Requirement Construction using Templates

Requirements templates provide a simple and easily understandable approach to reduce language effects when documenting requirements. Templates support the author in achieving high quality and syntactic unambiguousness in optimal time and at low costs.

The following is a step-by-step description of the correct application of requirements templates.

Step 1: Determine the Legal Obligation

In the beginning, you should determine the degree of legal obligation for a requirement. Usually, one distinguishes between legally obligatory requirements, urgently recommended requirements, future requirements, and desirable requirements. To achieve this within a requirement, you can use the modal verbs *shall*, *should*, *will*, and *may*. Alternatively, the legal obligation of a requirement can be documented by a specific requirements attribute.
Read about MoSCoW https://en.wikipedia.org/wiki/MoSCoW_method.

Step 2: The Requirement Core

The core of each requirement is the functionality that it specifies (e.g., print, save, paste, or calculate). This functionality is referred to as the *process*. Processes are activities and may only be described using verbs. The process that depicts the system behavior by means of a requirement is to be described in step 2.

Step 3: Characterize the Activity of a System

For functional requirements, the system activity can be classified as one of three relevant types:

- *Autonomous system activity:* The system performs the process autonomously.
- *User interaction:* The system provides the process as a service for the user.
- *Interface requirement:* The system performs a process depending on a third party (e.g., another system). The system is passive and waits for an external event.

In step 3, any kind of system activity that is specified by a requirement of the system is documented using exactly one of three requirements templates. These requirements templates are described in more detail in the following sections.
After performing steps 1 through 3, the structure of the requirement has been developed (see figure 1). The words that are written in angle brackets must be replaced accordingly.

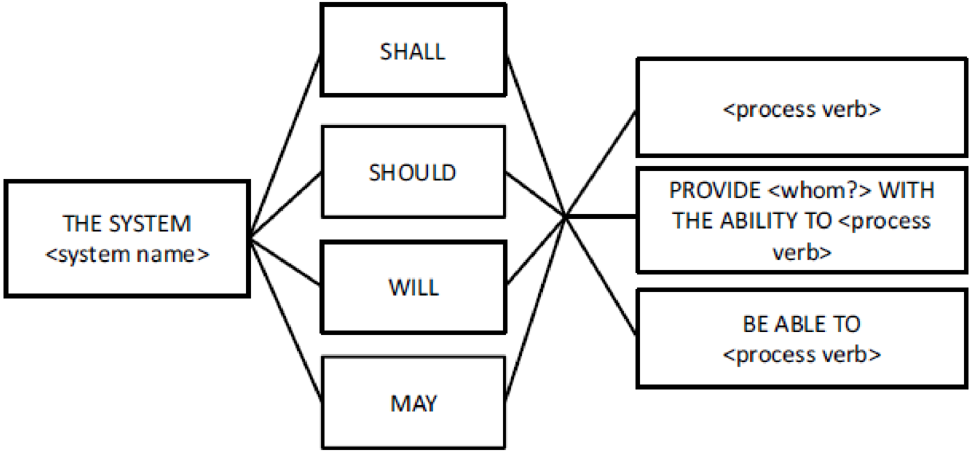
THE SYSTEM <system name>
SHALL
SHOULD
WILL
MAY
<process verb>
PROVIDE <whom?> WITH THE ABILITY TO <process verb>
BE ABLE TO <process verb>

Figure 1 *The core of a requirement and its legal obligation*

Type 1: Autonomous system activity

The first template type is used when requirements are constructed that depict system activities that are performed autonomously. The user does not interact with the activity. We define the following requirements template:

THE SYSTEM SHALL/SHOULD/WILL/MAY <process verb>

<Process verb> depicts a process verb as described in step 2, e.g., print for print functionality or calculate for some calculation that is performed by the system.

Type 2: User interaction

If the system provides a functionality to a user (for example, by means of an input interface), or the system directly interacts with a user, requirements are constructed using template type 2:

THE SYSTEM SHALL/SHOULD/WILL/MAY provide <whom?> with the ability to <process verb>

The user that interacts with the system is integrated into the requirement through <whom?>.

Type 3: Interface requirement

If the system performs an activity and is dependent on neighboring systems, the third template type is to be used. Whenever messages or data are received from a neighboring system, the system must react by executing specific behavior. The following template has proven itself as well suited:

THE SYSTEM SHALL/SHOULD/WILL/MAY be able to <process verb>

Step 4: Insert Objects

Some process verbs require one or more additional objects to be considered complete. In step 4, potentially missing objects and supplements of objects (adverbials) are identified and added to the requirement. For instance, the requirements template for the process verb print is amended by the information of what is being printed and where it is printed. The amendment can be seen in figure 2.

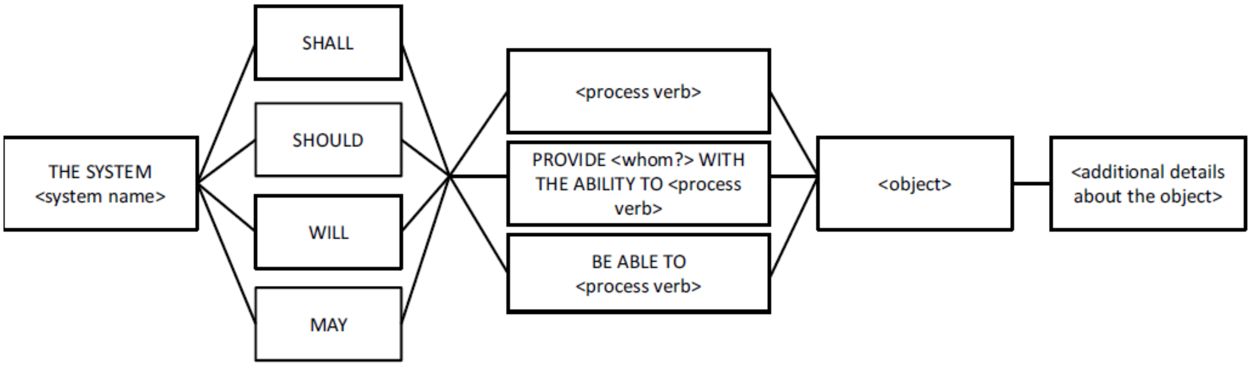
THE SYSTEM
<system name>
SHALL
SHOULD
WILL
MAY
<process verb>
PROVIDE <whom?> WITH THE ABILITY TO <process verb>
BE ABLE TO <process verb>
<object>
<additional details about the object>

Figure 2 *Principle of a complete requirements template without conditions*

Step 5: Determine Logical and Temporal Conditions

Add conditions. Typically, requirements do not document continuous functionalities, but functionalities that are performed or provided only under certain logical or temporal constraints. In order to easily differentiate between logical and temporal conditions, we choose the temporal conjunction *as soon as* for temporal conditions and the conditional conjunction *if* for logical conditions. The conjunction *when* makes not clear whether a temporal or a logical condition is described and should therefore be avoided. In step 5, quality requirements that describe the conditions under which a requirement is fulfilled are added to the beginning of a requirement as a subordinate clause.

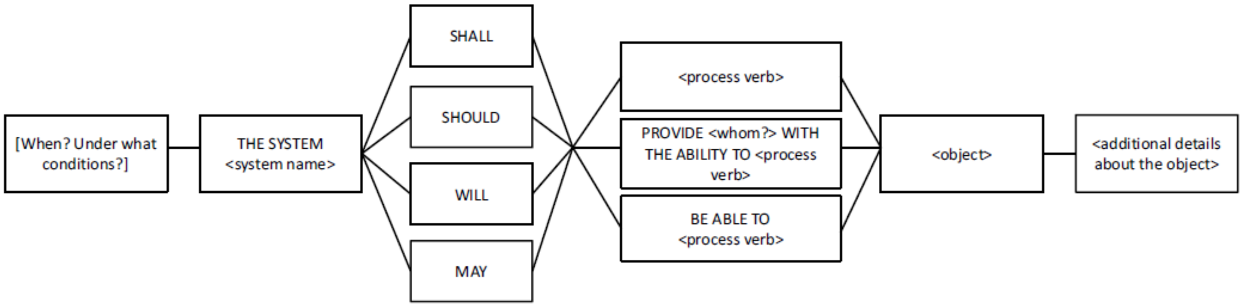
[When? Under what conditions?]
THE SYSTEM <system name>
SHALL
SHOULD
WILL
MAY
<process verb>
PROVIDE <whom?> WITH THE ABILITY TO <process verb>
BE ABLE TO <process verb>
<object>
<additional details about the object>

Figure 3 *The complete requirements template with conditions*

Requirements templates should be used when project members show interest in a formal development process. Style and creativity are harshly limited when requirements templates are used.