entry "DefaultValue" in section [6270subF].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1086) : error 21: Illegal format of entry "DefaultValue" in section [6270sub10].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1116) : error 21: Illegal format of entry "DefaultValue" in section [6270sub14].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1124) : error 21: Illegal format of entry "DefaultValue" in section [6270sub15].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1132) : error 21: Illegal format of entry "DefaultValue" in section [6270sub16].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1140) : error 21: Illegal format of entry "DefaultValue" in section [6270sub17].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1148) : error 21: Illegal format of entry "DefaultValue" in section [6270sub18].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1185) : error 21: Illegal format of entry "DefaultValue" in section [6270sub1D].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1193) : error 21: Illegal format of entry "DefaultValue" in section [6270sub1E].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1201) : error 21: Illegal format of entry "DefaultValue" in section [6270sub1F].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1209) : error 21: Illegal format of entry "DefaultValue" in section [6270sub20].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1239) : error 21: Illegal format of entry "DefaultValue" in section [6270sub24].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1247) : error 21: Illegal format of entry "DefaultValue" in section [6270sub25].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1255) : error 21: Illegal format of entry "DefaultValue" in section [6270sub26].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1263) : error 21: Illegal format of entry "DefaultValue" in section [6270sub27].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1271) : error 21: Illegal format of entry "DefaultValue" in section [6270sub28].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1301) : error 21: Illegal format of entry "DefaultValue" in section [6270sub2C].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1309) : error 21: Illegal format of entry "DefaultValue" in section [6270sub2D].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1317) : error 21: Illegal format of entry "DefaultValue" in section [6270sub2E].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1325) : error 21: Illegal format of entry "DefaultValue" in section [6270sub2F].
Test Error    : C:\Users\PRG2LO\Downloads\SVA-R2.eds(1333) : error 21: Illegal format of entry "DefaultValue" in section [6270sub30].
Test CAN      : CAN1   7FF  Tx  SD   6   02  01  00  00  04  00
Test Error    : EDS checker: Failed
```

Source: Personal file.

the message log from one of the PDO tests attempting to read the COB-ID from the PDO. Although the problem that made the tests failed was simple, it shows that the PDOs from the device are not operational yet.

Figure 44 – PDO error log. In yellow is a *read* request and in blue is the *error* message.



```
Test Info     : ### Test case : 3.3  -  Test name : PDO 03 ######################################
Test Info     : NMT state Operational: RTR on TPDO, if supported.
Test CAN      : CAN1   7FF  Tx  SD   6   01  03  03  00  00  00
Test Notice   : PDO 03: If RTR is allowed, read TPDO with RTR
Test Notice   : PDO 03: Setting state to Pre-Operational
Test CAN      : CAN1   000  Tx  SD   2   80  01
Test CAN      : CAN1   601  Tx  SD   8   40  00  18  01  00  00  00  00
Test CAN      : CAN1   601  Tx  SD   8   80  00  18  01  00  00  04  05
Test Error    : PDO 03: Accessing object 1800h:01h - Timeout for SDO request!
Test CAN      : CAN1   7FF  Tx  SD   6   02  03  03  00  04  00
```

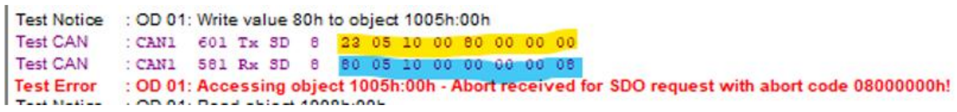
Source: Personal file.

- **OD:** among 9 tests, 6 succeeded, one failed and 2 were skipped. The only issue that came up in this section was that a variable could not be accessed, as shown in Figure 45. This outcome is very good considering that the device passed in most of the tests and the OD is a very important part of any CANopen device.

- **EMCY:** among 6 tests, 3 succeeded, 2 were skipped and 1 gave a warning. It wasn't clear why the warning happened but the general result was very satisfactory, due to the fact that the EMCY is an important feature so that any problems are informed.

- **SYNC:** among 10 tests, 1 succeeded, 7 failed and 2 were skipped. The problem in this section came from the same place as the OD test, which was a problem in accessing

Figure 45 – OD error log. In yellow is a *write* message and in blue is the *error* message.

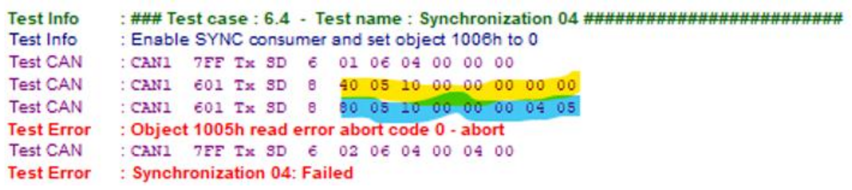


```
Test Notice   : OD 01: Write value 80h to object 1005h:00h
Test CAN      : CAN1   601 Tx SD   8   23 05 10 00 80 00 00 00
Test CAN      : CAN1   581 Rx SD   8   80 05 10 00 00 00 00 08
Test Error    : OD 01: Accessing object 1005h:00h - Abort received for SDO request with abort code 08000000h!
```

Source: Personal file.

one variable. This is very critical because even though the errors came from only one parameter, it shows that the device is not ready to operate with the SYNC service and this is a crucial part of a CANopen device.

Figure 46 – SYNC error log. In yellow is a *read* request and in blue is the *error* message.

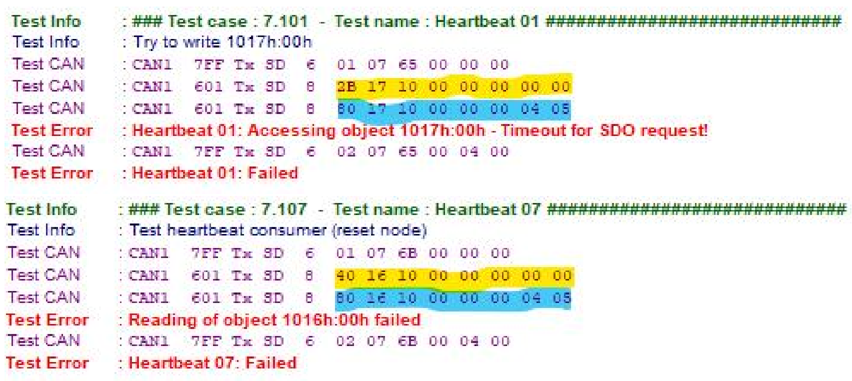


```
Test Info     : ### Test case : 6.4  -  Test name : Synchronization 04 ##########################
Test Info     : Enable SYNC consumer and set object 1006h to 0
Test CAN      : CAN1   7FF Tx SD   6   01 06 04 00 00 00
Test CAN      : CAN1   601 Tx SD   8   40 05 10 00 00 00 00 00
Test CAN      : CAN1   601 Tx SD   8   80 05 10 00 00 00 04 05
Test Error    : Object 1005h read error abort code 0 - abort
Test CAN      : CAN1   7FF Tx SD   6   02 06 04 00 04 00
Test Error    : Synchronization 04: Failed
```

Source: Personal file.

- **Error control:** among 20 tests, 7 failed and 13 were skipped. These tests failed because it wasn’t possible to access 2 variables related to the heartbeat. Although all the applicable tests didn’t work, we believe that isn’t so critical, because the general operation of the heartbeat works, and it isn’t necessary to implement all of its functions.

Figure 47 – Error control error log. In yellow is a *read* or *write* request and in blue is the *error* message.



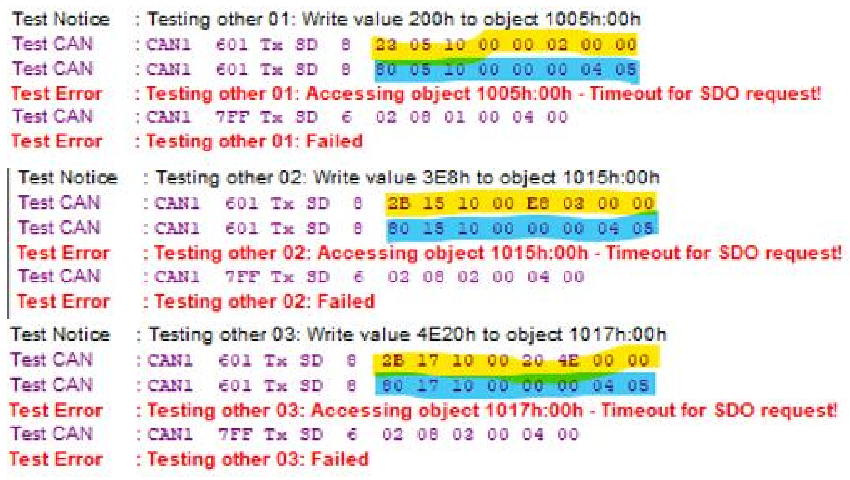
```
Test Info     : ### Test case : 7.101  -  Test name : Heartbeat 01 ##############################
Test Info     : Try to write 1017h:00h
Test CAN      : CAN1   7FF Tx SD   6   01 07 65 00 00 00
Test CAN      : CAN1   601 Tx SD   8   2B 17 10 00 00 00 00 00
Test CAN      : CAN1   601 Tx SD   8   80 17 10 00 00 00 04 05
Test Error    : Heartbeat 01: Accessing object 1017h:00h - Timeout for SDO request!
Test CAN      : CAN1   7FF Tx SD   6   02 07 65 00 04 00
Test Error    : Heartbeat 01: Failed

Test Info     : ### Test case : 7.107  -  Test name : Heartbeat 07 ##############################
Test Info     : Test heartbeat consumer (reset node)
Test CAN      : CAN1   7FF Tx SD   6   01 07 6B 00 00 00
Test CAN      : CAN1   601 Tx SD   8   40 16 10 00 00 00 00 00
Test CAN      : CAN1   601 Tx SD   8   80 16 10 00 00 00 04 05
Test Error    : Reading of object 1016h:00h failed
Test CAN      : CAN1   7FF Tx SD   6   02 07 6B 00 04 00
Test Error    : Heartbeat 07: Failed
```

Source: Personal file.

- **Testing other:** among 10 tests, 5 failed and 4 were skipped. The issues with this test concerned access to variables that were problematic in other sections, as shown in Figure

49. These tests aren't so fundamental for the device's operation, because they verify if it is possible to write in some parameters from the OD, but they highlight that there are some problematic parameters that need to be taken care of.

Figure 48 – Testing other error log. In yellow is a *read* or *write* request and in blue is the error message.

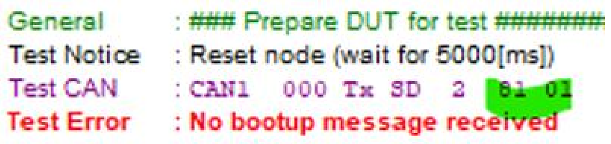


```
Test Notice    : Testing other 01: Write value 200h to object 1005h:00h
Test CAN       : CAN1   601 Tx SD   8   23 05 10 00 00 02 00 00
Test CAN       : CAN1   601 Tx SD   8   80 05 10 00 00 00 04 05
Test Error     : Testing other 01: Accessing object 1005h:00h - Timeout for SDO request!
Test CAN       : CAN1   7FF Tx SD   6   02 08 01 00 04 00
Test Error     : Testing other 01: Failed
Test Notice    : Testing other 02: Write value 3E8h to object 1015h:00h
Test CAN       : CAN1   601 Tx SD   8   2B 15 10 00 E8 03 00 00
Test CAN       : CAN1   601 Tx SD   8   80 15 10 00 00 00 04 05
Test Error     : Testing other 02: Accessing object 1015h:00h - Timeout for SDO request!
Test CAN       : CAN1   7FF Tx SD   6   02 08 02 00 04 00
Test Error     : Testing other 02: Failed
Test Notice    : Testing other 03: Write value 4E20h to object 1017h:00h
Test CAN       : CAN1   601 Tx SD   8   2B 17 10 00 20 4E 00 00
Test CAN       : CAN1   601 Tx SD   8   80 17 10 00 00 00 04 05
Test Error     : Testing other 03: Accessing object 1017h:00h - Timeout for SDO request!
Test CAN       : CAN1   7FF Tx SD   6   02 08 03 00 04 00
Test Error     : Testing other 03: Failed
```

Source: Personal file.

- **States - NMT:** Among 4 tests, 4 failed. The issue with this one was that the software didn't receive any bootup message after receiving a "Reset" message, which is the standard procedure of a device that implements the NMT state machine. This failure shows that the device wasn't able to implement the NMT state machine completely and this is a very important feature of all CANopen devices.

Figure 49 – NMT error log. In green is the 'Reset node' message.



```
General        : #### Prepare DUT for test ########
Test Notice    : Reset node (wait for 5000[ms])
Test CAN       : CAN1   000 Tx SD   2   81 01
Test Error     : No bootup message received
```

Source: Personal file.

- **Transitions - NMT:** Among 4 tests, 4 failed. The problem with this one was the same problem that happened in the last section, meaning that the device didn't send a bootup message after being reset. Also, the conclusion is that the device is not able to operate the NMT state machine and this is fundamental for every CANopen device.

After executing the tests, it was clear that there are many problems that came up for the same reasons - not being able to access one variable for example. This is a good outcome because it indicates that with just a few adjustments, many of the issues can be resolved. On the other side, currently, the ACM has many CANopen features that are not working properly and this is very critical for the general operation.

5.2.3 Automatic Tests Results

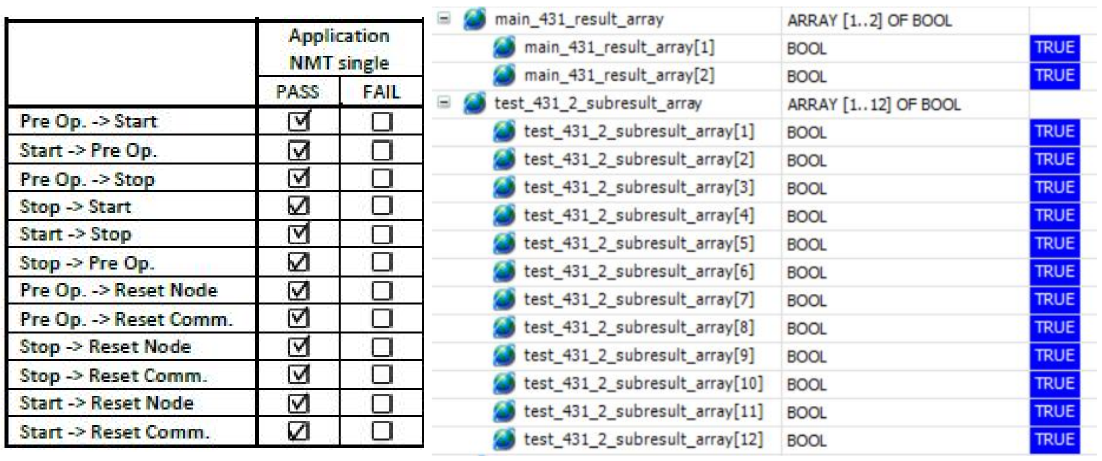
Unfortunately, we weren't able to implement all the tests on the SIIS document, due to the time available for this project. Among the 21 tests, we were able to automate 4 of them. the tests were: 4.3.1 - *Verify NMT Control*, 4.3.2 - *Verify OD of the bootloader*, 4.3.3 *Verify OD of the application* and a function from the 4.2.3 *Configure bit rate*.

Like it was explained in Section4.2.2, the software was developed one test at a time, but due to the fact that the HIL wasn't available initially, we made the software using the SBL-2300 controller and later we applied those tests into the HIL.

As for the evaluation of the results, as it was explained in Section 4.1.4, they were analyzed by comparing the outcome from the SIIS with the Indra Works results.

1. **4.3.1 - Verify NMT Control:** this test worked perfectly, the output from the Indra Works code performed exactly like the manual tests as shown in Figure 50. On the image, it is possible to see also the array *main_431_result_array* which represents the results of the main tests that were made inside the 4.3.1 test and the *test_431_2_subresult_array* are the results of the NMT transitions that are shown on the left. This was a great achievement because it showed that it is possible to manipulate the NMT states with an automatic approach.

Figure 50 – Comparison of SIIS tests result (left side) with the automatic tests result (right side).



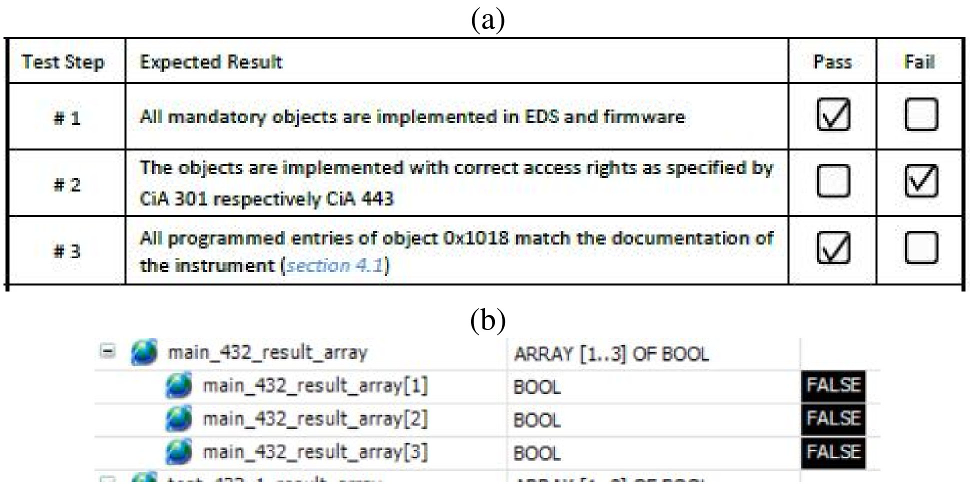
	Application NMT single	
	PASS	FAIL
Pre Op. -> Start	☑	☐
Start -> Pre Op.	☑	☐
Pre Op. -> Stop	☑	☐
Stop -> Start	☑	☐
Start -> Stop	☑	☐
Stop -> Pre Op.	☑	☐
Pre Op. -> Reset Node	☑	☐
Pre Op. -> Reset Comm.	☑	☐
Stop -> Reset Node	☑	☐
Stop -> Reset Comm.	☑	☐
Start -> Reset Node	☑	☐
Start -> Reset Comm.	☑	☐

main_431_result_array	ARRAY [1..2] OF BOOL	
main_431_result_array[1]	BOOL	TRUE
main_431_result_array[2]	BOOL	TRUE
test_431_2_subresult_array	ARRAY [1..12] OF BOOL	
test_431_2_subresult_array[1]	BOOL	TRUE
test_431_2_subresult_array[2]	BOOL	TRUE
test_431_2_subresult_array[3]	BOOL	TRUE
test_431_2_subresult_array[4]	BOOL	TRUE
test_431_2_subresult_array[5]	BOOL	TRUE
test_431_2_subresult_array[6]	BOOL	TRUE
test_431_2_subresult_array[7]	BOOL	TRUE
test_431_2_subresult_array[8]	BOOL	TRUE
test_431_2_subresult_array[9]	BOOL	TRUE
test_431_2_subresult_array[10]	BOOL	TRUE
test_431_2_subresult_array[11]	BOOL	TRUE
test_431_2_subresult_array[12]	BOOL	TRUE

Source: Personal file.

2. **4.3.2 - Verify OD of the bootloader:** with this test, apparently the automatic test got 1 out of 3 right, as shown in Figure 51, but looking more in detail the results, it is possible to have a more precise conclusion.

Figure 51 – Comparison of SIIS tests results (a) with the automatic tests results (b).


(a)

Test Step	Expected Result	Pass	Fail
# 1	All mandatory objects are implemented in EDS and firmware	☑	☐
# 2	The objects are implemented with correct access rights as specified by CiA 301 respectively CiA 443	☐	☑
# 3	All programmed entries of object 0x1018 match the documentation of the instrument (section 4.1)	☑	☐

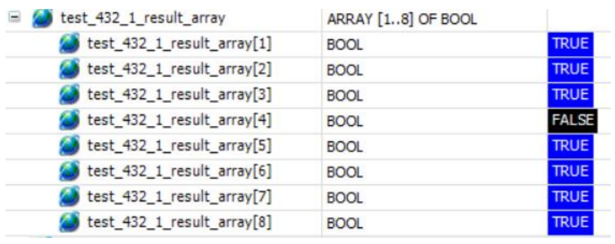
(b)

main_432_result_array	ARRAY [1..3] OF BOOL	
main_432_result_array[1]	BOOL	FALSE
main_432_result_array[2]	BOOL	FALSE
main_432_result_array[3]	BOOL	FALSE

Source: Personal file.

Test #1 asks to check if the device implements the objects listed in the test. The automatic test was able to verify the existence of almost all the parameters except one (0x1F50), as shown in Figure 52. To clarify, the correct result should be all the variables equal to *TRUE*. But after investigating this parameter, we saw that it gives an error message when it is read and that is why the test failed. From that perspective, it was a very good outcome because it accomplished what it was intended to do.

Figure 52 – Results of 4.3.1 #1.

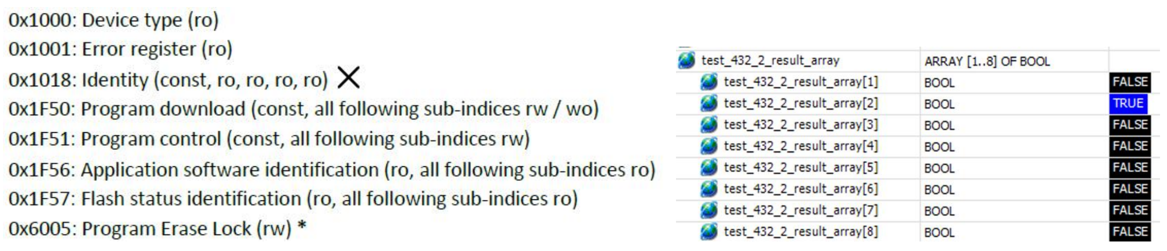


test_432_1_result_array	ARRAY [1..8] OF BOOL	
test_432_1_result_array[1]	BOOL	TRUE
test_432_1_result_array[2]	BOOL	TRUE
test_432_1_result_array[3]	BOOL	TRUE
test_432_1_result_array[4]	BOOL	FALSE
test_432_1_result_array[5]	BOOL	TRUE
test_432_1_result_array[6]	BOOL	TRUE
test_432_1_result_array[7]	BOOL	TRUE
test_432_1_result_array[8]	BOOL	TRUE

Source: Personal file.

Test #2 asked to verify if the implemented objects have the right access right. The automatic test evaluated this by trying to read and write the variable and checking the errors that came up if there were any. Unfortunately, the errors that came up when we tried to interact with the parameters were different than expected. This happened because the SBL-2300 didn't use the same error standard as the HIL and because the OD from the HIL requires different sizes of messages for every parameter and the software wasn't ready to do this. Consequently, it was possible to correctly check access to only one parameter, as shown in Figure 53.

Figure 53 – Comparison of test 4.3.1 #2 from SIIS (left side) and automatic tests (right side).



0x1000: Device type (ro)
0x1001: Error register (ro)
0x1018: Identity (const, ro, ro, ro, ro) ✕
0x1F50: Program download (const, all following sub-indices rw / wo)
0x1F51: Program control (const, all following sub-indices rw)
0x1F56: Application software identification (ro, all following sub-indices ro)
0x1F57: Flash status identification (ro, all following sub-indices ro)
0x6005: Program Erase Lock (rw) *

test_432_2_result_array	ARRAY [1..8] OF BOOL	
test_432_2_result_array[1]	BOOL	FALSE
test_432_2_result_array[2]	BOOL	TRUE
test_432_2_result_array[3]	BOOL	FALSE
test_432_2_result_array[4]	BOOL	FALSE
test_432_2_result_array[5]	BOOL	FALSE
test_432_2_result_array[6]	BOOL	FALSE
test_432_2_result_array[7]	BOOL	FALSE
test_432_2_result_array[8]	BOOL	FALSE

Source: Personal file.

Even though the general output wasn't as good as expected, the results were also satisfactory, given that the main issue with this test can be easily fixable with some adjustments.

Test #3 wasn't possible to be implemented, because, since the test requires that the information read in the objects are compared with the information inserted manually at the beginning of the document, it required a human interface to be built, and we didn't have time to do it.

3. **4.3.3 Verify OD of the application:** this test had a very similar outcome to the 4.3.2, since they both verify the objects in the OD but in different modes. Also, among the 6 sub-tests inside Test 4.3.3, we were only able to implement the first 3 ones.

   Test #1 asked to verify if the values of bits 31 and 30 are equal to zero and one respectively and, in the case that they are, the object 0x6000 should be implemented. Both automatic and manual tests failed because the object in question is implemented and both bits 31 and 30 are zero.

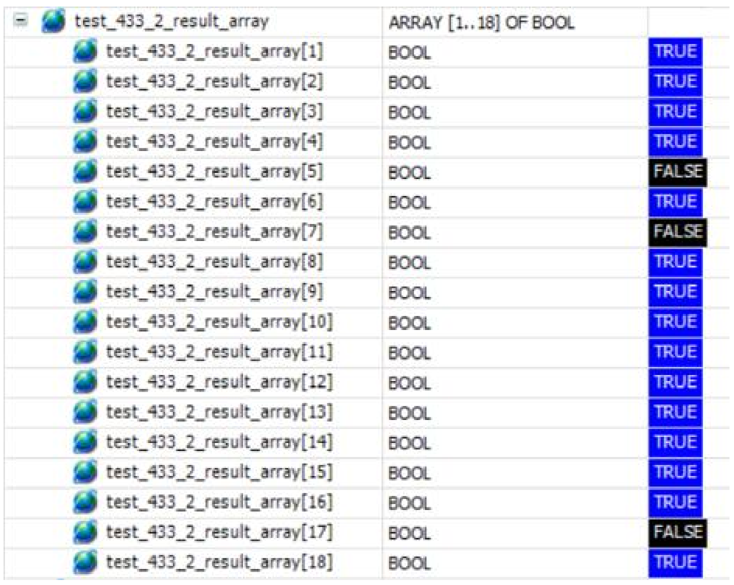
   Test #2 asked to verify if the objects are implemented and, just like Test 4.3.2 #1, there were some inconsistencies due to particular features from certain objects, as can be seen in Figure 54. To clarify, the correct result should be also all the variables equal to *TRUE*.

   Test #3 also asked to check the access type of the parameters, however, just like Test 4.3.2 #2 it wasn't possible to execute the tests properly.

   Since this test is so similar to Test 4.3.2, the conclusions were the same, the outcome of this test is very good although there were many mismatches between the manual test and the automatic one.

4. **Changing bit rate function:** This is not a test, it is a part of Test 4.2.3 - *Configure bit rate*. The idea is to make a function that could change the CAN bus bit rate automatically.

Figure 54 – Results of 4.3.3 #2.



test_433_2_result_array	ARRAY [1..18] OF BOOL	
test_433_2_result_array[1]	BOOL	TRUE
test_433_2_result_array[2]	BOOL	TRUE
test_433_2_result_array[3]	BOOL	TRUE
test_433_2_result_array[4]	BOOL	TRUE
test_433_2_result_array[5]	BOOL	FALSE
test_433_2_result_array[6]	BOOL	TRUE
test_433_2_result_array[7]	BOOL	FALSE
test_433_2_result_array[8]	BOOL	TRUE
test_433_2_result_array[9]	BOOL	TRUE
test_433_2_result_array[10]	BOOL	TRUE
test_433_2_result_array[11]	BOOL	TRUE
test_433_2_result_array[12]	BOOL	TRUE
test_433_2_result_array[13]	BOOL	TRUE
test_433_2_result_array[14]	BOOL	TRUE
test_433_2_result_array[15]	BOOL	TRUE
test_433_2_result_array[16]	BOOL	TRUE
test_433_2_result_array[17]	BOOL	FALSE
test_433_2_result_array[18]	BOOL	TRUE

Source: Personal file.

This was done using the CAN Bridge 420NT to change the baud rate, as explained in Section5.1.2. A code in LUA has been created and embedded into the CAN bridge so that a port bit rate will change when a certain message is sent.

So the operation of this function was to send the LSS messages to change the device bit rate and then send a message to the CAN Bridge so that the bit rate from the device port is changed.

The outcome of this function was a success, the bit rate of the device was set without flaws and the CAN Bridge bit rate changed perfectly when the message was sent.

Given the results exhibited in this chapter, we concluded that the automation of the SIIS test was successful, although there were many problems in the final results, we believe that with enough time and some iterations, the automation has the potential to facilitate the testing process of the SVA and any other device that implements CANopen.

5.3 EVALUATION OF TRL 5 FOR THE SVA

Based on the results described in Section5.2.1, 5.2.2 and the CAN bus problem described in Section 5.2, we concluded that the SVA didn't achieve the TRL 5 in the software aspect.

We have decided that firstly due to the device not being able to operate with Fault-Tolerant CAN. This is a major flaw because this was one of the SIIS requirements and this means that all the subsea should operate with it.

Also, Tests 4.2.2, 4.3.1, 4.3.4, and 4.3.6 through 4.3.10 explored many critical situations in real scenarios, and the device didn't have a good performance in those situations. With an emphasis on not being able to map properly the variables sent through the PDOs, not being able to update the firmware, and not being able to make the node ID and bit rate configurations while in application mode.

Besides the problem with the Fault Tolerant CAN and the issues with SIIS tests, the CTT results influenced this decision as well. The CTT test showed flaws that would seem not so critical - like not being able to access one of the parameters for example - but they interfered in fundamental CANopen services like the SYNC and EMCY. In addition, these tests reinforced flaws that were already detected in the SIIS tests like the PDO problems.

Due to those reasons, we concluded that the communication interface of the SVA would not be ready to be implemented in its intended scenario. However, this project was an excellent addition to the PJ-SAS, because with the issues identified and the procedure methodology defined, the problems can be easily fixed and the tests can be repeated iteratively so that new problems are identified and fixed until the interface is ready for a real scenario. Another reason why this project was a great inclusion to the group was that these issues were not yet identified by the team. At last, after these issues are fixed and a few iterations of the tests are run, it is expected that the SVA is able to attain TRL 5 in regards to the software communication.

6 CONCLUDING REMARKS AND FUTURE PERSPECTIVES

The present work had the objective of evaluating if the new subsea actuator (SVA) under development by Bosch Rexroth was ready to attain the TRL 5 from the software point of view. This was obtained by creating a test procedure that considers the TRL 5 requirements and we created a methodology to make the decision if TRL 5 was indeed achieved by the current state of development of the SVA or not.

First, it was necessary to understand what is the TRL and what are the requirements to achieve level 5. Through this investigation, we concluded that the main demand of this level is that the device is ready to be implemented in the intended system, in other words, the interfaces of the technology should be prepared for a real-world scenario. For this work, we chose to approach only the electrical and software interface of the SVA.

To do so, we had to create tests to assess whether our device was ready to communicate with the customer's system. Since communication in subsea devices is mandatorily made using the CAN and the CANopen protocol, due to the standards defined by the SIIS group, we needed to come up with a set of experiments that would test those features. Therefore, we used the tests from the same SIIS group for CANopen devices. These assessments estimate if a device implements all the main features stated by the CANopen protocol like PDO, SDO, LSS, etc.

For that reason, we made a setup so that we could use the CANopen services and monitor the messages on the bus at the same time. The setup is composed of a computer running two software, the CAN Device Monitor, which acted as the master sending commands and requests to the device, and the PCAN-View, which monitored the messages on the bus. This computer was connected to the CAN bus through a USB adapter and a bus converter so that the messages were converted to Low-Speed CAN - another demand of the SIIS group. Instead of using the actual SVA, we used a HIL that has the same software embedded in its controller as the actuator. Unfortunately, it wasn't possible to make the tests with Low-Speed CAN, because the ACM wasn't ready to output its messages into the Low-Speed port, so we made the tests with only the High-Speed CAN. Besides the SIIS tests, we also added the CTT software in the procedure so that we could test more of the CANopen features and we could be more sure that the device is ready to be implemented in a real scenario. The setup of this part of the tests was the same as with the SIIS tests.

Also, we had the idea of automating those tests so that they are easily reproducible and could be used with other devices. We did this by implementing the scripts from the SIIS tests on Indra Works in an XM22 controller. This controller was connected to the CAN bus and sent the commands through the same bus converter used in the manual setup to the HIL. Unfortunately, we were only able to develop three of the SIIS tests due to time restrictions.

After executing the manual tests (SIIS and CTT), we were able to identify many flaws that would be critical in a real scenario. With the prominence of the PDO not working properly, not being possible to make updates in the firmware and not being able to change the bit rate while in application mode.

Concerning the automatic tests, although they failed exactly like the SIIS tests, the results were very satisfactory, it was enough to prove that with enough time and iterations, the automatic tests could be used to make CANopen devices evaluations really quickly.

In conclusion, even though the device didn't pass many of the tests, this work was fundamental for the PJ-SAS - the group from Bosch Rexroth that is developing the SVA - because through this work it was possible to identify some flaws of the ACM, that were not yet identified by the team before this work and that need to be fixed before implementing the technology into a real scenario. However, after identifying those mistakes, we believe that only a few iterations are necessary to make the interface much better.

Regarding the final decision as to whether the SVA attained the TRL 5 on the software side, we determined that the technology is not TRL 5 in this area yet, due to not being able to execute some of the SIIS tests that would be important from the customer's perspective, not passing in some of the CTT tests and not being able to run with Fault-Tolerant CAN.

However, we expect that these problems are easily and quickly fixable - since the results make it clear where the issues lie - and, after their correction and some iterations of the tests are repeated., the SVA will be able to attain the TRL 5 from the software point of view.

Based on the conclusions and results of this work, these are some perspectives of future works:

1. Fix the errors described in this paper and run another iteration of tests.
2. Automate all the SIIS level 2 tests.
3. Execute the tests with the real SVA.
4. Execute the tests with Low-Speed CAN.
5. Instead of using the CAN bridge and a bus converter, use a device that is able to do both at the same time.

REFERÊNCIAS

AQUASIGN. **Why Are Oil and Gas Trees Called Christmas Trees?** [S.l.], 2022. Available from: https://www.aquasign.com/2017/why-are-subsea-trees-called-christmas-trees/. Visited on: 22 Dec. 2022.

AUTOMATION, CAN in. **CANopen application layer and communication profile**. [S.l.], 2021.

BAI, Yong; BAI, Qiang. **Subsea Structural Engineering**. [S.l.], 2010.

BERVEN, Jon. **Subsea production control systems for all-electric Xmas trees**. [S.l.], 2013.

BOSCH REXROTH. **SVA-R2**. [S.l.], 2022. Available from: https://www.boschrexroth.com/en/xc/sustainable-subsea-operations/. Visited on: 2 Dec. 2022.

CIA. **CiA**. [S.l.], 2022. Available from: https://www.can-cia.org/. Visited on: 5 Dec. 2022.

CLAUS, David. **Design and implementation of a CANopen 443 profile instance for the Bosch Rexroth SVA subsea valve actuator**. [S.l.], 2022.

DICKSON, Robbie. **Everything You Need to Know About Linear Actuators**. [S.l.], 2018. Available from: https://www.firgelliauto.com/blogs/actuators/linear-actuators-101. Visited on: 20 Jan. 2023.

DRUMOND, Claire. **What is scrum?** [S.l.], 2021. Available from: https://www.atlassian.com/agile/scrum. Visited on: 25 Jan. 2023.

FALCH, Martin. **CANopen explained**. [S.l.], 2022. Available from: https://www.csselectronics.com/pages/canopen-tutorial-simple-intro. Visited on: 21 Nov. 2022.

FUNAKUBO, Hiroyasu. **Actuators For Control**. 2. ed. Japan: CRC Press, 1991.

FUTURE, Market Research. **Subsea System Market Size**. [S.l.], 2022. Available from: `https://www.marketresearchfuture.com/reports/subsea-system-market-4448`. Visited on: 25 Oct. 2022.

IBISWORLD. **IBISWorld Oil and Gas Exploration and Production Market Size**. [S.l.], 2022. Available from: `https://www.ibisworld.com/global/market-size/global-oil-gas-exploration-production/`. Visited on: 25 Oct. 2022.

INSTITUTE, American Petroleum. **Recommended Practice on Subsea Production System Reliability, Tecnical Risk, and Integrity Management**. [S.l.], 2017.

INVESTOPEDIA. **What Percentage of the Global Economy Is the Oil and Gas Drilling Sector?** [S.l.], 2022. Available from: `https://www.investopedia.com/ask/answers/030915/what-percentage-global-economy-comprised-oil-gas-drilling-sector.asp#citation-8`. Visited on: 25 Oct. 2022.

JIANG, Jason. **Aker Solutions wins China subsea contract**. [S.l.], 2018. Available from: `https://splash247.com/aker-solutions-wins-china-subsea-contract/`. Visited on: 16 Jan. 2023.

MACKENZIE, Rory; HALVORSEN, Glenn-Roar; VEDELD, Henrik. **Subsea All Electric – A Game Changing Technology Going Forward**. [S.l.], 2020.

ORTH, Alexandre; NEDO, Amadeu Placido; GOTTFRIED, Hendrix. **Enabling All-Electric Subsea Control System Without Compromising Safety - A Case Study Comparing Functional Safety Systems Using Springs or Batteries**. [S.l.], 2022.

PANJAITAN, Toga. **Verification of subsea facilities**. [S.l.], 2017. Available from: `https://www.dnv.com/services/verification-of-subsea-facilities-3361`. Visited on: 17 Jan. 2023.

PETRO KNOWLADGE. **Subsea Systems, Processes and Topside Interface for Oil Gas Production**. [S.l.], 2023. Available from: `https://petroknowledge.com/courses/subsea-systems-processes-and-topside-interface-for-oil-gas-production`. Visited on: 17 Jan. 2023.

PFEIFFER, Olaf; AYRE, Andrew; KEYDEL, Christian. **Embedded Networking With CAN and CANopen**. [S.l.], 2003.

REXROTH, Bosch. **RD expenditure as a proportion of sales**. [S.l.], 2021. Available from: `https://www.boschrexroth.com/en/dc/company/figures/`. Visited on: 22 Jan. 2023.

SIIS. **SIIS**. [S.l.], 2022. Available from: `https://siis-jip.com/`. Visited on: 2 Dec. 2022.

SIIS. **Level 2 Test Specification**. [S.l.], 2016.

SILVA, João Pedro Duarte da. **EXPERIMENTAL AND THEORETICAL ANALYSIS OF AN ELECTRO-HYDROSTATIC ACTUATOR FOR ULTRA DEEPWATER APPLICATIONS**. [S.l.], 2019.

SOTOODEH, Karan. **All-Electric Subsea Control Systems and the Effects on Subsea Manifold Valves**. [S.l.], 2019.

THORAT, Sachin. **What is hydraulic actuators | Types Of hydraulic Actuators**. [S.l.], 2023. Available from: `https://learnmech.com/what-is-hydraulic-actuators-types-of-hydraulic-actuators/`. Visited on: 19 Jan. 2023.

TIPS, Motion Control. **What are rotary actuators and how do they differ from other rotary tables?** [S.l.], 2021. Available from: `https://www.motioncontroltips.com/what-are-rotary-actuators-and-how-do-they-differ-from-rotary-tables/`. Visited on: 20 Jan. 2023.

TWI. **Was sind Technology Readiness Levels (TRLs)?** [S.l.], 2021. Available from: `https://www.twi-global.com/locations/deutschland/was-wir-tun/haeufig-gestellte-fragen/was-sind-technology-readiness-levels-trls`. Visited on: 13 Jan. 2023.

VECTOR. **CAN Bus Levels**. [S.l.], 2022. Available from: `https://elearning.vector.com/mod/page/view.php?id=341`. Visited on: 16 Nov. 2022.

XIANG, Gao. **The working principle of electric linear actuator**. [S.l.], 2021. Available from: `https://www.electric-linear-actuators.com/blogs/news-1/the-working-principle-of-electric-linear-actuator`. Visited on: 22 Jan. 2023.

ZAHNG, Qin. **Basics of Hydraulic Systems**. 2. ed. Japan: CRC Press, 2018.