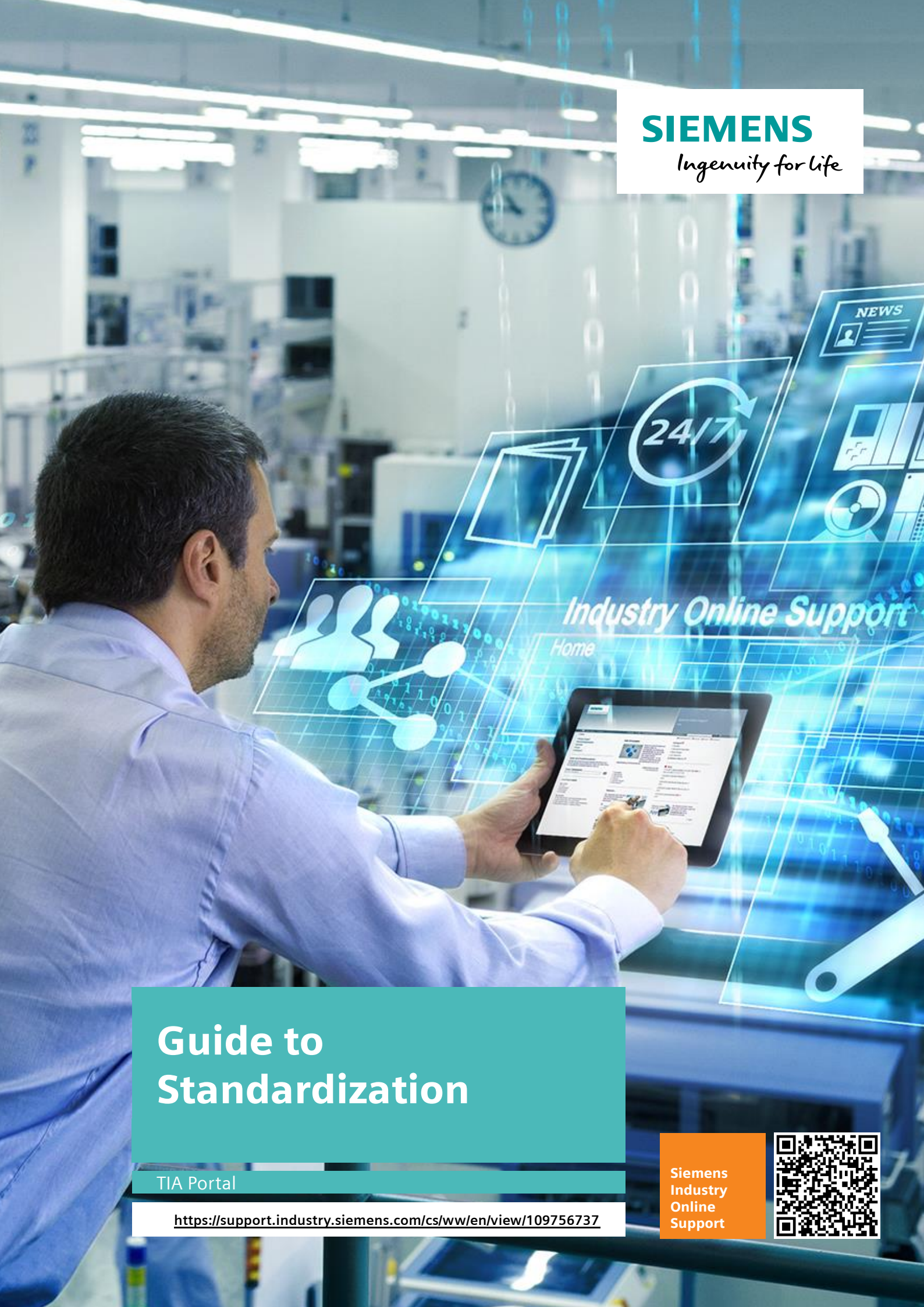




Guide to Standardization

TIA Portal

<https://support.industry.siemens.com/cs/ww/en/view/109756737>

Siemens
Industry
Online
Support



Legal information

Use of application examples

Application examples illustrate the solution of automation tasks through an interaction of several components in the form of text, graphics and/or software modules. The application examples are a free service by Siemens AG and/or a subsidiary of Siemens AG ("Siemens"). They are non-binding and make no claim to completeness or functionality regarding configuration and equipment. The application examples merely offer help with typical tasks; they do not constitute customer-specific solutions. You yourself are responsible for the proper and safe operation of the products in accordance with applicable regulations and must also check the function of the respective application example and customize it for your system.

Siemens grants you the non-exclusive, non-sublicensable and non-transferable right to have the application examples used by technically trained personnel. Any change to the application examples is your responsibility. Sharing the application examples with third parties or copying the application examples or excerpts thereof is permitted only in combination with your own products. The application examples are not required to undergo the customary tests and quality inspections of a chargeable product; they may have functional and performance defects as well as errors. It is your responsibility to use them in such a manner that any malfunctions that may occur do not result in property damage or injury to persons.

Disclaimer of liability

Siemens shall not assume any liability, for any legal reason whatsoever, including, without limitation, liability for the usability, availability, completeness and freedom from defects of the application examples as well as for related information, configuration and performance data and any damage caused thereby. This shall not apply in cases of mandatory liability, for example under the German Product Liability Act, or in cases of intent, gross negligence, or culpable loss of life, bodily injury or damage to health, non-compliance with a guarantee, fraudulent non-disclosure of a defect, or culpable breach of material contractual obligations. Claims for damages arising from a breach of material contractual obligations shall however be limited to the foreseeable damage typical of the type of agreement, unless liability arises from intent or gross negligence or is based on loss of life, bodily injury or damage to health. The foregoing provisions do not imply any change in the burden of proof to your detriment. You shall indemnify Siemens against existing or future claims of third parties in this connection except where Siemens is mandatorily liable.

By using the application examples you acknowledge that Siemens cannot be held liable for any damage beyond the liability provisions described.

Other information

Siemens reserves the right to make changes to the application examples at any time without notice. In case of discrepancies between the suggestions in the application examples and other Siemens publications such as catalogs, the content of the other documentation shall have precedence.

The Siemens terms of use (<https://support.industry.siemens.com>) shall also apply.

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the Internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial security measures that may be implemented, please visit <https://www.siemens.com/industrialsecurity>.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed at: <https://www.siemens.com/industrialsecurity>.

Table of contents

Legal information	2
1 Introduction	4
1.1 Definition Standardization	4
1.2 Purpose of the guidelines	4
1.3 Standardization benefits	4
1.4 Standardization areas	5
2 Norms and Standards	6
2.1 Norm IEC 61131-3	6
2.1.1 General Elements	6
2.1.2 Programming languages	8
2.2 Norm IEC 81346-1	9
2.3 Norm IEC 61512-1 (ISA-88)	10
2.4 PLCopen Standard	11
2.5 OMAC standards	11
2.6 Weihenstephaner Standards	12
3 Procedure models for software development	13
3.1 Waterfall model	13
3.2 V-model	14
3.3 Continuous Integration (CI)	15
4 Tools for standardization	16
4.1 Programming style guide for S7-1200 and S7-1500	16
4.2 Programming Guide for S7-1200 and S7-1500	16
4.3 Programming Guide Safety for S7-1200 and S7-1500	17
4.4 Guide to Library Handling in the TIA Portal	18
4.5 HMI Template Suite	18
4.6 Libraries	19
5 Procedure of standardization using a demo machine as an example	21
5.1 Requirements & Analysis	22
5.1.1 Requirements	22
5.1.2 Analysis	22
5.2 Draft design	24
5.2.1 Modules and module interfaces	24
5.2.2 Data storage	26
5.2.3 Operating and monitoring	27
5.3 Implementation	28
5.3.1 Program	28
5.3.2 Operating and monitoring	30
6 Automatic project generation and testing	32
6.1 TIA Portal Openness	32
6.2 SIMATIC Visualization Architect (SiVArc)	32
6.3 S7 Unit Test	33
7 Appendix	34
7.1 Service and support	34
7.2 Links and Literature	35
7.3 Change documentation	36

1 Introduction

1.1 Definition Standardization

Standardization literally means a standardization of, for example, dimensions, types, procedures and structures. The procedures for standardization are standardization and typing. The aim is to increase compatibility and therefore optimize resources.

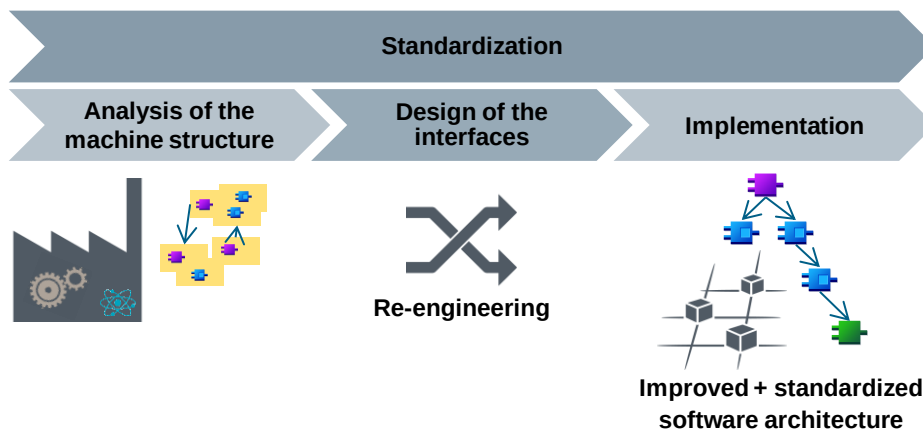
1.2 Purpose of the guidelines

In the development of new machines and systems, standardization possibilities are only used in a few cases today. PLC and HMI projects are often manually copied together from projects of existing machines and adapted at great expense. There is a very wide variety of programming styles and programming language dialects.

The standardization guide shows you how you can modularize your machines and systems. It gives you recommendations and hints for structured and standardized programming of your automation solutions.

An essential component in the development and construction of a standard is the interface definition of modules. In order for modules to be used flexibly, the interfaces should be as independent as possible of special customer requirements and the respective module.

Figure 1-1: Standardization



1.3 Standardization benefits

Standardization has the following advantages for the end customer.

- The software becomes more transparent for the user and thus easier to use.
- Significant reduction of error sources due to using already proven software components.
- Minimization of the effort for program creation and commissioning with existing standard.
- Simplified diagnosis and troubleshooting.
- Clear documentation of modules with defined behavior.
- Avoidance of "parallel engineering" of the same task.
- Defined interfaces to other departments.

Standardization has the following advantages for the manufacturer.

- Requirements of existing customers (e.g. automotive industry) are fulfilled.
- Maintaining and expanding competitiveness.
- Increased engineering efficiency.
- Simplified management of machine variants (flexibility).
- Division of tasks into work packages (clear definitions ensure better communication when working in teams).
- Cost reduction through virtual commissioning / "Digital Twin".

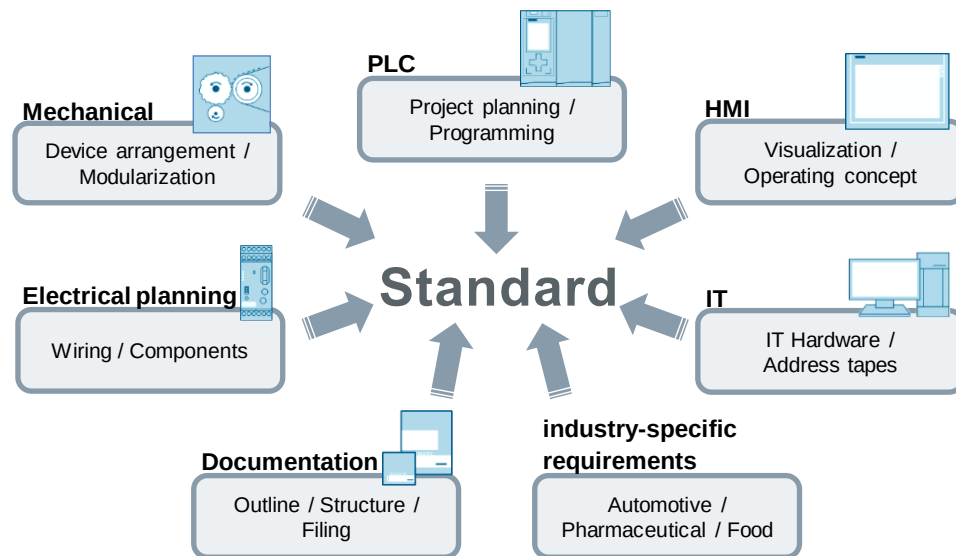
1.4 Standardization areas

When developing new machines and systems, you should consider all areas from planning to commissioning during standardization.

The following standardization options are available:

- During mechanics planning attention should be paid to a modular construction, since individual modules can be reused more easily.
- Electrical planning should define a standard for the hardware to be used and the wiring of signals.
- The programs of the different machines should be created with an engineering tool for one control type.
- A uniform operation and display should be created for template visualization.
- Etc.

Figure 1-2: Standardization areas with examples



2 Norms and Standards

Norms and standards are an important part of standardization. Norms and standards define rules, guidelines or characteristics for activities or their results for general and recurring applications, particularly in industry. By establishing and introducing specifications for recurring applications, national and international standardizations are created for a group of interested parties. Standards can be used to ensure that products and services are suitable, comparable and compatible for the intended purpose.

Some known norms and standards are described below.

2.1 Norm IEC 61131-3

The international standard IEC 61131 deals with the basics of programmable logic controllers. The standard is divided into different areas. IEC 61131-3 (Part 3) is a worldwide standard for programming languages of programmable logic controllers. It provides a basis for uniform PLC programming for the different systems. The standard standardizes the interface between programmer and control system.

The IEC 61131-3 standard consists of two parts.

- General Elements
- Programming languages

The programming languages of e.g. SIMATIC STEP 7 (TIA Portal) meet the requirements of the IEC 61131-3 standard.

2.1.1 General Elements

Data types

The elementary data types Bool, Byte, Integer, Word, etc. are defined in the IEC 61131-3 standard. You use the data types to define the type of the variable.

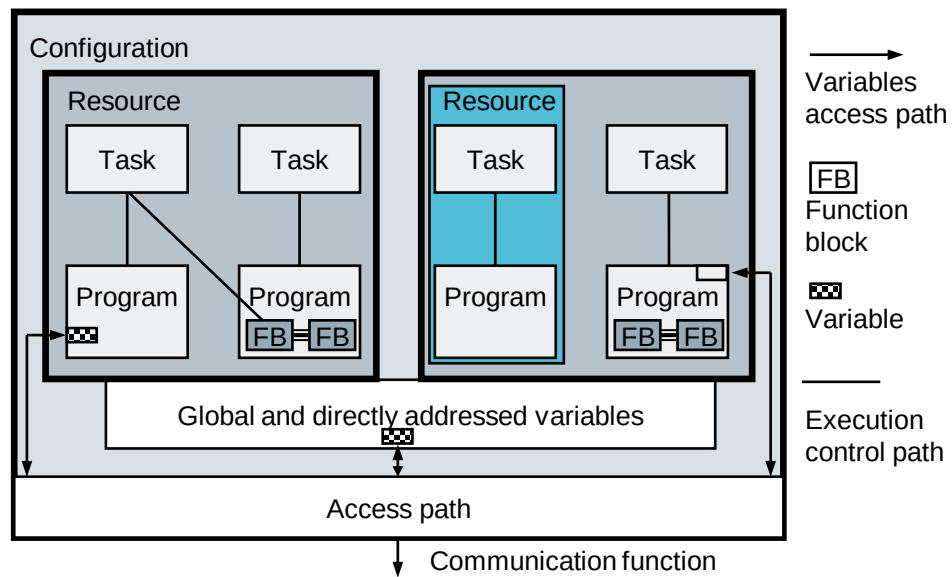
Variables

Variables are identified by a name/identifier. They occupy an address in the controller's memory or an address to the hardware inputs and outputs. Variables are defined with a meaningful name, e.g. "tempSensor1", and with a specific data type (Bool, Integer, etc.). You can use variables to program symbolically and independently of the hardware. This makes the programs more readable, easier to understand and you can reuse them more easily.

Software model

The software model of the IEC 61131-3 standard describes the relationship between the terms configuration, resources, tasks, program, function block and function. The software model (see [Figure 2-1](#)) represents a modern multiprocessor and multitasking capable PLC.

Figure 2-1: Software model, norm IEC 61131-3



The whole system for solving a control task is called configuration.

One or more resources are defined within a configuration. A resource is a control unit. The tasks of a resource control the execution of the associated programs. A conventional PLC (blue marking) consists of a resource, a task and a program.

The programs are created in a language defined in the IEC 61131-3 standard. The programs, function modules and functions are called program organizational units (POUs). You can modularize user programs with POUs. They are the basis for the reuse of once created program code.

IEC has defined general functions, e.g. ADD, SQRT, SIN, COS.

2.1.2 Programming languages

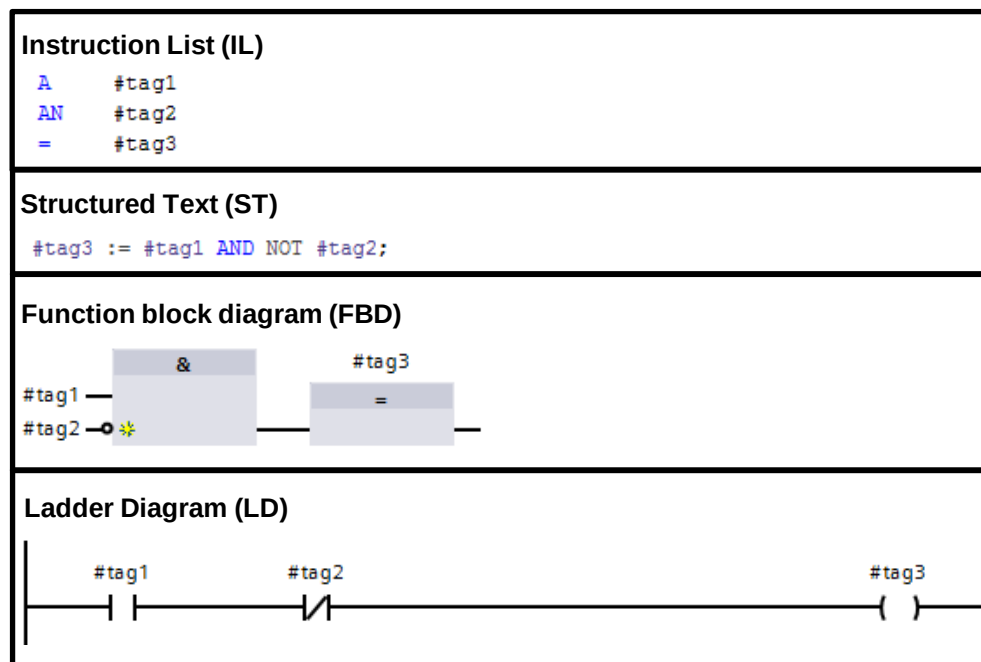
The IEC 61131-3 norm defines four programming languages. This means that the syntax and semantics of the languages are clearly defined.

The following languages are defined:

- **Instruction List (IL)**
In the TIA portal, the Statement List (STL) programming language corresponds to Instruction List (IL) defined in the IEC 61131-3 standard.
- **Structured Text (ST)**
In the TIA Portal, Structured Control Language (SCL) corresponds to the high-level textual language ST defined in the IEC 61131-3 standard.
- **Function Block Diagram (FBD)**
- **Ladder Diagram (LD)**
In the TIA portal, the Ladder Logic (LAD) programming language corresponds to the Ladder Diagram (LD) defined in the IEC 61131-3 standard.

The following figure shows the different programming languages that describe the same simple program part.

Figure 2-2: Programming languages of the IEC 61131-3 standard



The choice of programming language depends on the task to be solved. The programming languages LAD and FBD are well suited for bit processing.

For data management or for more complex calculations, the programming language ST (SCL in the TIA Portal) with the language elements known from the high level language is more suitable.

2.2 Norm IEC 81346-1

It is important to define the required structures as far as possible at the beginning of a project. This makes planning considerably easier and forces a system that leads to higher efficiency.

The standard IEC 81346 Industrial Systems, Plant and Equipment and Industrial Products - Structuring Principles and Reference Labelling shows ways to structure information about systems and to form reference labels (formerly: Equipment Labelling).

The IEC 81346-1 standard (Part 1) shows how to proceed with structuring. Examples of structuring criteria, called "aspect", are presented. However, it is also possible to use other criteria not mentioned. The structure obtained should enable simple embedding in a higher-level system.

A strength of the "aspect objects" concept defined in the standard is that objects can be arranged simultaneously in different hierarchies and yet the object identity is guaranteed. Typically, a system is broken down from the functional aspect as well as from the structural or product component point of view. Another sensible aspect could be the spatial position of the components. The provisions of the standard are very general in order to cover a wide range of applications. The terms used are abstract and are defined as follows:

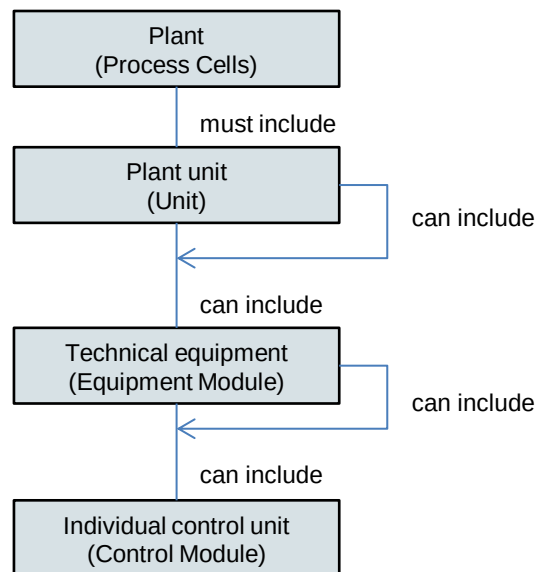
Table 1: Terms of the norm IEC 81346

Term	Meaning
Object	Unit that is treated in a design, planning, implementation, operation, maintenance and dismantling process.
Aspect	Specific approach to selecting information about a system or selecting a system or an object of a system. The norm distinguishes the following aspects: • Functional aspect • Product aspect • Location aspect
Structure	Organization of relationships between objects in a system that describes a component-of-relationship (consists of / is part of). The norm distinguishes the following structures: • Function-related structure • Product-related structure • Location-related structure
Reference code	Unique identification of a specific object in relation to the system of which the object is part. It is based on the aspects of the system.

2.3 Norm IEC 61512-1 (ISA-88)

The IEC 61512-1 standard (Part 1) is a standard for batch-oriented operation. Part 1 describes Models and Terminology. Among other things, the standard describes the structuring of the plant according to the physical model. Only the lower four levels of the model (installations, subsystems, technical equipment and individual control units) (see [Figure 2-3](#)) are specified in more detail. They refer to certain types of equipment. Such a type is a group of process engineering and control equipment that has been combined for a specific purpose.

Figure 2-3: Physical model according to IEC 61512-1



The installation level is the highest level of the technological hierarchy. Below the installation level, the hierarchy folders can then be classified as a sub-assembly and below as a technical installation.

The sub-plant comprises related individual control units and technical equipment in which larger processing activities are carried out.

The technical facility is a functional grouping of individual control units that can perform a finite number of certain smaller processing activities.

Folders or levels that do not have a type are described as neutral and can be used for further structuring or represent the level of the individual control units. The individual control unit is the lowest level of equipment grouping, the basic automation.

2.4 PLCopen Standard

PLCopen is an organization in the field of industrial control technology. Standards are being developed to increase the efficiency of application development and reduce the cost of maintaining such software. PLCopen aims to be independent of certain manufacturers and products and to ensure the dissemination of international standards and their application in a broad field by working in working groups. One of the core activities is located in the vicinity of IEC 61131, the global standard for industrial control programming.

The following standards have been defined by PLCopen:

- PLCopen Motion Control (Library)
- PLCopen Safety (safety properties of controls)
- PLCopen XML (XML specifications for the exchange of project data)
- PLCopen OPC UA for exchange of process data (together with OPC Foundation)
- Rules for determining compatibility between programming tools

Note

Further information on the standards of PLCopen can be found on the organization's homepage.

<http://www.plcopen.org>

2.5 OMAC standards

OMAC (Organization for Machine Automation and Control) is a user organization consisting of end users, machine builders, system integrators and technology providers. The majority of these come from the food and luxury food industry and the packaging industry. OMAC defines guidelines and standards that help manufacturers to shorten development and delivery times, to use existing resources more efficiently and at the same time to increase profitability.

The standard **PackML** has been defined by OMAC.

TR88.00.02 is a concrete implementation concept for packaging machines and was published by OMAC under the name PackML. PackML defines guidelines for machine modes and status and labeling. Information is contained in PackTags. These are information packages with a fixed format and a specific function.

Note

Further information about OMAC can be found on the organization's homepage,

<https://www.omac.org/>

and in the application examples "SIMATIC/SIMOTION OMAC PackML V3 Operating Mode Management",

<https://support.industry.siemens.com/cs/ww/en/view/49970441>

and "Line Integration in the Food and Luxury Food Industry - Overview".

<https://support.industry.siemens.com/cs/ww/en/view/109483779>

2.6 Weihenstephaner Standards

The Weihenstephan Standards (WS) are interfaces for machine data acquisition. They connect machines in packaging and filling technology (WS Pack) or the food industry (WS Food) with higher-level systems such as production data acquisition systems (PDA) or manufacturing execution systems (MES). The Weihenstephan standards define the physical interface, a proprietary instruction set, a machine-specific data offering and the utilization/processing of the data points.

The specifications have been subdivided as follows:

- Physical specification of the communication interface (cross-industry)
- Definition of data content (industry-specific libraries)
- Evaluation recommendations (industry-specific)
- Verification and secure operation (across industries)

During the development of the Weihenstephan standards, attention was paid to the integration of existing standards.

The following Weihenstephaner Standards exist:

- WS communication interface (For the data exchange of machines and higher-level IT systems)
- WS XML (For interface documentation)
- WS Pack (for BDE in beverage filling and packaging systems)
- WS Food (for PDA in machines in the food industry)
- WS Bake (For BDE in the baking industry)
- WS Brew (For brewing processes)

Note

Further information about WS can be found on the organization's homepage.

<http://www.weihenstephaner-standards.de>

3 Procedure models for software development

The demands on software are constantly increasing. The software is no longer developed by just one person. Meanwhile the software is created in a team of people with different know-how and background. Commissioning, testing and maintenance are essential parts of software development. Structured software development divides the programs into individual subprograms based on the function, for example. Structured software development is thus the basis for the reusability of the programs.

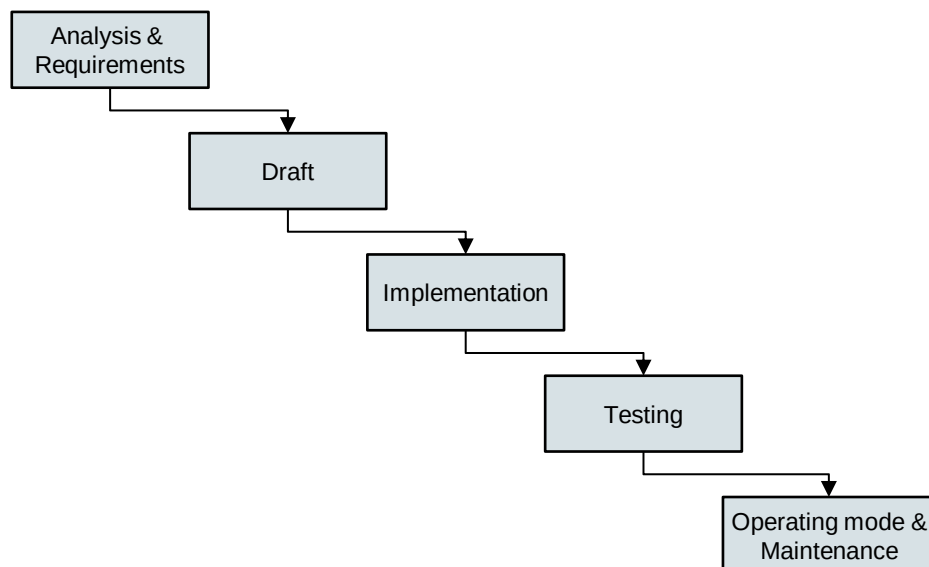
Modern software development processes offer you an approach for structured software development.

For the software development different models for the procedure were developed. The most popular models are described below.

3.1 Waterfall model

The waterfall model is a linear process model that is used especially for software development and is organized in successive project phases. As with a waterfall, the phase results are always used as binding guidelines for the next lower phase.

Figure 3-1: Waterfall model

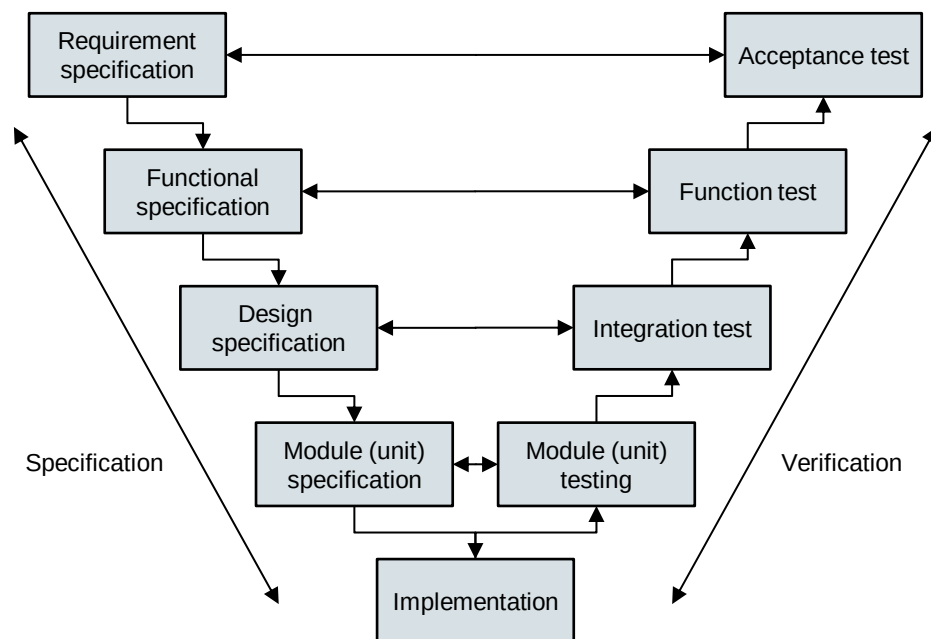


The waterfall model is more suitable for smaller projects.

3.2 V-model

The V-model is a version of the waterfall model extended by quality assurance measures. The V-model is a process model in software development in which the software development process is organized in phases. In addition to these development phases, the V-Model also defines the procedure for quality assurance (testing) in phases. On the left side, a functional/business specification is started, which is being expanded in ever greater detail into a technical specification and implementation basis. At the top, the implementation takes place, which is then tested on the right-hand side against the corresponding specifications on the left-hand side.

Figure 3-2: V-model



The V-model is suitable for large and complex projects. The early definition of the test cases minimizes the project risk and at the same time increases the software quality and transparency in the project.

3.3 Continuous Integration (CI)

Continuous Integration is a practice in software development. Isolated changes are immediately checked and then added to the overall code base of a software. The aim of continuous integration is to provide immediate feedback so that an accidentally integrated error can be identified and corrected as quickly as possible.

The principles can be applied to any iterative programming model, such as agile software development. Traditional development approaches such as the waterfall model can benefit from the methods of continuous integration during the build process.

CI means for TIA Portal:

- Use a central instance on which the entire program code is located and from which anyone can get the current version (and previous versions).
- Automate testing so that all tests are performed before the change is integrated.
- Automate the build process so that anyone can create the project from source with a single command.
- Ensure that everyone can get the most current, executable project.

4 Tools for standardization

The guidelines described in the following give you recommendations for programming and support you in implementing the standardization.

4.1 Programming style guide for S7-1200 and S7-1500

The programming style guide offers you a coordinated set of rules for uniform programming. These specifications describe the uniform assignment of variable and block names up to clear programming in SCL. This makes the programs more uniform, easier to read and understand.

Examples

- Identifiers should always be written in English, be unique and be written in "CamelCasing" notation. They should not exceed a maximum length of 24 characters and should not contain any special characters. (e.g. tempMaxLength)
- The identifier of individual instances should begin with "Inst" (e.g. InstHeater), multi-instances with "inst" (e.g. instHeater).
- Identifiers for constants always consist entirely of CAPITAL LETTERS and contain underscores as word separators (e.g. MAX_VALUE).
- Identifiers for data types have the prefix "type" (e.g. typeDiagnostic).
- Use PLC data types instead of structures (exception for know-how-protected blocks).

You can use these rules and recommendations freely. They serve as a proposal for uniform programming.

Note

The programming style guide for S7-1200 and S7-1500 can be found under the following link:

<https://support.industry.siemens.com/cs/ww/en/view/81318674>

4.2 Programming Guide for S7-1200 and S7-1500

The programming guide gives you many recommendations and hints for optimal programming of S7-1200 and S7-1500 controllers. It helps to standardize the program and define the interface.

Examples

- Create optimized building blocks.
- Program symbolically.
- Avoid ANY-Pointer. Use VARIANT, ARRAY and structures instead.
- Structure your automation task.
- Use functions for frequently recurring applications that are called several times at different points in the user program.
- Use multi-instances to reduce the number of instance DBs.

Note

The programming guide for S7-1200 and S7-1500 can be found under the following link:

<https://support.industry.siemens.com/cs/ww/en/view/81318674>

4.3 Programming Guide Safety for S7-1200 and S7-1500

The Safety programming guide gives you many recommendations and notes for optimum project planning and programming of fail-safe S7-1200 and S7-1500 controllers. He's helping to standardize the security program.

Examples

- Assign separate address ranges for each of the PROFIsafe address types 1 and 2.
- Define the F source addresses for all F CPUs uniquely.
- Divide the program code into modules, e.g. into sub-areas for acquisition, evaluation, reaction or according to system components.
- Create F-compatible PLC data types to structure data in the safety program.
- Create a separate function block for each sensor type (e.g. emergency stop command device, safety door, light curtain, etc.) that bundles the evaluation of the sensor and the necessary auxiliary functions.
- Use global standard data blocks to exchange data between the standard user program and the security program.

Note

The programming guide Safety for S7-1200 and S7-1500 can be found under the following link:

<https://support.industry.siemens.com/cs/ww/en/view/109750255>

4.4 Guide to Library Handling in the TIA Portal

The guide to library handling in the TIA Portal describes the creation of a corporate library using typed library elements. It supports the standardization, versioning and cross-project exchange of programs.

Examples

- Maintain the metadata (developer, version, comment...) of the "development library".
- Use folder structures in your company-specific development library for a better overview.
- Create a folder in your "company-specific development library" for each basic library.

Note

The guide to library handling in the TIA Portal can be found at the following link:

<https://support.industry.siemens.com/cs/ww/en/view/109747503>

4.5 HMI Template Suite

The "HMI Template Suite" provides an attractive template for the visualization of your plant. By using SiVArc you can generate standard elements and adapt the visualization to your plant.

The "HMI Template Suite" offers the following:

- An intuitive operating concept, optimized for touch operation.
- Allows you to seamlessly integrate your own content into this operating concept.
- Examples for the representation of different operator screens.
- Enables automatic visualization with SiVArc (optional).

Note

The "HMI Template Suite" can be found under the following link:

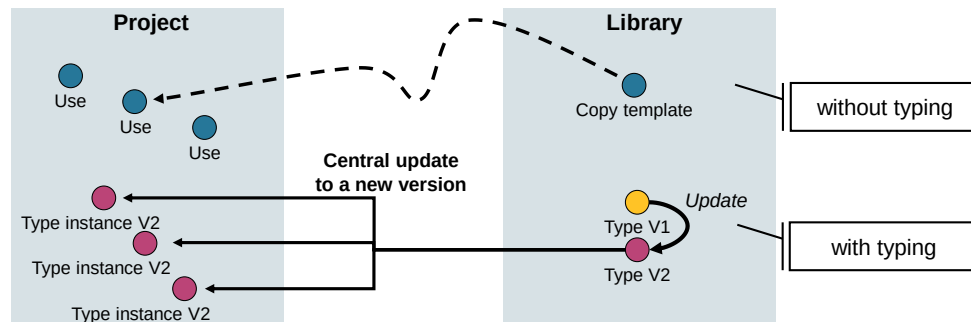
<https://support.industry.siemens.com/cs/ww/en/view/91174767>

4.6 Libraries

The libraries are used for easy storage of the objects projected in the TIA Portal. The libraries enable a cross-project exchange of TIA Portal objects. You can create frequently used objects once and save them as standard in a library and reuse them for other projects. The objects can be saved as templates or types.

The following figure shows the difference between copy model and type.

Figure 4-1: Difference between copy templates and types



Templates

Almost any object can be saved as a copy template and pasted back into the project later. You can use copy templates to create independent copies of the library object in your project. For example, the following elements can be created as templates in the library:

- Devices with their device configuration
- Variable tables or individual variables
- Statement profiles
- Watch tables
- Elements from the documentation settings, e.g. cover sheets
- Modules and groups containing several modules
- PLC data types (UDTs) and groups containing several PLC data types
- Text lists
- Message classes
- Technology objects

Types

Objects that are necessary for the execution of user programs are suitable as types. Types are versionable and therefore support professional further development. You use Types to create instances and use the instances in your project. Projects in which the types are used can be updated as soon as new versions of the types are available.

The following elements can be created as types in the library:

- Functions (FCs)
- Function blocks (FBs)
- PLC data types (UDTs)
- Image building blocks
- Images
- Styles
- User-defined functions

The functions and function blocks that are to be created as types must be library-compliant. To ensure this, the following must be observed:

- No access to global data blocks or individual instances in the block
- No access to global variables and constants

Whether a block is library-compliant can be checked in the properties of the block. The block must be translated.

Note

Examples of libraries with program blocks can be found in the following application examples:

Library with general functions (LGF) for STEP 7 (TIA Portal) and S7-1200 / S7-1500

<https://support.industry.siemens.com/cs/ww/en/view/109479728>

Library with PLC data types (LPD) for STEP 7 (TIA Portal) and S7-1200 / S7-1500

<https://support.industry.siemens.com/cs/ww/en/view/109482396>

"MindConnectFB" library for SIMATIC S7-1500 and MindSphere V3.0

<https://support.industry.siemens.com/cs/ww/en/view/109756878>

Note

Further information and examples of image modules can be found in the following application examples:

Creation of image modules for WinCC Runtime Advanced and Comfort Panels

<https://support.industry.siemens.com/cs/ww/en/view/68014632>

Basic Process Library for WinCC V7 und Basic Process Library for TIA Portal (S7-1200/S7-1500/Runtime Professional) with sample project

<https://support.industry.siemens.com/cs/ww/en/view/109749508>

Example modules for WinCC (TIA Portal) and STEP 7 (TIA Portal) for S7-1200 and S7-1500 in SCL

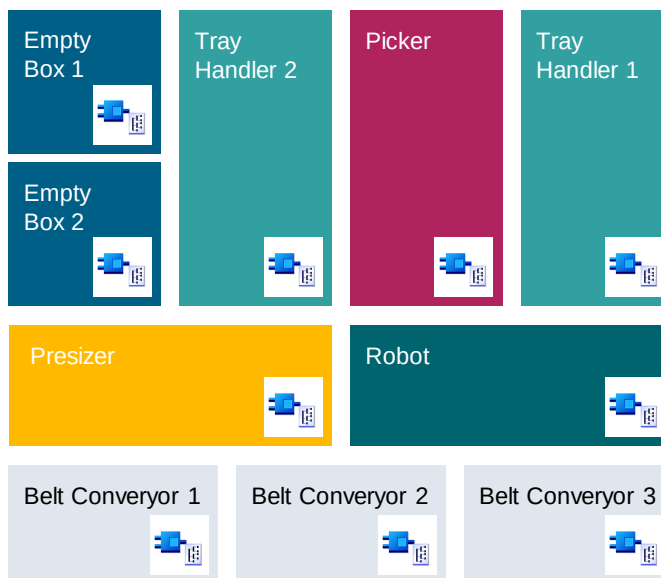
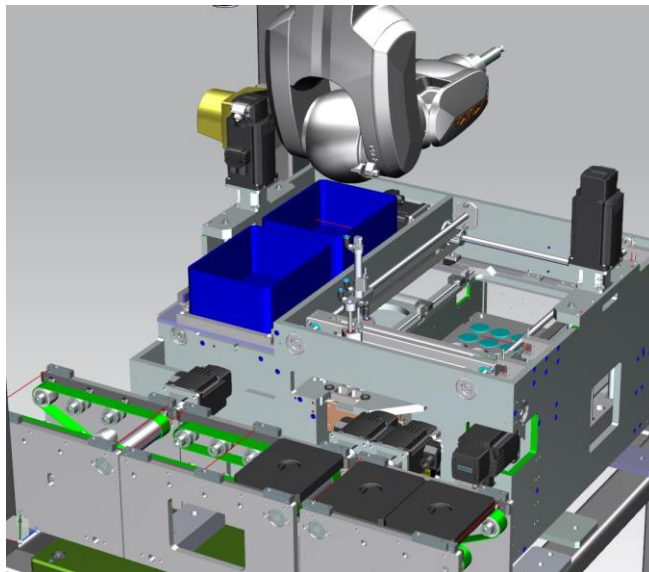
<https://support.industry.siemens.com/cs/ww/en/view/66839614>

5 Procedure of standardization using a demo machine as an example

This chapter describes on a demo machine (see [Figure 5-1](#)) how you should basically proceed with the standardization of a machine. The example describes software development according to the waterfall model, limited to the analysis, design and implementation phases.

The following figure shows the 3D model and the functional units of the demo machine.

Figure 5-1: Demo machine



5.1 Requirements & Analysis

5.1.1 Requirements

When planning a machine or system, you should first define the requirements for the software. A distinction is made between functional and non-functional requirements. The functional requirements determine what the machine or system should do. The non-functional requirements describe how well the machine should perform. The entire requirements are summarized in a specification sheet.

Implementation in the demo machine

For example, the software of the demo machine should meet the following functional requirements:

- The components (circles and rectangles) are provided on trays in the magazine.
- The gripper removes a component from the top tray and places it in the transfer position.
- The robot removes the parts from the transfer position and places them on workpiece carriers.
- Conveyor belts transport the workpiece carriers past the robot.
- Once all workpiece carriers are loaded, the robot removes the components again and sorts them into the storage boxes.

For example, the software of the demo machine should meet the following non-functional requirements:

- The robot is to deposit a part within three seconds.
- The machine should sort the components error-free.
- The machine should be easily expandable so that further components (e.g. triangles) can be processed.

5.1.2 Analysis

In the analysis phase you structure and structure your machine. The IEC 81346-1 standard (see chapter 2.2) describes how you can structure your machine. Dividing the machine into functional units is an important step towards standardization.

Implementation in the demo machine

In this example, the machine is divided into the following functional units according to the functional aspect (see [Figure 5-1](#)):

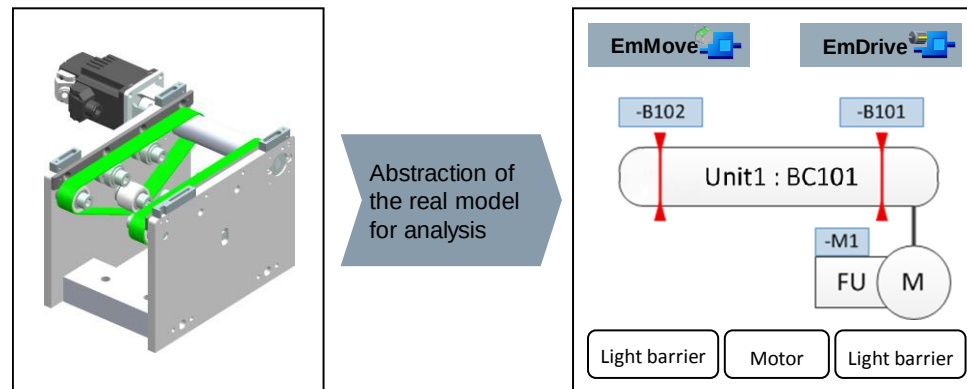
- Belt Conveyor 1
- Belt Conveyor 2
- Belt Conveyor 3
- Empty Box 1
- Empty Box 2
- Tray Handler 1
- Tray Handler 2
- Picker
- Robot
- Presizer transfer station

The division of the machine shows that some functions occur several times.

These functions are also referred to as "subsystems (units)". You can further subdivide these subsystems into "Technical Equipment Modules" and "Individual Control Modules" (see chapter 2.3).

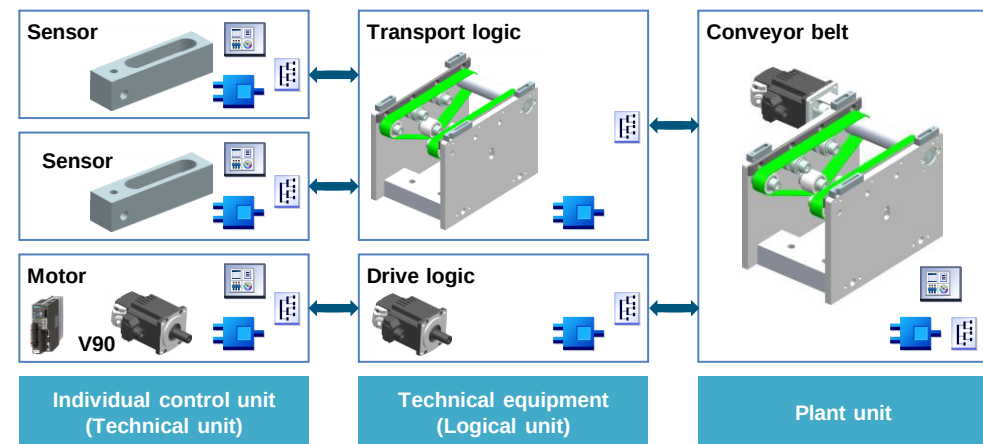
In the example, you can divide the conveyor belt sub-system into the individual control units "motor" and "light barriers". The individual control units are the "EmMove" and "EmDrive" technical equipment. (see Figure 5-2).

Figure 5-2: Abstraction of the conveyor belt section



The conveyor belt sub-system is represented by the connection of the technical units to form logical units and the connection of the logical units to the sub-system. Each unit takes care of its function.

Figure 5-3: Construction of the conveyor belt section



Result

The result of this analysis is a specification sheet.

When the machine is structured, the structure of the program is determined at the same time.

5.2 Draft design

The technical design is now implemented on the basis of the specifications.

This phase defines the following:

- Standard modules for processing actuators and sensors and for unified modules.
- Interfaces of the modules
- Communication interfaces
- Data retention
- Operation
- Programming languages
- Program sequences
- Libraries
- Technology used
- Define concept for software maintenance, development and distribution
- Documentation and review

5.2.1 Modules and module interfaces

In the Analysis phase, you have already structured the machine. In the Design phase, you define the functions of the individual modules and the call structure of the program.

The main work in the development and construction of a standard is the definition of the module interface. The signal exchange between the user program and the individual modules should always take place via the module interfaces. The interfaces should be defined as "openly" as possible so that the module can be used flexibly.

The following questions arise when defining the interfaces:

- Who exchanges data with whom?
- What information should be exchanged?

Advantages

Standardized modules offer the following advantages

- High parameterization (parameter transfer to interface)
- High degree of reusability (Library)
- Central changeability (Library)

Passing parameters at the module interface has the following advantages:

- The program can be assembled modularly from finished modules with partial tasks.
- Program is easily expandable and maintainable.
- Program code is easier to read and test because there are no hidden cross accesses.

Restrictions of the standardized modules

- More complex software structure
- Harder to analyze

Recommendations

Define modules with "open" interfaces for recurring functions.

Simplify the modules / functions as much as possible.

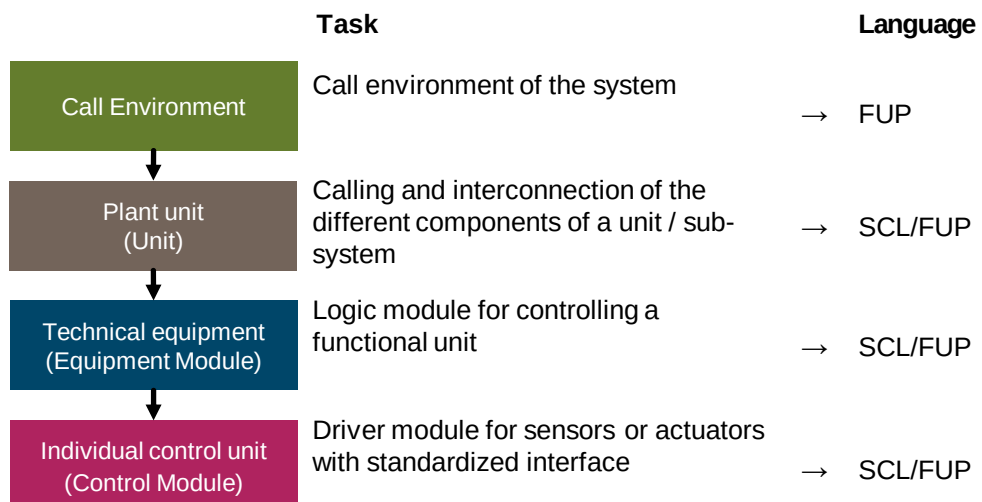
Define user-defined PLC data types for the interface parameters, so:

- Data can be exchanged easily via the module interface.
- Unintentional wrong connection of interfaces is prevented.
- Changes to the PLC data type are updated at all points of use.

Conversion in the conveyor belt section

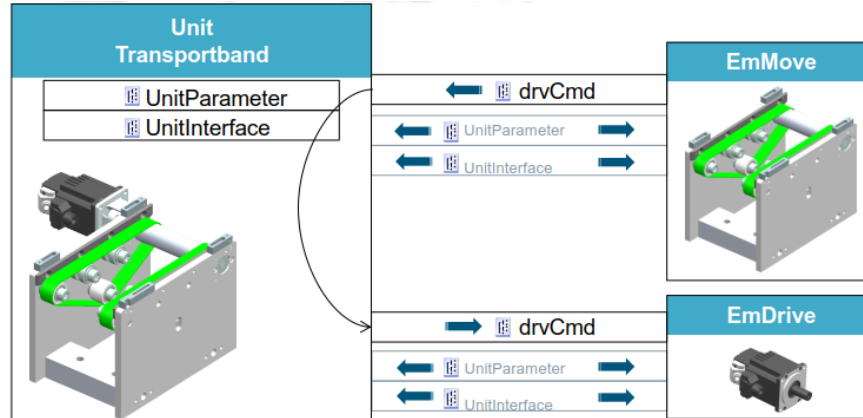
In the example of the conveyor belt subsystem, the tasks are distributed to the individual units (see [Figure 5-3](#)) according to the analysis (see [Figure 5-4](#)). For each task, standard blocks are defined in the programming languages FBD and SCL. This means that changes (e.g. the replacement of the motor) can be adapted in a modular manner.

Figure 5-4: Allocation of tasks to the units



Two PLC data types are defined for data exchange in the demo machine via the block interface. The PLC data type "UnitParameter" contains the parameters and the PLC data type "UnitInterface" contains the process values of the conveyor belt.

Figure 5-5: Interface parameters of the conveyor belt subsystem



5.2.2 Data storage

Here you define where and how the data of the machine is to be stored.

Recommendations

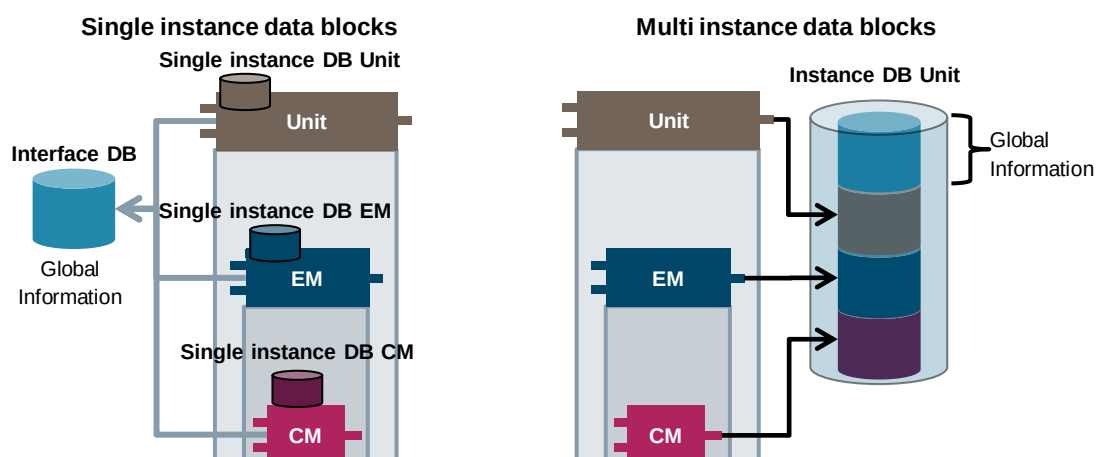
Store global data that is to be used by several program parts in global data blocks.

Use PLC data types to store related data.

The instance data of the function blocks can be stored globally in individual instance data blocks or collectively in multi-instance data blocks. (see [Figure 5-6](#)).

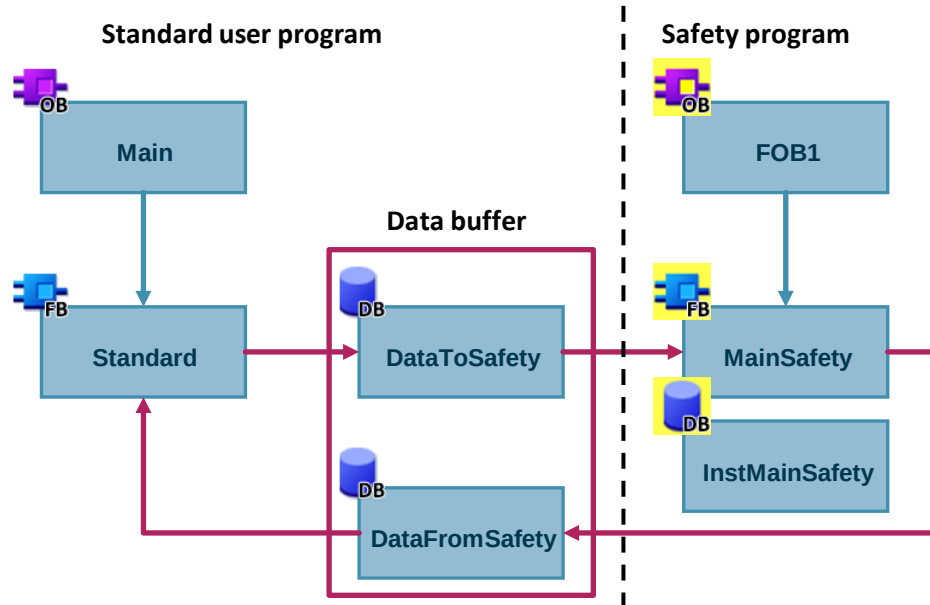
Use multi-instance data blocks to make the program easier to read, maintain and reuse. The disadvantage is that information may have to be looped through by the different instances.

Figure 5-6: Memory storage single instance vs. multi instance



To prevent data inconsistencies between the standard user program and the security program, you should use data buffers in the form of global data modules that are used for data exchange.

Figure 5-7: Data buffer between standard user and safety program



5.2.3 Operating and monitoring

Operating and monitoring the machine is an important part of automation. This should be uniform for the machine operators in the line. Machines from different manufacturers should be easy to operate and diagnose. When creating the visualization, standardization is also possible.

Recommendation

- When creating the visualization, standardization is also possible.
- Define faceplates for frequently used operating and display objects.
- Define PLC data types for the connection of image blocks to the control program of image blocks.
- Use the HMI Template Suite (see chapter 4.5).
- Use an OMAC-compliant operating mode management.

Conversion in the conveyor belt section

For the conveyor belt subsystem, a faceplate is defined for controlling, displaying status information and animating the conveyor belt subsystem.

PLC data types are defined for the connection of the faceplates to the user program.

5.3 Implementation

The modules defined in the design and the operating and monitoring concept are now implemented in the implementation phase.

5.3.1 Program

During implementation, the modules defined in the Design phase become a program. The program is created with one or more programming languages.

Recommendations

Divide the tasks of software development into sub-areas. In this way, a programmer can be assigned to each subarea.

Create the program according to the recommendations and rules described in the guidelines (see chapter 4).

Use the grouping function of the project navigation to group the hardware, the program blocks and the PLC data types.

Create optimized modules so that the program performs better.

Create standard functions for recurring functions.

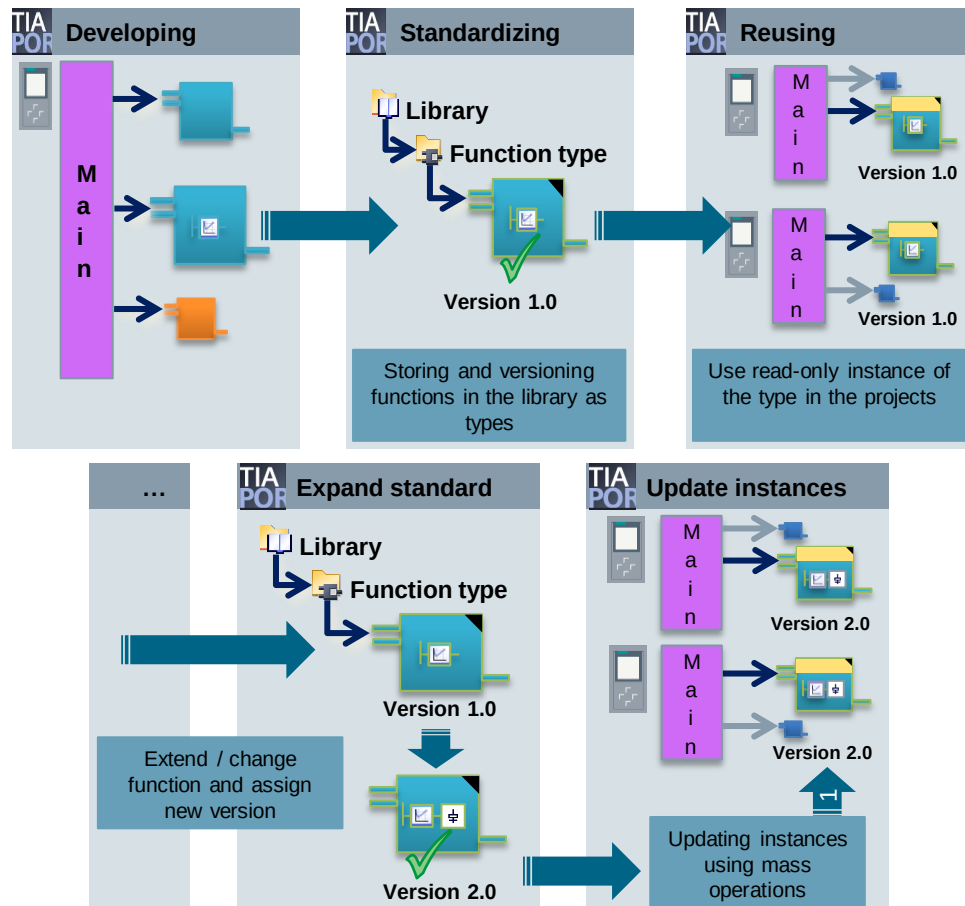
Use libraries to create your software.

Standardization of program blocks

Frequently used objects can be created once, tested and saved as standard in a library for reuse for other projects.

The following figure shows schematically how you type a program block by saving it in the library. Instances of this type can be reused in projects. If you extend or change the type, a new version is automatically generated. You can update the version in your project with a mass operation.

Figure 5-8: Standardization of program blocks



5.3.2 Operating and monitoring

During implementation, the HMI images defined in the Design phase are implemented in HMI project planning.

Recommendations

Divide the visualization tasks into subareas. This means that a project planner can be assigned to each subarea.

Use the grouping function of the project navigation to group the HMI images and the HMI variables.

Create faceplates for frequently used operating and display objects.

Standardization of image modules (faceplates)

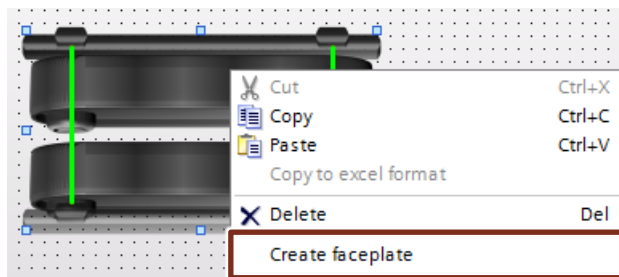
A screen module is a configured group of display and control objects. "Data exchange" with the display and operating objects contained in the screen module takes place via an interface on the created screen module. Screen blocks can be managed and modified centrally in a library (reusability).

Through the use of PLC data types, self-created HMI image blocks can be quickly and clearly interconnected. The interface of the screen modules can be formed from one or more data types.

The following steps show how to create an image block for the animation of the conveyor belt sub-system and interconnect the process variables with the variables of the PLC data type. The generated version of the image block is saved in the project library.

1. Create image block

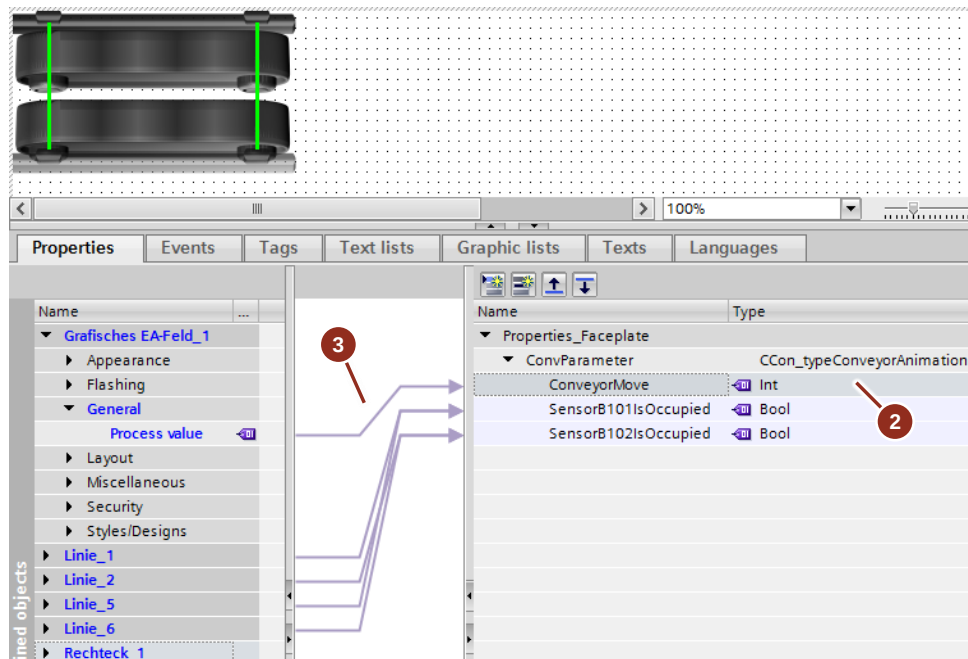
Figure 5-9: Create image block



5 Procedure of standardization using a demo machine as an example

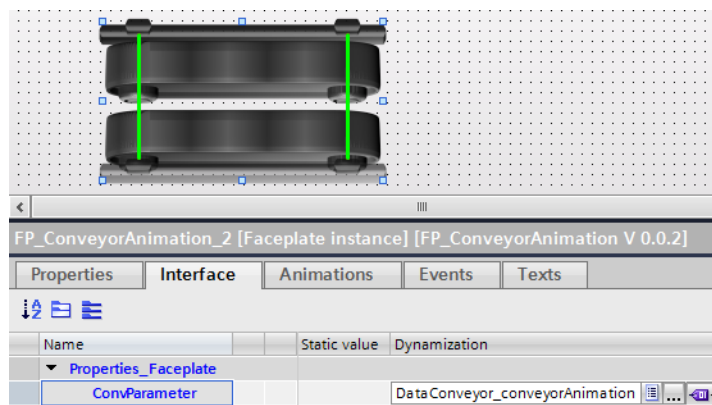
2. Create parameters with the corresponding PLC data type
3. Linking the properties of the individual objects via "Drag&Drop" with the parameters of the PLC data type

Figure 5-10: Configuring and connecting the screen module



4. Assign a variable with the corresponding PLC data type to the screen block interface

Figure 5-11: Assign variables to the screen module interface



6 Automatic project generation and testing

The future goal of standardization is the automatic generation of projects and automated testing. This procedure makes it possible to easily configure machines to avoid errors. This reduces the time to market and the effort of the IBS on-site.

The TIA Portal already offers the possibility to generate program components and HMI images.

6.1 TIA Portal Openness

TIA Portal Openness is used to generate program modules.

Requirements

- Standardized program modules are available
- A generation program has been created
- License required

Note

Further information about Openness can be found under the following link

<http://www.industry.siemens.com/topics/global/en/tia-portal/software/details/Seiten/openness.aspx>

and in the application example "TIA Portal Openness: Introduction and demo application"

<https://support.industry.siemens.com/cs/ww/en/view/108716692>

6.2 SIMATIC Visualization Architect (SiVArc)

The SiVArc software option automatically creates your visualization using the control program. To do this, you define rules that determine the assignment between visualization elements (e.g. screen modules) and program modules.

Requirements

- TIA Portal with STEP 7 Professional and at least WinCC Comfort
- Image modules are available
- Rules have been created
- License required

Note

Further information about SiVArc can be found at the following links

<https://www.industry.siemens.com/topics/global/en/tia-portal/software/details/seiten/sivarc.aspx>

<https://w3.siemens.com/mcms/automation-software/en/tia-portal-software/tia-portal-optionen/sivarc/seiten/default.aspx>

6.3 S7 Unit Test

With the S7UnitTest application, parts of a PLC program can be automatically tested with SIMATIC S7-PLCSIM Advanced.

S7UnitTest consists of two tools:

- S7UnitTest Editor
- S7UnitTest Application

S7UnitTest Editor

With the tool S7UnitTest Editor you create and edit a test unit for your PLC program and save it in an S7UnitTest XML file. A test unit consists of test cases with individual test steps. In the test steps you determine which PLC variables are read or written in a specific program cycle.

S7UnitTest Application

With the S7UnitTest application you load the created S7UnitTest XML file and start the tests. The application starts the virtual controller of PLCSIM Advanced for the simulation of the loaded PLC program and accesses the program in the memory storage of the virtual controller and executes the tests. The test result is displayed directly on the user interface and is also stored as an XML file.

Note

Further information on S7 Unit Test can be found in the application example "S7UnitTest: Automatic testing with "SIMATIC S7-PLCSIM Advanced".

<https://support.industry.siemens.com/cs/ww/en/view/109746405>

7 Appendix

7.1 Service and support

Industry Online Support

Do you have any questions or need assistance?

Siemens Industry Online Support offers round the clock access to our entire service and support know-how and portfolio.

The Industry Online Support is the central address for information about our products, solutions and services.

Product information, manuals, downloads, FAQs, application examples and videos – all information is accessible with just a few mouse clicks:

<https://support.industry.siemens.com>

Technical Support

The Technical Support of Siemens Industry provides you fast and competent support regarding all technical queries with numerous tailor-made offers – ranging from basic support to individual support contracts. Please send queries to Technical Support via Web form:

www.siemens.com/industry/supportrequest

SITRAIN – Training for Industry

We support you with our globally available training courses for industry with practical experience, innovative learning methods and a concept that's tailored to the customer's specific needs.

For more information on our offered trainings and courses, as well as their locations and dates, refer to our web page:

www.siemens.com/sitrain

Service offer

Our range of services includes the following:

- Plant data services
- Spare parts services
- Repair services
- On-site and maintenance services
- Retrofitting and modernization services
- Service programs and contracts

You can find detailed information on our range of services in the service catalog web page:

<https://support.industry.siemens.com/cs/sc>

Industry Online Support app

You will receive optimum support wherever you are with the "Siemens Industry Online Support" app. The app is available for Apple iOS, Android and Windows Phone:

<https://support.industry.siemens.com/cs/ww/en/sc/2067>

7.2 Links and Literature

Table 7-1

No.	Topic
\1\	Siemens Industry Online Support https://support.industry.siemens.com
\2\	Link to this entry page of this application example https://support.industry.siemens.com/cs/ww/en/view/109756737
\3\	Homepage PLCopen http://www.plcopen.org
\4\	Homepage OMAC https://www.omic.org/
\5\	Application Example "SIMATIC/SIMOTION OMAC PackML V3 Operating Mode Management" https://support.industry.siemens.com/cs/ww/en/view/49970441
\6\	Application example "Line integration in the food and luxury food industry - overview". https://support.industry.siemens.com/cs/ww/en/view/109483779
\7\	Homepage Weihenstephaner http://www.weihenstephaner-standards.de
\8\	Programming style guide for S7-1200 and S7-1500 https://support.industry.siemens.com/cs/ww/en/view/81318674
\9\	Programming Guide for S7-1200 and S7-1500 https://support.industry.siemens.com/cs/ww/en/view/81318674
\10\	Programming Guide Safety for S7-1200 and S7-1500 https://support.industry.siemens.com/cs/ww/en/view/109750255
\11\	Guide to Library Handling in the TIA Portal https://support.industry.siemens.com/cs/ww/en/view/109747503
\12\	"HMI Template Suite" https://support.industry.siemens.com/cs/ww/en/view/91174767
\13\	Library with general functions (LGF) for STEP 7 (TIA Portal) and S7-1200 / S7-1500 https://support.industry.siemens.com/cs/ww/en/view/109479728
\14\	Library with PLC data types (LPD) for STEP 7 (TIA Portal) and S7-1200 / S7-1500 https://support.industry.siemens.com/cs/ww/en/view/109482396
\15\	"MindConnectFB" library for SIMATIC S7-1500 and MindSphere V3.0 https://support.industry.siemens.com/cs/ww/en/view/109756878
\16\	Application example "Creation of image modules for WinCC Runtime Advanced and Comfort Panels" https://support.industry.siemens.com/cs/ww/en/view/68014632
\17\	Application example "Basic Process Library for WinCC V7 and Basic Process Library for TIA Portal (S7-1200/S7-1500/Runtime Professional) with sample project". https://support.industry.siemens.com/cs/ww/en/view/109749508
\18\	Application example "Example modules for WinCC (TIA Portal) and STEP 7 (TIA Portal) for S7-1200 and S7-1500 in SCL" https://support.industry.siemens.com/cs/ww/en/view/66839614
\19\	Product page Openness http://www.industry.siemens.com/topics/global/en/tia-portal/software/details/Seiten/openness.aspx

No.	Topic
\20\	Application example "TIA Portal Openness: Introduction and demo application" https://support.industry.siemens.com/cs/ww/en/view/108716692
\21\	Product page SiVArc https://www.industry.siemens.com/topics/global/en/tia-portal/software/details/seiten/sivarc.aspx
\22\	TIA Portal Option SiVArc https://w3.siemens.com/mcms/automation-software/en/tia-portal-software/tia-portal-optionen/sivarc/seiten/default.aspx
\23\	Application example "S7UnitTest: Automatic testing with "SIMATIC S7-PLCSIM Advanced" https://support.industry.siemens.com/cs/ww/en/view/109746405

7.3 Change documentation

Table 7-2

Version	Date	Modifications
V1.0	09/2018	First version