

Seven Sins of Specifiers



Hi, what do you know about seven sins of specefiers from your memory?



The "seven sins of specifiers" are issues or pitfalls that should be avoided when specifying requirements. Based on the context you've provided, here are the common ones:

1. **Ambiguity** – Requirements that are unclear or open to multiple interpretations.
2. **Over-Specification** – Providing unnecessary detail or specifying the solution instead of the need, which can limit creativity.
3. **Under-Specification** – Failing to provide sufficient detail, leaving critical aspects of the requirement undefined.
4. **Contradiction** – Conflicting requirements that cannot both be true, leading to confusion and errors.
5. **Unrealistic Requirements** – Defining requirements that are impractical or impossible to implement, often due to technological or budgetary constraints.
6. **Omission** – Leaving out important aspects or edge cases that should be covered in the requirements.
7. **Forward Reference** – Referring to concepts or details that have not yet been defined or introduced in the requirements, leading to confusion.

These sins are typically considered when ensuring that requirements are clear, complete, and feasible.



tell me the correct names




Here are the correct names of the "seven sins of specifiers" based on standard terminology:

1. **Ambiguity** – Unclear language that allows multiple interpretations.
2. **Over-Specification** – Defining the solution instead of the need, which limits flexibility.
3. **Noise** – Including irrelevant or redundant information that does not add value to the requirement.
4. **Contradiction** – Conflicting requirements that cannot both be true or achieved at the same time.
5. **Omission** – Failing to specify important details that are necessary for completeness.
6. **Unfeasibility** – Setting unrealistic expectations that cannot be achieved due to technical or resource limitations.
7. **Unverifiability** – Defining requirements that cannot be tested or measured to confirm they were met.

These names focus on common mistakes that can lead to incomplete, unclear, or problematic requirements.



 Uploaded image
here, remember them



 **Memory updated**

Got it! The seven sins of the specifier are now memorized with the correct names:

1. **Noise**
2. **Silence**
3. **Overspecification**
4. **Contradiction**
5. **Ambiguity**
6. **Forward reference**
7. **Wishful thinking**

Let me know if you need anything else!



Now, i will provide you a system and its linked software requirement. Your task is to thoroughly check if the software requirement sufficiently details its system requirement or not. If there are any issues from the seven ins, highlight it.



Sure! Please provide the system requirement and its linked software requirement, and I will thoroughly evaluate them for completeness and check for any issues related to the seven sins.



```
"SYSRS 015: Increment KAC
AFTER Sending PDCM Packet After Reception of the BRC
THEN
the Sensor IC shall
increment the PDCM_RSP Parameter KAC by 1; the counter is reset to 0 at overflow" "SWRS 017:
Increment KAC
AFTER Sending PDCM Packet After Reception of the BRC
THEN
the Software shall
increment the PDCM_RSP Parameter KAC by 1; "
```



Let's break down the system and software requirements and check if the software requirement (SWRS 017) sufficiently details its system requirement (SYSRS 015). We will also check for any of the seven sins: **Noise, Silence, Overspecification, Contradiction, Ambiguity, Forward Reference, Wishful Thinking.**

Analysis of the Requirements:

- **SYSRS 015 (System Requirement):**
 - **Condition:** AFTER sending PDCM Packet after the reception of the BRC (Broadcast Read Command).
 - **Action:** The Sensor IC shall increment the PDCM_RSP Parameter KAC by 1.
 - **Additional Condition:** The counter (KAC) is reset to 0 at overflow.
- **SWRS 017 (Software Requirement):**
 - **Condition:** AFTER sending PDCM Packet after the reception of the BRC.
 - **Action:** The software shall increment the PDCM_RSP Parameter KAC by 1.
 - **Missing Detail:** No mention of the reset to 0 at overflow condition.

Identified Issues:

1. Silence (Missing Information):

- **Reset to 0 at overflow:** The system requirement (SYSRS 015) explicitly states that the KAC should reset to 0 at overflow. However, this detail is missing from the software requirement (SWRS 017). This is a critical piece of information because it specifies how the counter behaves after reaching its limit, and the absence of this behavior can lead to incorrect implementation.

2. Potential Ambiguity:

- Both requirements seem to be aligned in the core logic regarding the **incrementing of the KAC** parameter, but without the overflow condition in SWRS 017, the behavior of the system may be misinterpreted. The software requirement should explicitly mention this behavior to avoid any ambiguity in how the KAC behaves at overflow.

Summary of Sins Identified:

- **Silence:** The missing detail of resetting the counter to 0 at overflow.
- **Ambiguity:** The absence of this detail may lead to confusion about the correct behavior of the KAC.

Suggested Revision for SWRS 017:

To ensure the software requirement fully aligns with the system requirement, the following revision is recommended for SWRS 017:

Revised SWRS 017:
Increment KAC

AFTER Sending PDCM Packet After Reception of the BRC

THEN the software shall increment the PDCM_RSP Parameter KAC by 1, **and the counter shall be reset to 0 at overflow.**

This revision addresses the missing information and ensures completeness in the software requirement.

Let me know if you'd like to proceed with this revision or need further analysis!



"SYSRS 016: After Reception of the Valid CRM_FCC_CMD,
THEN
the Sensor IC shall
set the PDCM_RSP Parameter KAC = 0x0." "SWRS 018: Increment KAC
AFTER Sending PDCM Packet After Reception of the BRC
THEN
the Software shall
increment the PDCM_RSP Parameter KAC by 1;

After Reception of the Valid CRM_FCC_CMD,
THEN
the Software shall
set the PDCM_RSP Parameter KAC = 0x3."



Let's analyze **SYSRS 016** and its linked software requirement **SWRS 018** in detail to check if there are any issues, and specifically look for any of the seven sins: **Noise, Silence, Overspecification, Contradiction, Ambiguity, Forward Reference, Wishful Thinking.**

Analysis of the Requirements:

- **SYSRS 016 (System Requirement):**
 - **Condition:** After Reception of the Valid CRM_FCC_CMD.
 - **Action:** The Sensor IC shall set the PDCM_RSP Parameter KAC to 0x0 (reset the KAC to 0).
- **SWRS 018 (Software Requirement):**
 - **Condition 1:** After sending PDCM Packet after Reception of the BRC (Broadcast Read Command).
 - **Action:** The software shall increment the PDCM_RSP Parameter KAC by 1.
 - **Condition 2:** After Reception of the Valid CRM_FCC_CMD.
 - **Action:** The software shall set the PDCM_RSP Parameter KAC to 0x3.

Identified Issues:

1. Contradiction:

- The **system requirement (SYSRS 016)** clearly specifies that after the reception of a **Valid CRM_FCC_CMD**, the **KAC** should be set to **0x0** (reset to 0).
- However, in the **software requirement (SWRS 018)**, the KAC is set to **0x3** instead of 0x0, which directly **contradicts** the system requirement. This is a critical mismatch that could lead to incorrect software behavior.

2. Noise:

- The software requirement **SWRS 018** contains the first condition, "**After sending PDCM Packet After Reception of the BRC,**" which is unrelated to the system requirement (SYSRS 016). This is considered **Noise** because it introduces extra information (incrementing the KAC) that is not part of the system requirement being analyzed.
- Although this increment action might be necessary elsewhere, it does not belong in a requirement meant to describe the behavior after receiving the **Valid CRM_FCC_CMD**. This extra detail can confuse the primary focus of SYSRS 016.

3. Overspecification:

- The software requirement sets KAC to **0x3** after the reception of **Valid CRM_FCC_CMD**, which is **not specified** in the system requirement. The system requirement only mentions resetting KAC to **0x0**, not any other value. Therefore, the software requirement is **over-specifying** the behavior by adding an unnecessary detail (setting to 0x3) that is not aligned with the system's intent.

Summary of Sins Identified:

- **Contradiction:** The software requirement contradicts the system requirement by setting KAC to 0x3 instead of 0x0.
- **Noise:** Unnecessary detail about incrementing KAC after the BRC reception is present in the software requirement, which is unrelated to SYSRS 016.
- **Overspecification:** SWRS 018 introduces a value (0x3) for KAC that is not required by SYSRS 016.

Suggested Revision for SWRS 018:

To align with the system requirement and remove any contradictions, noise, and overspecification, the following revision is recommended for SWRS 018:

Revised SWRS 018:

Reset KAC

AFTER Reception of the Valid CRM_FCC_CMD,
THEN the software shall set the PDCM_RSP Parameter KAC to **0x0**.

This revision removes the unrelated information about incrementing KAC after the BRC reception and corrects the contradiction regarding the KAC value.

Let me know if you'd like to proceed with this revision or need further assistance!



"SYSRS 012: BEFORE Leaving the Discovery Mode
AND IF the effective value of the parameter DSI3_physicalAddress is NOT equal to the default

```

value of the Parameter DSI3_physicalAddress
THEN
the Sensor IC shall
set Error: err_discPhysAddrMismatch = 0x1 ("error detected").
"    "SWRS 012 : BEFORE Leaving the Discovery Mode
AND IF the effective value of the Parameter DSI3_physicalAddress (RAM) is NOT equal to the
default value of the Parameter DSI3_physicalAddress (NVM)
THEN
the Software shall
set Error: err_discPhysAddrMismatch = 0x1 ("error detected").
OTHERWISE
the Software shall
set Error: err_discPhysAddrMismatch = 0x0 ("no error").
"

```



Let's break down the system and software requirements and check for any issues, particularly focusing on the seven sins: **Noise, Silence, Overspecification, Contradiction, Ambiguity, Forward Reference, Wishful Thinking.**

Analysis of the Requirements:

- **SYSRS 012 (System Requirement):**
 - **Condition:** BEFORE leaving the Discovery Mode AND IF the effective value of the parameter **DSI3_physicalAddress** is NOT equal to its default value.
 - **Action:** The Sensor IC shall set **Error: err_discPhysAddrMismatch = 0x1** (error detected).
- **SWRS 012 (Software Requirement):**
 - **Condition:** BEFORE leaving the Discovery Mode AND IF the effective value of the **DSI3_physicalAddress** (RAM) is NOT equal to the default value of the **DSI3_physicalAddress** (NVM).
 - **Action:** The software shall set **Error: err_discPhysAddrMismatch = 0x1** (error detected).
 - **Additional Condition:** OTHERWISE, set **Error: err_discPhysAddrMismatch = 0x0** (no error).

Identified Issues:

1. Silence (Missing Information):

- **SYSRS 012** does not specify any action for when the two values of **DSI3_physicalAddress** (effective and default) are equal. In contrast, **SWRS 012** introduces an **OTHERWISE** condition, specifying that **err_discPhysAddrMismatch** should be set to **0x0** (no error) in that case.
- While this is not strictly an error, it introduces behavior that was **silent** in the system requirement. The addition of this condition could be considered acceptable, but it is important to note the mismatch since it adds information not explicitly stated in the system requirement.

2. Overspecification:

- In **SWRS 012**, the software requirement explicitly specifies that the effective value of **DSI3_physicalAddress** comes from **RAM** and the default value comes from **NVM**.
- **SYRS 012** does not mention the specific memory locations (RAM or NVM) for these parameters. By introducing specific sources (RAM and NVM) in the software requirement, it **overspecifies** where these values should come from. This could restrict flexibility if other memory sources are valid but not mentioned here.
- It might be necessary to align the memory source detail with the system requirement or ensure that this does not limit future implementation possibilities.

Summary of Sins Identified:

- **Silence:** The system requirement is silent on what to do when the two values of **DSI3_physicalAddress** are equal. The software requirement introduces an action (setting the error to 0x0) for this scenario, which may be acceptable but should be noted as an addition.
- **Overspecification:** The software requirement overspecifies the memory sources (RAM and NVM) for the **DSI3_physicalAddress** values, whereas the system requirement does not restrict the memory sources.

Suggested Revision for SWRS 012:

To address the overspecification and ensure consistency with the system requirement, the following revision is recommended for SWRS 012:

Revised SWRS 012:

BEFORE Leaving the Discovery Mode

AND IF the effective value of the Parameter **DSI3_physicalAddress** is NOT equal to the default value of the Parameter **DSI3_physicalAddress**

THEN the Software shall set **Error: err_discPhysAddrMismatch = 0x1** (error detected).

OTHERWISE, the Software shall set **Error: err_discPhysAddrMismatch = 0x0** (no error).

This revision removes the explicit mention of RAM and NVM, avoiding potential overspecification.

Let me know if you'd like to proceed with this revision or need further analysis!



"SYRS 005: IF the Sensor IC generates more information than possible to store in Event Buffer or Time Series Data Buffer
THEN
the Sensor IC shall forget the according information." "SWRS 005: - The software shall store information for up to 5 Echo Events.

SWRS 006: - The software shall hold at least the information for up to 100 time series data samples (time stamp and value)
IF NOT possible regarding memory constraints could be reduced up to 50 time series data samples. "



Let's analyze **SYSRS 005** and its linked software requirements, **SWRS 005** and **SWRS 006**, to check for alignment and identify any issues, focusing on the seven sins: **Noise, Silence, Overspecification, Contradiction, Ambiguity, Forward Reference, Wishful Thinking**.

Analysis of the Requirements:

- **SYSRS 005 (System Requirement):**
 - **Condition:** If the Sensor IC generates more information than can be stored in the **Event Buffer** or **Time Series Data Buffer**.
 - **Action:** The Sensor IC shall "forget" the corresponding information (i.e., discard the data that exceeds the buffer capacity).
- **SWRS 005 (Software Requirement):**
 - **Action:** The software shall store information for up to **5 Echo Events**.
- **SWRS 006 (Software Requirement):**
 - **Action:** The software shall hold information for at least **100 time series data samples** (timestamp and value).
 - **Exception:** If memory constraints arise, the buffer can be reduced to store up to **50 time series data samples**.

Identified Issues:

1. Silence (Missing Explicit Handling of Overflow):

- **SYSRS 005** specifies that if the event or time series data exceeds buffer capacity, the **Sensor IC** will discard the excess information ("forget the according information").
- Neither **SWRS 005** nor **SWRS 006** explicitly mention how the software should handle the case where the amount of data exceeds the buffer capacity. There is no explicit action for **discarding or forgetting** data beyond the limits of the buffer (5 echo events or 100/50 time series samples). This absence is a clear **silence** in the software requirements since the system requirement mandates data disposal when the buffer is full.
- **Recommendation:** The software requirements should include explicit instructions to handle the overflow, aligning with the system requirement.

2. Contradiction (Potential Conflict with SYSRS 005):

- **SYSRS 005** allows the system to handle more data than the buffers can store by discarding it when necessary.
- However, **SWRS 006** states that the software **shall hold at least 100 time series data samples** and only reduce this number to 50 in case of memory constraints. While the **50-sample fallback** might be considered an attempt to meet memory limitations, this conflicts with the system requirement's approach of **forgetting** data once buffer limits are exceeded.
- The phrase "hold at least" implies a strict requirement to store this data, but if the buffer limit is reached (as in SYSRS 005), some data may need to be discarded. This introduces a contradiction, as the software requirement implies a strict minimum without clearly stating how to handle overflow.

3. Overspecification (Exact Event and Time Series Limits):

- **SWRS 005** and **SWRS 006** specify exact limits for echo events (5) and time series samples (100, or 50 if constrained). However, **SYSRS 005** does not impose specific numbers for how much data the buffers should hold.
- By specifying the exact limits in the software requirements, there is a risk of **overspecification**, which could limit the flexibility of the system if the buffer capacities change in future implementations.
- **Recommendation:** Consider rephrasing these limits to be adaptable rather than hardcoded, allowing the system to define appropriate buffer capacities based on hardware or memory availability.

Summary of Sins Identified:

- **Silence:** The software requirements do not mention how the system should handle buffer overflow (i.e., "forgetting" or discarding excess data), as specified in the system requirement.
- **Contradiction:** The strict "hold at least" language in the software requirements (particularly SWRS 006) conflicts with the system requirement, which allows for data to be discarded when buffers are full.
- **Overspecification:** The software requirements impose strict limits (5 events, 100/50 time series samples) without flexibility for different buffer capacities, which could be considered overspecification.

Suggested Revision for SWRS 005 and SWRS 006:

To resolve these issues, here is a suggested revision that aligns the software requirements with the system requirement:

Revised SWRS 005:

- The software shall store information for up to 5 Echo Events.
- **If the buffer reaches its capacity, the software shall discard any additional Echo Event data exceeding the buffer limit.**

Revised SWRS 006:

- The software shall hold information for at least **100 time series data samples** (timestamp and value).
- **If memory constraints prevent storing 100 samples, the software shall store up to 50 samples.**
- **If the buffer reaches its capacity, the software shall discard any additional time series data exceeding the buffer limit.**

These revisions address the silence by specifying what happens when buffers are full (data is discarded), resolve the contradiction by removing strict "hold at least" language, and soften the overspecification by allowing flexibility.

Let me know if you'd like to proceed with these changes or need further clarification!



"SYSRS 016: After Reception of the Valid CRM_FCC_CMD,
THEN

the Sensor IC shall
 set the PDCM_RSP Parameter KAC = 0x0." "SWRS 018: Increment KAC
 AFTER Sending PDCM Packet After Reception of the BRC
 THEN
 the Software shall
 increment the PDCM_RSP Parameter KAC by 1;

After Reception of the Valid CRM_FCC_CMD,
 THEN
 the Software shall
 set the PDCM_RSP Parameter KAC = 0x3."



Let's analyze **SYSRS 016** and **SWRS 018** for any potential issues with alignment and examine them for any of the seven sins: **Noise, Silence, Overspecification, Contradiction, Ambiguity, Forward Reference, Wishful Thinking**.

Analysis of the Requirements:

- **SYSRS 016 (System Requirement):**
 - **Condition:** After Reception of the **Valid CRM_FCC_CMD**.
 - **Action:** The Sensor IC shall set the **PDCM_RSP Parameter KAC = 0x0** (reset the KAC to zero).
- **SWRS 018 (Software Requirement):**
 - **Condition 1:** After sending PDCM Packet after Reception of the BRC (Broadcast Read Command).
 - **Action:** The software shall increment the **PDCM_RSP Parameter KAC by 1**.
 - **Condition 2:** After Reception of the **Valid CRM_FCC_CMD**.
 - **Action:** The software shall set the **PDCM_RSP Parameter KAC = 0x3**.

Identified Issues:

1. Contradiction:

- **SYSRS 016** specifies that after the reception of a **Valid CRM_FCC_CMD**, the **PDCM_RSP Parameter KAC** must be set to **0x0**.
- However, **SWRS 018** states that after receiving the same **Valid CRM_FCC_CMD**, the software shall set **KAC = 0x3**, which **contradicts** the system requirement. This is a critical mismatch, as the two requirements specify different values for the same condition.
- **Recommendation:** The software requirement should be corrected to align with the system requirement, ensuring that the KAC is reset to **0x0**, not 0x3.

2. Noise:

- The software requirement **SWRS 018** includes an unrelated condition, "**After sending PDCM Packet after Reception of the BRC,**" where the KAC is incremented. This condition is unrelated to the behavior described in **SYSRS 016**, which only concerns the action after receiving the **CRM_FCC_CMD**.

- Although the increment behavior might be necessary for other aspects of the system, including it here introduces **Noise**, as it distracts from the specific requirement under review. The system requirement deals exclusively with resetting the KAC upon receiving the CRM_FCC_CMD, so the increment condition should be moved to a separate requirement if necessary.
- **Recommendation:** Remove or separate the PDCM Packet and BRC-related increment condition from SWRS 018 to maintain focus and clarity.

3. Overspecification:

- The software requirement (**SWRS 018**) specifies that the KAC should be set to **0x3** after receiving the **CRM_FCC_CMD**, even though the system requirement (**SYSRS 016**) does not call for any value other than **0x0**. This is a form of **overspecification**, introducing unnecessary and potentially incorrect behavior.
- The system requirement is clear about setting the value to **0x0**, and the software should follow this directive without introducing an alternate value.
- **Recommendation:** The software requirement should strictly follow the system requirement and set the KAC to **0x0** without introducing any other values (such as 0x3).

Summary of Sins Identified:

- **Contradiction:** SWRS 018 contradicts SYSRS 016 by setting **KAC = 0x3** instead of the required **0x0** after receiving a **Valid CRM_FCC_CMD**.
- **Noise:** The inclusion of an unrelated condition about incrementing KAC after the BRC reception introduces noise, as it is unrelated to the behavior required by SYSRS 016.
- **Overspecification:** The software requirement unnecessarily specifies a value of **0x3** for KAC, while the system requirement only mandates resetting it to **0x0**.

Suggested Revision for SWRS 018:

To resolve these issues and ensure consistency with the system requirement, the following revision is recommended for SWRS 018:

Revised SWRS 018:

After Reception of the Valid CRM_FCC_CMD,

THEN the software shall set the PDCM_RSP Parameter KAC = 0x0.

By removing the unrelated increment condition and correcting the KAC reset value to **0x0**, this revised software requirement aligns with the system requirement.

Let me know if you'd like to proceed with this revision or need further assistance!



if you identify soimething as a sin, dont identify that part again



Got it! Let's reanalyze **SYSRS 016** and **SWRS 018**, keeping your instruction in mind to avoid repeating identified sins.

Analysis of the Requirements:

- **SYSRS 016 (System Requirement):**
 - **Condition:** After Reception of the **Valid CRM_FCC_CMD**.
 - **Action:** The Sensor IC shall set the **PDCM_RSP Parameter KAC = 0x0** (reset the KAC to zero).
- **SWRS 018 (Software Requirement):**
 - **Condition 1:** After sending PDCM Packet after Reception of the BRC (Broadcast Read Command).
 - **Action:** The software shall increment the **PDCM_RSP Parameter KAC by 1**.
 - **Condition 2:** After Reception of the **Valid CRM_FCC_CMD**.
 - **Action:** The software shall set the **PDCM_RSP Parameter KAC = 0x3**.

Identified Issue:

- **Contradiction:** Already noted.

Suggested Revision for SWRS 018:

Revised SWRS 018:

After Reception of the Valid CRM_FCC_CMD,

THEN the software shall set the **PDCM_RSP Parameter KAC = 0x0**.

This revision corrects the contradiction and ensures the software behavior aligns with the system requirement.



no, i mean you identified contradiction again as over specification



Ah, I see! Thanks for clarifying. Let's address the analysis by focusing on identifying unique sins without overlapping them. Here's the corrected breakdown:

Analysis of the Requirements:

- **SYSRS 016 (System Requirement):**
 - **Condition:** After Reception of the **Valid CRM_FCC_CMD**.
 - **Action:** The Sensor IC shall set the **PDCM_RSP Parameter KAC = 0x0** (reset the KAC to zero).
- **SWRS 018 (Software Requirement):**
 - **Condition 1:** After sending PDCM Packet after Reception of the BRC (Broadcast Read Command).
 - **Action:** The software shall increment the **PDCM_RSP Parameter KAC by 1**.
 - **Condition 2:** After Reception of the **Valid CRM_FCC_CMD**.

- **Action:** The software shall set the **PDCM_RSP Parameter KAC = 0x3**.

Identified Sin:

1. **Contradiction:** The software requirement (SWRS 018) specifies setting **KAC = 0x3** after receiving the **Valid CRM_FCC_CMD**, whereas the system requirement (SYSRS 016) specifies **KAC = 0x0**. This is a contradiction that must be resolved.

Suggested Revision for SWRS 018:

Revised SWRS 018:

After Reception of the Valid CRM_FCC_CMD,

THEN the software shall set the **PDCM_RSP Parameter KAC = 0x0**.

This revision eliminates the contradiction and aligns the software behavior with the system requirement.

Let me know if you need further clarification!



what other sin you have in this?



Let's take a fresh look and see if there is any other **unique** sin besides the **Contradiction**.

Requirements Recap:

- **SYSRS 016:** After receiving the **Valid CRM_FCC_CMD**, the Sensor IC must set **PDCM_RSP Parameter KAC = 0x0**.
- **SWRS 018:**
 - After sending PDCM Packet after Reception of the BRC, the software increments **KAC by 1**.
 - After receiving the **Valid CRM_FCC_CMD**, the software sets **KAC = 0x3**.

Additional Sin Identified:

- **Noise:** The software requirement **SWRS 018** includes a condition about incrementing the KAC after sending the **PDCM Packet after Reception of the BRC**, which is unrelated to the system requirement (SYSRS 016). This introduces **noise** because this condition is not relevant to the system's intent of handling the **CRM_FCC_CMD**. The system requirement is focused only on resetting **KAC** when the **CRM_FCC_CMD** is received, so this additional detail about incrementing KAC elsewhere is unnecessary in this context.

Summary of Sins Identified:

- **Contradiction:** SWRS 018 specifies **KAC = 0x3**, whereas SYSRS 016 specifies **KAC = 0x0**.
- **Noise:** The incrementing condition related to the **BRC** is irrelevant to the system requirement and creates unnecessary clutter.

Suggested Revision for SWRS 018:

Revised SWRS 018:

After Reception of the Valid CRM_FCC_CMD,

THEN the software shall set the PDCM_RSP Parameter KAC = 0x0.

This revision removes both the contradiction and the noise, ensuring the software requirement aligns precisely with the system requirement.

Let me know if you'd like further assistance!

This conversation is archived. To continue, please unarchive it first.

 Unarchive