Uni-REPM Embedded System Module

Version 1.0

Authors: Tarcísio Couto Pereira
Comments: jffv

March 11, 2020

Contents

1. OS Organizational Support

This main process area evaluates the amount of support given to requirements engineering practices from the surrounding organization. Organizational support is important, since ultimately the success of any time-consuming activity needs to be understood and supported by the organization. The embedded system module added tow new areas: Process Requirements for Embedded Systems and Acquire and Supply Products or Services.

1.1. OS.PR Process Requirements for Embedded Systems

Purposes: (1) To identify a set of stakeholders who have a direct interest in the system development. Additionally, the stakeholders and their roles must be defined, registered and shared with the project team. Requirements engineering for embedded systems is a multi-disciplinary approach. It requires domain experts from several areas, such as mechanical engineers and electrical engineers. Particular attention should be taken on the identification of the stakeholders. Depending on the embedded system context, stakeholders with different skills may be required.

(2) To identify and select a set of implementation technology and tools to be used during the requirements engineering. A Standard is defined to ensure that the recommended technology will be employed through the project.

Inputs: Project constraints.

Actions:

- **OS.PR.a1:** Identify and document the stakeholders involved during the system life cycle. Requirements engineering for embedded systems is a multi-disciplinary approach. It requires domain experts from several areas who are responsible for system requirements and constraints. **(Level 1)**;

Examples of stakeholders [18]:
▷ Clinician, patient, APP, hospital pharmacy, and physician;
▷ Mechanical engineers for physical context;
▷ Electrical engineers for the hardware context;
▷ Human-Machine-Interface (HMI) experts for the usability aspects;
▷ Requirements engineers, software developers, testers, and maintainers with a computer science background.

- **OS.PR.a2:** Include in the documentation the roles and responsibilities performed by each stakeholder. This information is helpful to deal with different characteristics of requirements engineering **(Level 1)**;

  Examples of stakeholders [18]:
  ▷ Clinician - initialization, attachement, basal infusion, detatchment;
  ▷ Patient - extra dose upon patient-determined need;
  ▷ Clinician or App - suspend audible alarm from ICE;
  ▷ Human Machine-Interface expert for the usability aspects;

- **OS.PR.a3:** Elicit and document stakeholder requirements. This information include their needs, wants, desires, and expectations about the operation of the system **(Level 1)**;

- **OS.PR.a4:** Define one or more tools, or an environment, required to support the activities towards the requirements engineering **(Level 2)**;

- **OS.PR.a5:** Define a set of implementation technology such as programming languages, code pattern, and modeling languages to support the embedded systems development **(Level 1)**;

  Examples of implementation technology:
  ▷ UML, SysML, MARTE, and iStar;
  ▷ Assembler, C, Java, and Prolog;

- **OS.PR.a6:** Maintain and make available the requirements engineering document throughout the product life-cycle. Embedded real-time systems typically have long product life cycles from inception to retirement. It is a result of many changes in the environment of the system, such as components, and operational local **(Level 2)**.

Work Products:

- **OS.PR.WP1 - System Stakeholders:** The stakeholders, their roles and responsibilities, and their requirements documented and published for the project team [OS.PR.a1, OS.PR.a2, OS.PR.a3].

- **OS.PR.WP3 - Tools and Environment:** Tools and/or an environment established to support all requirements engineering activities [OS.PR.a4].

- **OS.PR.WP7 - Implementation Technology:** The technologies to support the embedded systems development [OS.PR.a5];

- **OS.PR.WP8 - Requirements Document:** A requirements document available and accessible by all authorized stakeholder [OS.PR.a6];

1.2. OS.ASP Acquire and Supply Products or Services

Purpose: To acquire and/or supply products or services from/for other companies. The supply process describes an agreement between two companies under which one of them acquires products or services from the other, while the acquirer process describes an agreement between two companies under which one of them supply products or services to the other.

Inputs: Acquirer need, supplier candidates, supplier enterprise, acquirer enterprise, and acquirer request.

Actions:

- **OS.ASP.a1:** Establish an acquisition plan including selection criteria and a schedule of milestones **(Level 3)**;
- **OS.ASP.a2:** Identify and document a set of supplier candidates for the project that is being carried out. The suppliers are selected according to selection criteria and their suitability to meet the overall need **(Level 3)**;
- **OS.ASP.a3:** Establish a formal contract with the negotiated terms including the requirements for accept delivery of product or service **(Level 1)**;
- **OS.ASP.a4:** Analyze and evaluate a set of acquirer requests in order to deliver a response that meets acquirer needs and complies with industry and other standards **(Level 2)**;
- **OS.ASP.a5:** Establish a formal contract with the negotiated terms including the requirements to delivery of product or service **(Level 3)**.

Work Products:

- **OS.ASP.WP1 - Acquisition Process:** A process that describes how a company will acquire products or services from the other. The scope of the acquisition process should include activities that are related to the definition of an acquisition plan, supplier selection, agreement negotiation, determination of the compliance with agreement, and rendering payment [OS.ASP.a1, OS.ASP.a2, OS.ASP.a3];
- **OS.ASP.WP2 - Supply Process:** A process that describes how a company will supply products or services to the other. The scope of the supply process should include activities that are related to the identification of the acquire, contract negotiation, execution agreement, delivering product or service and acknowledging payment [OS.ASP.a4, OS.ASP.a5].

2. PM Requirements Process Management

The requirements process management covers all the activities to manage, control requirements change as well as to ensure the organization of the process and coherence among team members. The embedded system module defines a new sub-process called Contextual Information for Embedded Systems.

2.1. PM.CI Contextual Information for Embedded Systems

Purpose: To identify and specify the contexts that can affect the system operation. It involves the description and modeling of the statements, facts, and monitored variables and association to the software and hardware behaviors.

Inputs: RE.DSG.WP1 System description, RE.IEA.WP1 Variables definition, and DS.DBR.WP1 Behavioral requirements

Actions:

- **PM.CI.a1:** Analyze the system requirements, behavioral requirements, and variables to identify the contexts that can affect the system operation **(Level 2)**;
- **PM.CI.a2:** Identify and document the context information. According to [1], contexts can be analyzed through a set of facts and statements. The goal is to get a formula of facts supported by monitored variables (defined in RE.IEA sub-process) **(Level 2)**.

  Supporting action(s):
  - RE.IEA.a1 Establish and document the monitored and controlled variables
  - RE.IEA.a2 Define the type, range, and units required for all variables
  - RE.IEA.a4 Define a status attribute (invalid/valid) for each monitored variable

  Example of statements, facts, and variables [29].
  ▷ Context: C1
  ▷ Statement: occlusion occurs;
  ▷ Fact1: flow rate is less than X;
  ▷ Variable1: basalFlowRate;

- **PM.CI.a3:** Associate the contexts with the software or hardware behaviors defined in DS.DBR sub-process **(Level 2)**;

Supporting action(s):
- DS.DBR.a1 Provide a set of software behaviors to document the actions the software should perform
- DS.DBR.a2 Provide a set of hardware behaviors to document the actions the hardware should perform

Example:
▷ Context: C1;
▷ Hardware Behavior: DS.DBR.a2 - HB2.

- **PM.CI.a4:** Perform modeling to visually represent the contexts that can affect the system operation. Thus, the graphical representation of the context diagrams provide a way to show the external entities it interacts with and those interactions [19] **(Level 3)**.

  Supporting action(s):
  - OS.PR.a4 Define one or more tools, or an environment, required to support the activities towards the requirements engineering
  - OS.PR.a5 Define a set of implementation technology such as programming languages, code pattern, and modeling languages to support the embedded systems development

Work Products:

- **PM.CI.WP1 - Contextual Information Document:** A document describing in natural language the components that describe contextual information [PM.CI.a1, PM.CI.a2, PM.CI.a3];
- **PM.CI.WP2 - Contextual Models:** A diagram with a representation of all contexts related to the system operation [PM.CI.a4].

3. EL Requirements Elicitation

Elicitation is the process of discovering, understanding, anticipating and forecasting the needs and wants of the potential stakeholders in order to convey this information to the system developers. The potential stakeholders can include customers, end users and other people who have the stake in the system development. In the process, the application domain and organizational knowledge are necessary among other things.

3.1. RE.ER Environmental Requirements for Embedded Systems

Purposes: In this sub-process, stakeholders work together to understand the application domain, the business rules required by the system and the environment in which the system will operate.

(1) To describe the business and business rules of the system. [3] affirms the importance of considering the business process in an organization because the system must fit the organizational expectations and needs. Additionally, this sub-process should specify the environment under which the system will operate.

(2) To identify and describe the target domain. It is related to the identification and documentation of the system's scope.

Inputs: Existing assets and glossary. Examples of existing assets: business policies, standards, business environmental requirements, regulatory statues, scenarios, actors, and constraints.

Project constraints, system operational platform, business processes, business rules, stakeholders.

Actions:

- **RE.ER.a1:** Elicit and include in the business documentation a set of business processes in which the embedded system will operate **(Level 2)**;

  Supporting action(s):
  - OS.PR.a2 Include in the documentation the roles and responsibilities performed by each stakeholder. This information is helpful to deal with different characteristics of requirements engineering

Examples of business processes [18]:
▷ The process of monitor a patient's blood oxygenation and pulse rate;
▷ The process of infuse narcotic or liquid pain-killer at a prescribed basal rate;

- **RE.ER.a2:** Elicit and include in the business documentation a set of business rules **(Level 2)**;

  Examples of business rules [18, 24]:
  ▷ The PCA pump should be available 24 hours a day;
  ▷ The system shall allow drug infusion to one patient at a time;
  ▷ The (external) pump should be able to operate at relative humidity ranging from Hmin = 0% to Hmax = 100% (non-condensing).

- **RE.ER.a3:** Elicit and document a set of environmental information. Environmental information refers to the operating environment (e.g., technical infrastructure, temperature, climate conditions, humidity) in which the system will be installed **(Level 3)**.

  Examples of environmental description:
  ▷ Background information about the environment;
  ▷ Modeling of the technical infrastructure;
  ▷ Description of the temperature, humidity, pressure, voltage, etc;
  ▷ Identification and description of the agents that interact with the environment: the environment of the PCA pump is the patient, the clinician, the prescribing physician, the hospital room, and the hospital pharmacy.

  Examples of environmental information
  [18, 29]:
  ▷ Controlled ambient temperature;
  ▷ The PCA pump will operate in a hospital room or similar clinical setting;
  ▷ Source of power is an implementation choice, defaulting to 60 Hz 120V;
  ▷ The pump shall be able to operate as intended within a temperature range of x to y degrees Celsius;
  ▷ The pump should be able to withstand and operate as intended under atmospheric pressure ranging from x to y mmHg.

- **RE.ER.a4:** Model and document the business processes identified in RE.ER.a1 **(Level 3)**;

  Supporting action(s):
  - OS.PR.a4 Define one or more tools, or an environment, required to support the activities towards the requirements engineering
  - OS.PR.a5 Define a set of implementation technology such as programming languages, code pattern, and modeling languages to support the embedded systems development
  - RE.ER.a1 Elicit and include in the business documentation a set of business processes in which the embedded system will operate

- **RE.ER.a5:** Elicit information about domain restrictions and technical infrastructure of the system **(Level 1)**;

  Supporting action(s):
  - OS.PR.a2 Include in the documentation the roles and responsibilities performed by each stakeholder. This information is helpful to deal with different characteristics of requirements engineering

> Examples of technical infrastructure [18]:
> ▷ Systems installation platform;
> ▷ External hardware that interacts with the system;
> ▷ External software that interacts with the system: Integrated Clinical Environment (ICE) - A system which allows clinician's to remotely monitor the operation of the pump.

Work Products:

- **RE.ER.WP1 - List of Business Processes and Business Rules:** A document that contains a list of business processes and business rules that will be used as input in the business modeling action. The scope of the list should include only information that will serve to the embedded system development [RE.ER.a1, RE.ER.a2];
- **RE.ER.WP2 - List of Environmental Information:** A document containing a list of the environmental information necessary for the correct system operation [RE.ER.a3].
- **RE.ER.WP3 - Modeled Business Processes:** A document that summarizes the modeled business processes [RE.ER.a4];
- **RE.ER.WP4 - Domain Constraints:** A document that captures the domain constraints imposing on the system. The document can contain models depicting physical and/or logical constraints, and description of regulatory needs [RE.ER.a5];
- **RE.ER.WP5 - Technical Infrastructure Considerations:** A document describing the operational environment under which the system will be installed [RE.ER.a5];

3.2. RE.DSG Define Embedded System Goals

Purpose: To define and describe the system goals. It is related to the system description, statements that depict what the system must accomplish, and the management of the system goals. This information can be used to derive functional and non-functional requirements.

Inputs: OS.PR.WP1 System Stakeholders, RE.ER.WP2 List of environmental information, RE.ER.WP3 Modeled Business Processes, RE.ER.WP5 Domain Constraints, RE.ER.WP6 Technical Infrastructure Considerations, RE.ER.WP7 System's Operational Domain, and RE.ER.WP8 System Boundaries.

Actions:

- **RE.DSG.a1:** Provide short statements describing what the system must accomplish. The statements should summarize the system capabilities **(Level 1)**;

  Supporting action(s):
  - OS.PR.a3 Elicit and document stakeholder requirements. This information include their needs, wants, desires, and expectations about the operation of the system
  - RE.ER.a1 Elicit and include in the business documentation a set of business processes in which the embedded system will operate
  - OS.PR.a1 Identify and document the stakeholders involved during the system life cycle. They are responsible for system requirements and constraints

  Examples of system statement [18, 11]:
  ▷ A patient-controlled analgesia (PCA) infusion pump infuses narcotic, liquid pain-killer at a prescribed basal rate plus any bolus doses that the patient may request to alleviate their pain, or be commanded by an attending clinician, most often, a registered clinician.
  ▷ Patient-Controlled Analgesic (PCA) Pump is used to pump an analgesic drug (opioids, pain-killers) into a patient's intravenous infusion (IV) line to provide pain relief;
  ▷ The infusion is "patient controlled" because the patient presses a button to release a dose of the drug. The "burst of drug" released when the button is pressed is called a "bolus dose".

- **RE.DSG.a2:** Elicit and specify a set of system goals from stakeholder's needs **(Level 1)**;

  Supporting action(s):
  - OS.PR.a3 Elicit and document stakeholder requirements. This information include their needs, wants, desires, and expectations about the operation of the system

  Examples of system goals [18]:
  ▷ G1— The pump will safely infuse drugs intravenously for pain relief;
  ▷ G2— The patient should receive enough drug to reduce his pain;
  ▷ G3— Clinician(s) should be notified upon occurrence of hazardous conditions, unless alarms have been inactivated.

- **RE.DSG.a3:** Manage the goals to include information such as origin, origin date, author, priority, stakeholders, and schedule date **(Level 2)**.

Work Products:

- **RE.DSG.WP1 - System Description:** A document containing a description of what the system does, including statements that describe what the system must accomplish, and a set of system goals. This document provides insights to derive functional and non-functional requirements [RE.DSG.a1, RE.DSG.a2, RE.DSG.a3].

3.3. RE.SHC Software and Hardware Constraints

Purpose: The goal of this sub-process is to provide a set of software and hardware constraints to ensure that software and hardware components will be in accordance with the architecture, design, or implementation constraints. It is important to note that the constraints can be imposed for technical reasons or business reasons. Here, the results aim at producing a set of constraints which are useful for the software and hardware engineers on the embedded software development.

Inputs: RA.MMP.WP2 Legal Constraints, OS.ASP.WP1 Acquisition Process, OS.ASP.WP2 Supply Process, RE.ER.WP5 Domain Constraints, RE.ER.WP6 Technical Infrastructure Considerations, RE.ER.WP7 System's Operational Domain, and RE.ER.WP8 System Boundaries.

Actions:

- **RE.SHC.a1:** The constraints for each hardware device previously identified are defined and documented. These constraints are related to cost, physical size, physical weight, CPU, memory, energy (e.g., voltage), water and/or fire resistance, temperature, operating voltage, the architecture of the circuit, and others **(Level 1)** [28].

Fictitious examples of hardware constraints:
▷ The final hardware of the embedded system shall weigh no more than 5 kilograms;
▷ The embedded system shall only use CPU from company "ABC";
▷ The physical dimension of the product shall have approximate dimensions of 242mm x 228mm x 32mm;
▷ The power supply shall be tolerant to electric current of 110 and 220 volts;

- **RE.SHC.a2:** Identify and document the software constraints. These constraints are related to program memory space, design constraint, implementation constraint, and code size **(Level 1)**.

Examples of software constraints that are not related to the pump:
▷ All software components shall be programmed in C programming language;
▷ The size of the software code should not exceed 5MB;
▷ The response time of the software functions should not exceed 0.5 second;
▷ The design pattern to be used must be the First Class ADT pattern;

Work Products:

- **RE.SHC.WP1 - Hardware Constraints:** A document containing hardware constraints and all relevant information to support the specification of the most suitable hardware devices. This document includes considerations about the brand and the device vendor [RE.SHC.a1];
- **RE.SHC.WP2 - Software Constraints:** A document describing the software constraints of the embedded system. The constraints will be useful to support the embedded software development [RE.SHC.a2].

3.4. RE.IEA Identification of Environmental Assumptions for Embedded Systems

Purpose: To identify specific assumptions about the environment in which the system will operate. It is related to the inputs the system will accept, the outputs it will produce, and constraints required by the system to its correct operation. This information can be specified by monitored and controlled variables that describe environment behavior. Incorrect environmental assumptions can be the cause of several system failures, such as the number of architectural elements (e.g., control variables). Correct assumptions can benefit system reuse and integration [19].

Inputs: RE.ER.WP2 List of Environmental Information, RE.ER.WP5 Domain Constraints, RE.ER.WP7 System's Operational Domain, and RE.DSG.WP1 System Description.
Actions:

- **RE.IEA.a1:** Establish and document the monitored and controlled variables **(Level 1)**;

  Supporting action(s):
  - RE.ER.a2 Elicit and include in the business documentation a set of business rules
  - RE.ER.a3 Elicit and document a set of environmental information. Environmental information refers to the operating environment (e.g., technical infrastructure, temperature, climate conditions, humidity) in which the system will be installed

Examples of monitored and controlled variable [18]:
▷ Volume To Be Infused (VTBI) - controlled;
▷ How fast to infuse the drug (Rate) - controlled;
▷ Basal flow rate - controlled;
▷ External temperature - monitored;
▷ Relative humidity - monitored;
▷ Atmospheric pressure - monitored;

- **RE.IEA.a2:** Define the type, range, and units required for all variables **(Level 1)**;

  Supporting action(s):
  - RE.ER.a2 Elicit and include in the business documentation a set of business rules
  - RE.ER.a3 Elicit and document a set of environmental information. Environmental information refers to the operating environment (e.g., technical infrastructure, temperature, climate conditions, humidity) in which the system will be installed

  Examples of a monitored variable:
  ▷ Name: external temperature;
  ▷ Type: integer;
  ▷ Range: [10..50];
  ▷ Units: Celsius;

- **RE.IEA.a3:** Provide the reasons why the environmental assumptions were included **(Level 2)**;

  Supporting action(s):
  - RE.ER.a1 Elicit and include in the business documentation a set of business processes in which the embedded system will operate
  - RE.ER.a2 Elicit and include in the business documentation a set of business rules
  - RE.ER.a3 Elicit and document a set of environmental information. Environmental information refers to the operating environment (e.g., technical infrastructure, temperature, climate conditions, humidity) in which the system will be installed

  Example of rationale:
  ▷ The current temperature will be provided to the panel in celsius degree;
  ▷ Rationale: consistency with external interface. (All temperatures will be displayed in celsius degree.);

- **RE.IEA.a4:** Define a status attribute (invalid/valid) for each monitored variable **(Level 1)**;

  Examples [7]:
  ▷ External temperature: isTemperatureOutOfRange Boolean;
  ▷ Relative humidity: isHumidityOutOfRange Boolean;

- **RE.IEA.a5:** Establish a list of devices that monitor the monitored and controlled variables **(Level 1)**. Supporting action(s):
  - DS.DEH.a5 Identify and document a set of sensors. Thus its functionalities and relevant design rationale are registered, including the required / provided interfaces. The documentation must also contain a tutorial on how the sensors should be used

Example [7]:
▷ External temperature: environment sensor controller;

Work Products:

- **RE.IEA.WP1 - Variables Definitions:** A document describing a set of monitored and controlled variables including its particularities, their rationales, and the devices it is related [DS.DEH.a1, DS.DEH.a2, DS.DEH.a3, DS.DEH.a4, DS.DEH.a5].

4. RA Requirements Analysis

Requirements gathered from different sources need to be analyzed to detect incomplete or incorrect ones as well as to estimate necessary information for later activities (e.g. risk, priorities…). It is also recommended that you should perform some analysis to dismiss irrelevant requirements to avoid wasting effort in next steps.

4.1. RA.SA Safety Analysis for Embedded Systems

Purpose: To Identify and manage risks, and software and hardware failures aiming to maintain product integrity and reliability. A risk assessment should be performed on the requirements identified so far to estimate possible problems. An action plan must be prepared and executed to overcome those risks [25, 21]. The failure occurrence can be dangerous and can cause accidents. Thus, it is necessary to carefully manage and provide a mitigation strategy to resolve a situation of failure.

Inputs: DS.ESR.WP2 Software Requirements Documentation, DS.EHR.WP5 Hardware Requirements Documentation,
DS.DBR.WP1 Behavioral Requirements, and RV.IHS.WP1 HW/SW Co-Verification Plan.

Actions:

- **RA.SA.a1:** Identify and document potential risks. It relies on determining risks that might affect the embedded system execution and safety **(Level 1)**;

Examples of risk area according to [15]:
▷ Development process, architecture, design, implementation, verification & validation, project management, and people;

Examples of risks [30]:
▷ Area: software source
▷ Risk1: pump unexpectedly restores to default factory settings without the user's awareness;
▷ Contributing factor: user inadvertently selects a restore of the factory settings;
▷ Area: hardware source
▷ Risk2: audio notifications or prompts too loud;
▷ Contributing factor: abnormal audio device; incorrect audio volume settings.

- **RA.SA.a2:** Analyze and check the identified risks as acceptable or not acceptable. Additionally, document their impact on the system performance, cost, schedule, scope, and probability of its occurrence **(Level 1)**;

  Example:
  ▷ Risk1: acceptable;
  ▷ Risk2: acceptable;

- **RA.SA.a3:** Develop an action plan to overcome each risk. It is related to define actions or decisions that will contribute to reducing the probability or the impact of each risk **(Level 3)**;

- **RA.SA.a4:** Identify, classify, and document potential software and hardware failures. The goal relies on the identification of deviations from the expected (hardware and / or software) behavior [8] **(Level 1)**.

  Examples of kinds of failures [16]:
  ▷ Mechanical (deterioration, shock);
  ▷ Electronic hardware (noise, heat);
  ▷ Software (design defects);
  ▷ Outside influence (people, environment);

  Examples of failures [18]:
  ▷ Kind of failure1: electronic;
  ▷ Failure1.1: internal electronic failure. PCA pump detects its own failure (memory fails, processor fails, power supply fails, and others);
  ▷ Failure1.2: no audible alarm;

- **RA.SA.a5:** Analyze the identified failures, and associate the probability to each one (frequent, probable, occasional, remote, and implausible). Additionally, the impact of each failure on the system performance, cost, schedule, and scope are documented **(Level 2)**;

  Example:
  ▷ Failure1.1: remote;
  ▷ Failure1.2: occasional.

- **RA.SA.a6:** Develop a mitigation strategy plan to overcome each failure. It is related to the definition of actions or decisions to mitigate the faulty situation. Contextual information may be used to represent failures, and actions to mitigate them **(Level 3)**.

- **RA.SA.a7:** Specify the fault containment requirements. The goal is to ensure that a failure in one component must not propagate to cause failure in another component **(Level 3)**;

  Supporting action(s):
  - RA.SA.a4 Identify, classify, and document potential software and hardware failures.

- **RA.SA.a8:** Identify the safety functional requirements. The goal is to describe what actions and/or constraints should or should not be performed in order to maintain the system in a safe state **(Level 2)**;

  Supporting action(s):
  - DS.DBR.a1 Provide a set of software behaviors to document the actions the software should perform
  - DS.DBR.a2 Provide a set of hardware behaviors to document the actions the hardware should perform

Work Products:

- **RA.SA.WP1 - Risk and Action Plan:** The risk plan contains descriptions of risks, the type of risk (acceptable or not acceptable), and an action plan to overcome each risk [RA.SA.a1, RA.SA.a2, RA.SA.a3];
- **RA.SA.WP2 - Failure and Action Plan:** A document describing failures, its probability, and an action plan to overcome each failure [RA.SA.a4, RA.SA.a5, RA.SA.a6];
- **RA.SA.WP3 - Safety Strategy:** A safety strategy is defined and documented. A plan to avoid failures in components is established as well as the functional safety requirements [RA.SA.a7, RA.SA.a8].

4.2. RA.MMP Management of Market Pressure

Purpose: To investigate and manage the consumer changes, environmental, and regulatory needs. All of these can impact the requirements engineering since it is necessary to move fast and change requirements to facilitate the agility and flexibility needs of its container.

Inputs: Consumer changes requests, changes of the operational environment, and legal/ regulatory constraints.
Actions:

- **RA.MMP.a1:** Analyze, evaluate, store, and implement the consumer requests/needs according to the project needs **(Level 3)**;
- **RA.MMP.a2:** Analyze, evaluate, and store the changes of the operational environment of the system **(Level 3)**;

- **RA.MMP.a3:** Document and share with the project team the applicable legislation, organizational constraints, industry standards, and regulatory needs **(Level 1)**.

> Examples of industry standards and regulatory constraints [18, 4]:
> ▷ ASTM International F2761-09 Medical Devices and Medical Systems;
> ▷ IEC 60601-1-8 Medical electrical equipment;
> ▷ FDA: Guidance – Total Product Life Cycle: Infusion Pump-Premarket Notification Submissions;
> ▷ Medical Device Process Standard IEC 62304;
> ▷ ISO 14971:2007 - Medical devices – Application of risk management to medical devices.

Work Products:

- **RA.MMP.WP1 - Change Plan:** A document containing the changes required by the consumers, including reasons why they have requested the changes, and the plan to perform the approved changes. Additionally, the document must contain the changes of the operational environment under which the system will operate, including size, humidity, etc [RA.MMP.a1, RA.MMP.a2];

- **RA.MMP.WP2 - Legal Constraints:** A document describing a set of considerations/ implications such as applicable legislation, organizational constraints, industry standards, and regulatory needs that should be considered during the system development [RA.MMP.a3].

4.3. RA.ALD Abstraction Level Definition for Embedded Systems

Purpose: The goal of the Abstraction Level Definition for Embedded System is to define the abstraction level of the models that will be produced at each sub-process. Stakeholder has different skills. Therefore it is necessary the definition of representations and the level of detail of them.

> Examples of six abstraction levels [2]:
> ▷ Scenarios, Functions, Functional-network;
> ▷ Logical system architecture;
> ▷ Technical system architecture;
> ▷ Electrical and Mechanical plant;
> ▷ Implementation;

Inputs: OS.PR.WP1 System Stakeholders and RA.MMP.WP2 Legal Constraints.
Actions:

- **RA.ALD.a1:** Identify and define different kinds of representations, such as models, to be produced by the sub-processes. The representations can be the system architecture, system and hardware requirements, and electrical and mechanical plant **(Level 1)**;

  > Examples of representations:
  > ▷ Define Embedded System Goals: natural language;
  > ▷ Embedded Software Requirements: scenarios;
  > ▷ Define Embedded Hardware Devices: natural language narrative.

- **RA.ALD.a2:** Identify and associate the stakeholders who have different skills with each model to be produced **(Level 1)**;

  > Examples of representations:
  > ▷ Natural language - system user;
  > ▷ Scenarios - requirements engineer;
  > ▷ Natural language narrative - electrical and mechanical engineer.

- **RA.ALD.a3:** Establish the levels of detail of the representations based on the different skills of the people involved after the stakeholder's identification **(Level 2)**.

> Examples of level of detail:
> ▷ Define Embedded System Goals - System user: description in natural language of the primary system functionalities;
> ▷ Embedded Software Requirements: Programmer - detailed information on software requirements including its relationships, traceability, and model representations.

Work Products:

- **RA.ALD.WP1 - A set of Abstraction Levels:** Each stakeholder involved in the embedded system development has its skills. Therefore, they can be familiar with some representations, not so familiar with others, and they can unknown a set of them. Thus, abstraction levels define restrictive views upon different kinds of representations (models) [RA.ALD.a1, RA.ALD.a2, RA.ALD.a3].

5. RP Release Planning

Release planning covers crucial steps aiming to determine the optimal set of requirements for a certain release to be implemented at a defined/estimated time and cost to achieve some goals. Performing this step carelessly would lead to high risky situations or fail to achieve planned goals. For example, placing important features at a too late release would make the product miss the right moment to gain the customers' impression.

5.1. RP.ESC Embedded System Configuration

Purpose: The goal of this sub-process is to identify the best functional requirements configuration based on a set of non-functional requirements.

Inputs: DS.ESR.WP3 Traceability Plan, DS.ESR.WP2 Software Requirements Documentation

Actions:

- **RP.ESC.a1:** Identify the best functional requirements configuration considering the non-functional requirements. A big challenge in ES domain is to manage conflicting NFRs (e.g., cost and safety). Hence, sophisticated techniques have to be used to find good design solutions using the space of exploration. Space of exploration is a space of possible solution **(Level 3)**.

  Supporting action(s):
  - DS.ESR.a3 Define and maintain a strategy for component reuse requirements
  - DS.ESR.a5 Perform traceability between each functional requirement and the system goals described in RE.DSG sub-process, including its relationships and the way they relate to each other
  - DS.ESR.a7 Elicit and specify a set of non-functional requirements

Work Products:

- **RP.ESC.WP1 - System Configuration:** The best requirements configuration for an embedded system considering a set of non-functional requirements [RP.ESC.a1].

6. DS Documentation and Requirements Specification

Documentation and Requirement specification deal with how a company organizes requirements and other knowledge gathered during requirements engineering process into consistent, accessible and reviewable documents. The software requirements specification (SRS) contains the product's detailed functional and quality requirements.

6.1. DS.ESR Embedded Software Requirements

Purpose: The goal of this sub-process is to specify the software requirements of the embedded system. The software requirements include the capabilities of the embedded software, as well as the non-functional requirements [13, 12].

Inputs: RE.DSG.WP1 System description, RE.SHC.WP2 Software Constraints, RE.IEA.WP1 Variables definition, and RA.ALD.WP1 A set of Abstraction Levels.
Actions:

- **DS.ESR.a1:** Define and maintain a strategy for component reuse requirements. Maintenance and integration issues may cause compatibility problems, mainly with legacy code developed for the original core component model. Hence, a set of procedures for conducting reuse should be defined **(Level 3)**;
- **DS.ESR.a2:** Identify reusable software requirements. The reuse strategy defined must be followed and the requirements should be reused **(Level 3)**;

  Supporting action(s):
  - DS.ESR.a1 Define and maintain a strategy for component reuse requirements.
- **DS.ESR.a3:** Elicit and specify a set of functional software requirements of the system. The requirements should summarize the software functionalities **(Level 1)**;

Examples of software functional requirements [24]:
▷ Patient data validations - (1) The application shall only allow Patient Data to be changed when Infusion is not in progress
▷ Infusion monitoring - (2) when Infusion is in progress, The application shall always maintain the current Infusion status and mode
▷ Power - (3) the application shall indicate the power source of the system - AC or Battery at all times during operation
▷ (3.1) if the system works with AC power suppy then the application shall check supply voltage every X units of time and commands 'Voltate Out of Range' alarm message, if the voltage is not in the desired range;

- **DS.ESR.a4:** Model the functional requirements **(Level 1)**;

  Supporting action(s):
  - OS.PR.a4 Define one or more tools, or an environment, required to support the activities towards the requirements engineering
  - OS.PR.a5 Define a set of implementation technology such as programming languages, code pattern, and modeling languages to support the embedded systems development

Examples of modeling languages for functional requirements:
▷ UML, SysML, MARTE, Use Case diagram, i* (i star) ;
▷ Statechart, Petrinets, SPICE, Z Notation;
▷ Activity diagram, Natural language, SIGNAL language;
▷ Boilerplate notation, Astral language, Duration calculus;
▷ SOFL (structured object-based formal language);
▷ Causal function representation language (CFRL);

- **DS.ESR.a5:** Perform traceability between each functional requirement and the system goals described in RE.DSG sub-process, including its relationships and the way they relate to each other **(Level 2)**.

  Supporting action(s):
  - RE.DSG.a2 Elicit and specify a set of system goals from stakeholder's needs

- **DS.ESR.a6:** Define the software interfaces. Each artifact must have required / provided interfaces, describing its characteristics, consistent design rationale, and configuration. Additionally, the software components to be connected by the interfaces must be described **(Level 1)**;

- **DS.ESR.a7:** Elicit and specify a set of non-functional requirements. **(Level 2)**;

▷ The NFR4ES (Non-Functional Requirements for Embedded Systems) catalog can be used to support the elicitation and specification. The requirements can also be described in a specification card [27].

Examples of non-functional requirements for embedded systems:
▷ Performance, security, reliability, safety;
▷ Development cost, real-time requirements;
▷ Usability, robustness, survivability;
▷ User interface, resource consumption, and others;

Examples of non-functional requirements description (not related to the pump) [15]:
▷ Response time to a button press shall be lass than 200 msec (performance);
▷ The software shall support internationalization for at least four languages including: English, German, Chinese, and Portuguese (usability);
▷ Software access shall be password protected (security);
▷ The system shall check code integrity with a secure hash function at startup and shall halt if code has been altered (integrity);

- **DS.ESR.a8:** Model the non-functional requirements **(Level 3)**;

  Supporting action(s):
  - DS.ESR.a7 Elicit and specify a set of non-functional requirements
  - OS.PR.a4 Define one or more tools, or an environment, required to support the activities towards the requirements engineering
  - OS.PR.a5 Define a set of implementation technology such as programming languages, code pattern, and modeling languages to support the embedded systems development

Examples of modeling languages for non-functional requirements:
▷ Natural language, SysML, MARTE;
▷ Temporal logic, Duration calculus, NFR Framework;

- **DS.ESR.a9:** Plan software connection to integrate different software components. The software interfaces should be used to allow the connection between the software components **(Level 2)**;

Work Products:

- **DS.ESR.WP1 - Reuse-based Process:** A process with a set of activities describing how software requirements reuse is carried out by the company [DS.ESR.a1, DS.ESR.a2].

- **DS.ESR.WP2 - Software Requirements Documentation:** The software requirements document describes the capabilities of the embedded software, as well as the interfaces and constraints under which the software has to perform, and the modeling languages that can be used to model the requirements [DS.ESR.a3, DS.ESR.a4, DS.ESR.a7, DS.ESR.a8];

- **DS.ESR.WP3 - Traceability Plan:** It is a section of the software requirements specification document that contains information about the traceability between functional requirements and the system goals [DS.ESR.a5];

- **DS.ESR.WP4 - Software Interfaces Documentation:** The software interfaces and a document containing guidelines, features, and information about them, i.e., how to use, and how to integrate. It can be a section of the software requirements documentation [DS.ESR.a6];

- **DS.ESR.WP5 - Software Connection Plan:** The software interfaces and a document that describes the connection process that aims to use the software interfaces to connect two different software components. It can be a section of the software requirements documentation [DS.ESR.a9];

6.2. DS.EHR Embedded Hardware Requirements

Purpose: The goal of this sub-process is to identify and specify a set of hardware devices to be used in the development of an embedded system. The focus is on functionalities, standards, actions required for the system to work properly, and electrical and mechanical requirements. In this sub-process, the action DS.EHR.a7 was exclusively provided for the choice of the microcontroller, since it is the embedded computer in embedded system and therefore, deserves a particular consideration [28]. It is important to highlight the concern with the result of the Software and Hardware Constraints (RE.SHC) sub-process, since it may be used as a guideline to support the execution of this sub-process.

Inputs: RE.DSG.WP1 System description, RE.SHC.WP1 Hardware Constraints, RE.IEA.WP1 Variables definition, and RA.ALD.WP1 A set of Abstraction Levels. **Actions:**

- **DS.EHR.a1:** Specify and document a set of hardware architecture standards for the domain of the embedded system that will be developed **(Level 1)**;

  Example:
  ▷ Generic infusion pump user interface (GIP-UI) architecture [22];

- **DS.EHR.a2:** Elicit and specify a set of non-functional requirements that the hardware devices must fulfill **(Level 1)**;

  ▷ The NFR4ES (Non-Functional Requirements for Embedded Systems) catalog can be used to support the elicitation and specification. The requirements can also be described in a specification card [27].

Examples of non-functional requirements:
▷ Dimensions, weight, shape, durability, ease of transport, non-flammable, temperature, and waterproof;
Examples of non-functional electrical requirements that must be upheld by the design [18]:
▷ Leakage current;
▷ Component failure;
▷ Electromagnetically compatible;
▷ Electrostatic discharge;
▷ Filter power interference.

Examples of non-functional electrical requirements description [22]:
▷ The pump shall be immune to 25 kV air discharge (minimum) when tested according to IEC 61000-4-2;
▷ The pump shall be immune to 20 V/m radiated RF, minimum;
Others examples not related to the pump:
▷ The external case temperature shall not exceed 110 degrees F;
▷ The hardware device should remain operable even at temperatures above 122 degrees F.

- **DS.EHR.a3:** Model the non-functional requirements **(Level 2)**;

  Supporting action(s):
  - DS.EHR.a2: Elicit and specify a set of non-functional requirements that the hardware devices must fulfill
  - OS.PR.a4 Define one or more tools, or an environment, required to support the activities towards the requirements engineering
  - OS.PR.a5 Define a set of implementation technology such as programming languages, code pattern, and modeling languages to support the embedded systems development

  Examples of modeling languages for non-functional requirements:
  ▷ Natural language, SysML, MARTE;
  ▷ Temporal logic, Duration calculus, NFR Framework;

- **DS.EHR.a4:** Provide an overview of the hardware components to be used in the development of an embedded system **(Level 1)**.

Examples of an overview of the hardware components
(retrieved from [18]):
▷ The pump must have a power supply that converts alternating current energy into direct current that powers the electronic components. The communication between system and user is done via Control Panel. It combines a touch panel with a speaker by which a clinician can enter and confirm configuration and see and hear alarms and warnings;

- **DS.EHR.a5:** Elicit the mechanical requirements (e.g., dimensions and materials) for each identified hardware component **(Level 1)**;

Examples of mechanical requirements for memory and USB port
[5, 6]:
▷ The width of the plated input/output contact measured at the lateral midpoint of the contact: min 0.95mm, max 1.05mm;
▷ The total number of contacts: 168;
▷ Substrate Material: 0.30 + 0.05 mm minimum half-hard phosphor bronze or other high strength copper based material;

- **DS.EHR.a6:** Elicit electrical requirements for each identified hardware component. This include information such as maximum ratings, thermal, electromagnetic interference, and electromagnetic static discharge characteristics, and voltage regulator controller **(Level 1)**;

Examples of electrical requirements for microcontrollers [23]:
▷ Electromagnetic static discharge for human body model: 2000 V;
▷ Input / Output supply voltage: min -0.3 V, max 4.6 V;
▷ Flash read voltage: symbol VFlash, min 3.0, max 3.6 V;

- **DS.EHR.a7:** Identify and document the microcontroller of the embedded system based on the results of previous actions. Thus, its functionalities and relevant design are registered, including its architecture (Harvard or von Neumann), processor, amount of memory (RAM or ROM), number or types of Input/Output pins on an a microcontroller, language (Assembler, C), power consumption and constraints and development support [26] **(Level 2)**;

Examples of questions to keep in mind when choosing a microcontroller
[26]:
▷ What hardware peripherals are required?
▷ Are external communications needed?
▷ What architecture should be used?
▷ What sort of community and resources are available for the microcontroller?
▷ What is the market availability of the microcontroller?

Examples of requirements when choosing a microcontroller capable to execute software peripherals at the same time with the main application [20]:
▷ High performance core;
▷ Fast interrupt response;
▷ Set of hardware functions;
▷ Programmable input/output pins;
▷ Sufficient amount of memory.

- **DS.EHR.a8:** Specify and document a set of additional hardware devices based on the results of previous actions. Thus its functionalities and appropriate design are registered. This action is not related to hardware devices such as input and output user interface, sensors and actuators, external interfaces, and hardware adapter. These hardware components have their actions **(Level 1)**.

Examples of hardware devices:
▷ Memory RAM, Memory ROM, Storage Device;
▷ Power Supply, Digital Signal Processor;

- **DS.EHR.a9:** Perform hardware analysis. The goal of this analysis is to select the hardware devices based on the business goals correctly, and the quality attributes, also known as non-functional requirements, to be achieved by the embedded system **(Level 2)**;

- **DS.EHR.a10:** Elicit a set of manufacture requirements. They are important to minimize the production cost since it depends on the hardware components of the product. Questionnaires can be used to help to improve the design for manufacture. The questions of the questionnaire depend on the manufacturing technology for a particular product [10, 9] **(Level 2)**.

Examples of questions for laser processing [9]:
▷ Is the construction possible to be laser processed from one direction or at least in one plane?;
▷ Is the possibility to use various material combinations considered?;
▷ Are the possibilities to use different laser processing methods for the same construction or multi-processing methods considered?

- **DS.EHR.a11:** Perform traceability between the functionalities of each hardware device and the definition of behavioral requirements described in sub-process DS.DBR **(Level 3)**.

  Supporting action(s):
  - DS.DBR.a1 Provide a set of software behaviors to document the actions the software should perform

- DS.DBR.a2 Provide a set of hardware behaviors to document the actions the hardware should perform

> Example [17, 18]:
> ▷ Hardware device: Control Panel;
> ▷ Associated behavioral requirements: see DS.DBR.a1.

Work Products:

- **DS.EHR.WP1 - Hardware Architecture Standards:** The hardware architecture standards and a document containing guidelines, features, and information about them, i.e. how to use and how to integrate it [DS.EHR.a1];
- **DS.EHR.WP2 - Hardware Components Overview:** It is a document that provides an overview of the hardware components and serve as a guideline for the DS.EHR.a7 to support the specification of additional hardware devices [DS.EHR.a4];
- **DS.EHR.WP3 - Microcontroler Documentation:** The microcontroller of the embedded system and its documentation containing guidelines, features, and information about them, i.e. how to use and how to integrate [DS.EHR.a7];
- **DS.EHR.WP4 - Electrical, Mechanical, and Manufacturing Requirements:** A document containing the electrical and mechanical requirements of the hardware components, and the manufacturing requirements of the embedded system product [DS.EHR.a5, DS.EHR.a6, DS.EHR.a10];
- **DS.EHR.WP5 - Hardware Requirements Documentation:** The hardware requirements document describes the capabilities of the hardware devices, as well as the interfaces and constraints under which the devices has to perform. Additionally, information about them such as how to use and how to integrate are included [DS.EHR.a2, DS.EHR.a3, DS.EHR.a8, DS.EHR.a9].
- **DS.EHR.WP6 - Traceability Plan:** It is a section of the hardware requirements specification document that contains information about the traceability between the functionalities of the hardware devices and the behavioral requirements [DS.EHR.a11];

6.3. DS.DBR Definition of Behavioral Requirements for Embedded Systems

Purpose: Desired behavior and system behavior relies on customer wishes to the final embedded product. In this way, customer expectations should be managed as well as the undesired behavior, the system reactions that should be avoided. Thus, it is possible to find different ways to treat them. Hence, the goal of this sub-process is to identify and specify the behavior of an embedded system and the services offered by it. It involves describing and modeling these behaviors within the system scope.

Examples of state-oriented behavioral models of the DS.DBR sub-process:
▷ Finite state machines;
▷ Statecharts;
▷ Petri nets;

Inputs: RE.DSG.WP1 System Description, DS.ESR.WP2 Software Requirements Documentation, DS.EHR.WP5 Hardware Requirements Documentation, and DS.DEH.WP6 Actuators Documentation.

Actions:

- **DS.DBR.a1:** Provide a set of software behaviors to document the actions the software should perform **(Level 1)**;

Examples of software behaviors:
▷ After the start button has been pushed, a timer counter shall be displayed;
▷ When the infusion is in progress, a boolean signal shall be displayed;
▷ In case of errors, a message describing the errors shall be displayed to the user.

- **DS.DBR.a2:** Provide a set of hardware behaviors to document the actions the hardware should perform **(Level 1)**;

Examples of Hardware Behaviors (HB) [29, 18]:
▷ HB1. After the dose button has been pushed, two beeps shall be sounded and the pump
will begin delivering the demand dose;
▷ HB2. An occlusion alarm shall be triggered if the pump senses an upstream (insulin supply side) occlusion;
▷ HB3. the mechanical pump shall halt pumping when commanded,
or caused in response to an alarm condition
▷ HB4. After the button A has been pushed, a red light shall be lit;
▷ HB5. After the start button has been pushed, the motor shall be turned on.

- **DS.DBR.a3:** For each behavior, perform an analysis to identify the undesired ones and describe what should be done to avoid them **(Level 3)**;

Example of undesired behaviors:
▷ The system continues operating even after the user presses the stop button in the monitor;
▷ Action: the system should provide a manual way to stop the system operation.

- **DS.DBR.a4:** Perform an analysis to identify hardware actions that trigger software actions and software actions that trigger hardware actions. These actions must be traceable **(Level 3)**;

Examples:
▷ During the demand dose delivering (hardware action), the main screen should show "RUNNING" (software action);
▷ In a safety-critical system, when a stop button is pressed in order to stop the process (hardware action), a message is displayed asking the user if he really wants to stop it (software action). If so, then the process will be stopped (hardware action);

- **DS.DBR.a5:** Perform modeling to visually document the software and hardware behaviors of the system **(Level 2)**.

  Supporting action(s):
  - OS.PR.a4 Define one or more tools, or an environment, required to support the activities towards the requirements engineering
  - OS.PR.a5 Define a set of implementation technology such as programming languages, code pattern, and modeling languages to support the embedded systems development

Work Products:

- **DS.DBR.WP1 - Behavioral Requirements:** A document describing the software and hardware behaviors. The software and hardware behaviors will be used to derive software and hardware requirements [DS.DBR.a1, DS.DBR.a2];
- **DS.DBR.WP2 - Action Plan:** An action plan containing the undesired behaviors of the system and a set of guidelines providing actions to be taken to avoid them [DS.DBR.a3, DS.DBR.a4];
- **DS.DBR.WP3 - Behavioral Models:** A diagram with a representation of all software and hardware behaviors of the domain [DS.DBR.a5].

6.4. DS.DEH Define Embedded Hardware Devices

Purposes: (1) To define a set of interfaces required by the system. The focus is on functionalities, standards, and interfaces constructed to provide human interaction with the system. These interactions can occur using input and output user interface. Depending on the domain of the embedded system and the kind of interface, there may be a set of interface standards that should be considered.

(2) To define a set of sensors and actuators to be used in the system development. The focus is on functionalities, standards, and actions for the system to work properly. The actions include the monitoring of control and environmental variables and the software and hardware behavior. It is assumed that the behavioral requirements and the controlled and monitored variables have been previously specified.

(3) To define the external interfaces and the hardware adapters that will be used in the system. The focus is on functionalities and standards. The external interfaces to be chosen depend on the type of communication and the connection interface of the integrated circuit board. Similarly, a hardware adapter must be carefully specified to set the hardware connectors that will connect one hardware to another.

Inputs: RE.DSG.WP1 System description, RE.SHC.WP1 Hardware Constraints, RE.IEA.WP1 Variables definition, and RA.ALD.WP1 A set of Abstraction Levels.
Actions:

- **DS.DEH.a1:** Define and document a set of interface standards **(Level 2)**;
- **DS.DEH.a2:** Define and document a set of input user interface. Thus, its characteristics and relevant information such as signal (analog and/or digital), communication with the microcontroller, and microcontroller pin **(Level 1)**;

Example of input user interface [18]:
▷ Bolus request button - patients press a button to request a drug bolus in addition to a constant basal rate;
▷ Scanner - the scanner reads information from patient wristbands, clinician badges, and drug labels;
▷ Control panel - a control panel allows the pump to be started and stopped. It allows a clinician to command delivery of a bolus. It also allows a clinician to specify the duration a prescribed volume-to-be-infused is delivered.

- **DS.DEH.a3:** Define and document a set of output user interface. Thus, its characteristics and relevant information such as signal (analog and/or digital), communication with the microcontroller, and microcontroller pin **(Level 1)**;

Example of output user interface [18]:
▷ Control panel - the control panel must display currently-programmed patient data and physician's prescription;
▷ Alarm audible - alarms shall cause audible alarms signals. Each tone in the alarm melody shall be composed of a minimum of 4 harmonic components in the range 300 Hz to 4000 Hz comprising an inverted 9th jazz chord. (IEC 60601-1-8).

- **DS.DEH.a4:** Identify and document a set of sensors and actuators standards **(Level 2)**;
- **DS.DEH.a5:** Identify and document a set of sensors. Thus its functionalities and relevant design rationale are registered, including the required / provided interfaces. The documentation must also contain a tutorial on how the sensors should be used **(Level 1)**;

Example of sensors [29, 18, 7]:
▷ The pump shall have an occlusion sensor - The occlusion sensor shall trigger an occlusion alarm whenever the actual flow rate is less than the programmed rate by at least x% for y seconds due to occlusion;
▷ Environmental sensor - This component senses pump operation parameters like temperature, pressure and humidity and sounds alarms if they lie outside the operational range of the device.
▷ The PCA pump shall detect air-in-line embolism (bubble);
Others examples of sensors that are not related to the pump:
▷ Physical position sensor.

- **DS.DEH.a6:** For each sensor, perform a linkage between them and the variables defined in RE.IEA sub-process indicating the ones that the sensor should monitors. If there is no variable, the sub-process (RE.IEA) may be performed again and the work product RE.IEA.WP1 updated **(Level 1)**.

  Supporting action(s):
  - RE.IEA.a1 Establish and document the monitored and controlled variables
  - RE.IEA.a2 Define the type, range, and units required for all variables
  - RE.IEA.a4 Define a status attribute (invalid/valid) for each monitored variable

Example [18, 29]:
▷ Sensor: occlusion sensor;
▷ Monitored variable: basal flow rate;
▷ Sensor: environmental sensor;
▷ Monitored variables: temperature, pressure, and humidity;

- **DS.DEH.a7:** The sensors must send to the software the values of the variables they monitor in a predefined time interval. This information should be planned and documented **(Level 2)**;

  Supporting action(s):
  - RE.IEA.a1 Establish and document the monitored and controlled variables
  - RE.IEA.a2 Define the type, range, and units required for all variables
  - RE.IEA.a4 Define a status attribute (invalid/valid) for each monitored variable

Example [18, 29]:
▷ Sensor: occlusion sensor;
▷ Monitored variable: basal flow rate;
▷ Time interval: during infusion.

- **DS.DEH.a8:** Identify and document a set of actuators. Thus its functionalities and relevant design rationale are registered, including the required / provided interfaces. The documentation must also contain a tutorial on how the actuators should be used **(Level 1)**;

Example of actuators [18]:
▷ Mechanical pump;
Examples that are not related to the pump:
▷ Motors;
▷ Software component;

- **DS.DEH.a9:** For each actuator, perform a linkage between them and the software and/or hardware behavior defined in DS.DBR sub-process indicating the behaviors that the actuator should trigger. If there is no behavior, the (DS.DBR) sub-process may be performed again and the work product DS.DBR.WP1 updated **(Level 2)**.

  Supporting action(s):
  - DS.DBR.a1 Provide a set of software behaviors to document the actions the software should perform
  - DS.DBR.a2 Provide a set of hardware behaviors to document the actions the hardware should perform

Example:
▷ Mechanical pump: DS.DBR.a2 - HB3;

- **DS.DEH.a10:** Define and document a set of external interfaces and hardware adapters standards **(Level 2)**;

- **DS.DEH.a11:** Elicit and document the external interfaces. Thus its characteristics and relevant design rationale are registered, including its interface type (e.g., cable, optical fiber, physical device), physical configuration (size, shape), and data/control flow **(Level 1)**;

Examples of external interfaces [18, 17]:
▷ Integrated Clinical Environment (ICE) interface: the ICE interface allows the PCA pump to be monitored and controlled remotely, either by a clinician using an ICE supervisor user interface or an ICE app;
▷ ICE network interface;
Others examples that are not related to the pump:
▷ Serial Peripheral Interface;
▷ Serial, USB, and CAN;

- **DS.DEH.a12:** Elict and document the hardware adapters. Thus its characteristics and relevant design rationale are registered, including its physical interface (input and output) and physical configuration, and the devices that each hardware adapter will connect **(Level 1)**;

Example of hardware adapter [18]:
▷ Cable - the bolus request button is connected by a cable to the PCA pump, and may have a clip to attach to patient's bedding.

- **DS.DEH.a13:** Plan and perform hardware connection. The hardware devices that will be connected must be identified, and a hardware adapter should be used to allow the connection between them **(Level 1)**.

Examples [18]:
▷ Hardware device: request button;
▷ Hardware device: PCA pump;
▷ Hardware adapter: cable

- **DS.DEH.a14:** Identify the resources that are shared by several components. This information is usefull to detect and avoid collisions between sharing components in the system **(Level 1)**.

  Supporting action(s):
  - DS.EHR.a4 Provide an overview of the hardware components to be used in the development of an embedded system.
  - DS.ESR.a9 Plan software connection to integrate different software components. The software interfaces should be used to allow the connection between the software components.
  - DS.DEH.a5 Identify and document a set of sensors. Thus, its functionalities and relevant design rationale are registered, including the required / provided interfaces. The documentation must also contain a tutorial on how the sensors should be used.
  - DS.DEH.a8 Identify and document a set of actuators. Thus, its functionalities and relevant design rationale are registered, including the required / provided interfaces. The documentation must also contain a tutorial on how the actuators should be used.

Example:
▷ Electric motor and memory;

Work Products:

- **DS.DEH.WP1 - Interface Standards:** The interfaces standards and a document containing guidelines, features, and information about them, i.e., how to use and how to integrate it [DS.DEH.a1];

- **DS.DEH.WP2 - Input User Interface:** The input user interfaces and a document containing guidelines, features, and information about them, i.e., how to use and how to integrate it [DS.DEH.a2];

- **DS.DEH.WP3 - Output User Interface:** The output user interfaces and a document containing guidelines, features, and information about them, i.e., how to use and how to integrate it [DS.DEH.a3];
- **DS.DEH.WP4 - Sensors and Actuators Standards:** The sensors and actuators standards and a document containing guidelines, features, and information about them, i.e., how to use and how to integrate it [DS.DEH.a4];
- **DS.DEH.WP5 - Sensors Documentation:** The sensors that will be part of the system and its documentation containing guidelines, features, and information about them, i.e., how to use and how to integrate it [DS.DEH.a5, DS.DEH.a6, DS.DEH.a7];
- **DS.DEH.WP6 - Actuators Documentation:** The Actuators that will be part of the system and a document that contains guidelines, features, and information about them, i.e., how to use and how to integrate it [DS.DEH.a8, DS.DEH.a9];
- **DS.DEH.WP7 - External Interfaces and Hardware Adapters Standards:** The external interfaces, hardware adapters, and a document containing guidelines, features, and information about them, i.e., how to use and how to integrate it [DS.DEH.a10, DS.DEH.a11, DS.DEH.a12];
- **DS.DEH.WP8 - Hardware Connection Plan:** The hardware adapters and a document that describes the connection process that aims to use the hardware adapter to connect two different hardware devices [DS.DEH.a13].
- **DS.DEH.WP9 - Resources Traceability:** A document with information about shared resources. This document should contain the resources and links to the components that make use of them [DS.DEH.a14].

7. RV Requirements Validation

This process involves checking the documented requirements against defined quality standards and the real needs of various stakeholders. It ensures that the documented requirements are complete, correct, consistent, and unambiguous.

7.1. RV.IHS Integration Between Hardware and Software

Purpose: This main process area aims to support the system integration. It verifies the interaction among software components and the communication interfaces between the software and hardware [25]. This is the most crucial step in embedded system design, which the coded software meets the designed hardware for the first time. It consists of verifying whether the embedded software runs correctly on the hardware design before the design is committed for fabrication. This process is called HW/SW Co-Verification [14].

Inputs: DS.ESR.WP2 Software Requirements Documentation, DS.EHR.WP5 Hardware Requirements Documentation, and DS.DEH.WP6 Actuators Documentation.
Actions:

- **RV.IHS.a1 - Plan and perform an HW/SW Co-Verification. (Level 3)** The coded software and the designed hardware must be integrated and verified, to detect integration errors. The plan must include the co-verification method and a tool that should provide control and visibility for both software and hardware engineers;

  Supporting action(s):
  - OS.PR.a4 Define one or more tools, or an environment, required to support the activities towards the requirements engineering
  - OS.PR.a5 Define a set of implementation technology such as programming languages, code pattern, and modeling languages to support the embedded systems development

Examples of Co-Verification methods [14]:
▷ Host-Code Mode with Logic Simulation;
▷ Instruction Set Simulation with Logic Simulation;
▷ Evaluation Board with Logic Simulation;
▷ C Simulation;

Work Products:

- **RV.IHS.WP1 - HW/SW Co-Verification Plan:** A document describing the integration process that decides if the hardware and software requirements that were implemented are working correctly.

[RV.IHS.a1]

A. Conformity to Uni-REPM actions

Table A.1 presents the embedded systems actions identical to those specified in Uni-REPM. However, we refined these actions to address the particularities of embedded systems.

Table A.1.: Actions of embedded system module identical to Uni-REPM actions

ES Module Action	**Uni-REPM Action**
OS.PR.a1	RE.SI.a1
OS.PR.a2	RE.SI.a2
EL.ER.a1	RE.DC.a3
EL.ER.a6	RE.DC.a1
EL.ER.a7	RE.DC.a2
EL.ER.a8	RE.DC.a4
EL.ER.a9	RE.DC.a5
RA.SA.a1	RA.GA.a1

Bibliography

[1] Raian Ali, Fabiano Dalpiaz, and Paolo Giorgini. A goal-based framework for contextual requirements modeling and analysis. *Requirements Engineering*, 15(4):439–458, 2010.

[2] Peter Braun and Martin Rappl. Abstraction levels of embedded systems. *OMER 2 Workshop*, 2011.

[3] James F Chang. *Business process management systems: strategy and implementation*. CRC Press, 2016.

[4] Yihai Chen, Mark Lawford, Hao Wang, and Alan Wassyng. Insulin pump software certification. In *International Symposium on Foundations of Health Informatics Engineering and Systems*, pages 87–106. Springer, 2013.

[5] Intel Corporation. Pc sdram unbuffered dimm specification - revision 1.0. Technical report, Intel Corporation, 1997.

[6] Compaq Corporation Computer. Universal serial bus specification. Technical report, Compaq Computer Corp., Hewlett-Packard Co., Intel Corp., Lucent Technologies Inc., Microsoft Corp., NEC Corp., Koninklijke Philips Electronics NV, 2000.

[7] Penn Engineering. The generic patient controlled analgesia pump model. http://rtg.cis.upenn.edu/gip-UPenn/gip-docs/GPCA%20Pump%20Model.doc, 2018. Accessed: 2018-05-29.

[8] Vina Ermagan, Jun-ichi Mizutani, Kentaro Oguchi, and David Weir. Towards model-based failure-management for automotive software. In *Proceedings of the 4th International Workshop on Software Engineering for Automotive Systems*, page 8. IEEE Computer Society, 2007.

[9] Harri Eskelinen, Janne-Matti Heinola, Pertti Silventoinen, et al. Dfm (a)-aspects for a microwave waveguide ring resonator design. *Research report/Lappeenranta University of Technology, Department of Mechanical Engineering*, 2004.

[10] Alberto Ferrari and Alberto Sangiovanni-Vincentelli. System design: Traditional concepts and new paradigms. In *Computer Design, 1999.(ICCD'99) International Conference on*, pages 2–12. IEEE, 1999.

[11] John Hatlicff. Lecture pca pump: Introduction. http://www.santoslab.org/pub/high-assurance/module-pca-pump/slides/PCA-Pump--Intro.pdf, 2016. Accessed: 2018-06-09.

[12] ISO/IEC. Iso/iec standard for systems and software engineering - software life cycle processes. *IEEE Std 12207-2008*, pages c1–138, Jan 2008.

[13] ISO/IEC. Iso/iec international standard - systems and software engineering – life cycle processes –requirements engineering. *ISO/IEC/IEEE 29148:2011(E)*, pages 1–94, Dec 2011.

[14] Ganssle Jack, Noergaard Tammy, Eady Fred, J. Katz David, Gentile Rick, Arnold Ken, Hyder Kamal, and Bob Perrin. *Embedded Software: Know It All (Newnes Know It All)*. Newnes, 2007.

[15] Philip Koopman. *Better embedded system software*. Drumnadrochit Education, 2010.

[16] Philip Koopman. Dependability for embedded systems, 2015. Lectures.

[17] Brian R Larson, John Hatcliff, and Patrice Chalin. Open source patient-controlled analgesic pump requirements documentation. In *Software Engineering in Health Care (SEHC), 2013 5th International Workshop on*, pages 28–34. IEEE, 2013.

[18] Brian R. Larson and John Hatlicff. Open patient-controlled analgesia infusion pump system requirement. http://santoslab.org/pub/open-pca-pump/artifacts/Open-PCA-Pump-Requirements.pdf, 2018. Accessed: 2018-05-28.

[19] David L Lempia and Steven P Miller. Requirements engineering management handbook, report. *Disponible sur http://www. faa. gov/aircraft/air_cert/design_-approvals/air_software/media/AR-08-32. pdf*, 2009.

[20] D Lioupis, A Papagiannis, D Psihogiou, and M Stefanidakis. Software peripherals-requirements and constraints for real-time embedded systems. In *In IEEE Real-Time Embedded Systems Workshop*. Citeseer, 2001.

[21] L. E. G. Martins and T. de Oliveira. A case study using a protocol to derive safety functional requirements from fault tree analysis. In *2014 IEEE 22nd International Requirements Engineering Conference (RE)*, pages 412–419, Aug 2014.

[22] Paolo Masci, Yi Zhang, Paul Jones, Harold Thimbleby, and Paul Curzon. A generic user interface architecture for analyzing use hazards in infusion pump software. In *OASIcs-OpenAccess Series in Informatics*, volume 36. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.

[23] Semiconductors NXP. Mpc5554 microcontroller data sheet. Technical report, NXP Semiconductors, 2012.

[24] University of Minessota. Infusion pump requirements. http://greggay.com/courses/fall15csce740/Documents/InfusionRequirements.pdf, 2012. Accessed: 2018-05-29.

[25] Kim H Pries and Jon M Quigley. *Project Management of Complex and Embedded Systems: Ensuring Product Integrity and Program Quality.* CRC Press, 2008.

[26] Margaret Rouse. microcontroller. http://internetofthingsagenda.techtarget.com/definition/microcontroller, 2017. Accessed: 2017-07-22.

[27] Reinaldo Antônio da SILVA. Nfr4es: um catálogo de requisitos não-funcionais para sistemas embarcados. Master's thesis, Universidade Federal de Pernambuco, 2019.

[28] Tim Wilmshurst. *Designing embedded systems with PIC microcontrollers: principles and applications.* Newnes, 2006.

[29] Yi Zhang, Raoul Jetley, Paul L Jones, and Arnab Ray. Generic safety requirements for developing safe insulin pump software. *Journal of diabetes science and technology*, 5(6):1403–1419, 2011.

[30] Yi Zhang, Paul L Jones, and Raoul Jetley. A hazard analysis for a generic insulin infusion pump. *Journal of diabetes science and technology*, 4(2):263–283, 2010.