First Example of Requirement Set that is Complete, NOT Unique and Conflicting

R1: When [Left_Input_Device is active], then the SOI shall activate Middle_Output_Device.

R2: When [Left_Input_Device is inactive] OR [Right_Input_Device is inactive], then the SOI shall de-activate Middle_Output_Device.

Step 1:

L = Left_Input_Device is Active   R1: $L \rightarrow M$
R = Right_Input_Device is Active   R2: $\neg L \lor \neg R \rightarrow \neg M$
M = Middle_Output_Device is Active

						Step 4:		
			Step 3:			Conflict	*(Completeness)*	*(Uniqueness)*
L	R	¬L	¬R	L	¬L V ¬R	(∃x A(x)) ∧ (∃x∃y (A(x) ∧ A(y) ∧ (x≠y))	*(∃x A(x))*	*(∃x A(x)) ∧ (∀x (∀y ((A(x) ∧ A(y)) → (x = y)))*
0	0	1	1	0	1	0	1	1
0	1	1	0	0	1	0	1	1
1	0	0	1	**1**	**1**	**1**	1	**0**
1	1	0	0	1	0	0	1	1

Step 5:
Is Conflict Test a Contradiction?   **FALSE**
Conclusion: Requirement set is Conflicting

Example of Requirement Set that is Complete, NOT Unique and NOT Conflicting

R1: When [Left_Input_Device is active] AND [Right_Input_Device is active], then the SOI shall activate Middle_Output_Device.

R2: When [Right_Input_Device is active] AND [Left_Input_Device is active], then the SOI shall activate Middle_Output_Device.

R3: When [Left_Input_Device is inactive] OR [Right_Input_Device is inactive], then the SOI shall de-activate Middle_Output_Device.

Step 1:

L = Left_Input_Device is Active	R1:	L ∧ R → M
R = Right_Input_Device is Active	R2:	R ∧ L → M
M = Middle_Output_Device is Active	R3:	¬L ∨ ¬R → ¬M

Step 2: R1_2   L ∧ R → M

Step 3:   **Step 4:**

L	R	¬L	¬R	L ∧ R	R ∧ L	¬L ∨ ¬R	L ∧ R	Conflict (∃x A(x)) ∧ (∃x∃y (A(x) ∧ A(y) ∧ (x≠y))	(Completeness) (∃x A(x))	(Uniqueness) (∃x A(x)) ∧ (∀x (∀y ((A(x) ∧ A(y)) → (x = y)))
0	0	1	1	0	0	1	0	0	1	1
0	1	1	0	0	0	1	0	0	1	1
1	0	0	1	0	0	1	0	0	1	1
1	1	0	0	1	1	0	1	0	1	**0**

Step 5:

Is Conflict Test a Contradiction?   **TRUE**

Conclusion: **Requirement set is NOT Conflicting**

Example of Requirement Set that is Incomplete

R1: When [Left_Input_Device is active] AND [Right_Input_Device is active], then the SOI shall activate Middle_Output_Device.

R2: When [Left_Input_Device is inactive] AND [Right_Input_Device is inactive], then the SOI shall de-activate Middle_Output_Device.

Step 1:

L = Left_Input_Device is Active   R1:   L ∧ R → M
R = Right_Input_Device is Active   R2:   ¬L ∧ ¬R → ¬M
M = Middle_Output_Device is Active

L	R	¬L	¬R	L ∧ R	¬L ∧ ¬R	Step 4: Conflict (∃x A(x)) ∧ (∃x∃y (A(x) ∧ A(y) ∧ (x≠y))	(Completeness) (∃x A(x))	(Uniqueness) (∃x A(x)) ∧ (∀x (∀y ((A(x) ∧ A(y)) → (x = y)))
0	0	1	1	0	1	0	1	1
0	1	1	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	1	1

(Header note: "Step 3:" appears above the L ∧ R / ¬L ∧ ¬R columns)

Step 5:

Is Conflict Test a Contradiction?   **TRUE**

Conclusion: Requirement set is NOT Conflicting

Example of Requirement Set that is Complete, Unique, not Conflicting

R1: When [Left_Input_Device is active] AND [Right_Input_Device is active], then the SOI shall activate Middle_Output_Device.

R2: When [Left_Input_Device is inactive] OR [Right_Input_Device is inactive], then the SOI shall de-activate Middle_Output_Device.

Step 2:

L = Left_Input_Device is Active	R1:	L ∧ R → M
R = Right_Input_Device is Active	R2:	¬L ∨ ¬R → ¬M
M = Middle_Output_Device is Active		

Step 4:

						Conflict	(Completeness)	(Uniqueness)
				Step 3:		Conflict	(Completeness)	(Uniqueness)
L	R	¬L	¬R	L ∧ R	¬L ∨ ¬R	(∃x A(x)) ∧ (∃x∃y (A(x) ∧ A(y) ∧ (x≠y))	(∃x A(x))	(∃x A(x)) ∧ (∀x (∀y ((A(x) ∧ A(y)) → (x = y)))
0	0	1	1	0	1		0	1	1
0	1	1	0	0	1		0	1	1
1	0	0	1	0	1		0	1	1
1	1	0	0	1	0		0	1	1

Step 5:

Is Conflict Test a Contradiction?   **TRUE**

Conclusion: **Requirement set is NOT Conflicting**