To instantiate the safety case pattern based on the given predicates, placeholders in the goals, strategies, contexts, and solutions need to be filled with concrete values. For this example, let's assume we are working with a pacemaker system. Below is an instantiated safety case with placeholders replaced and displayed in a hierarchical tree format using dashes (-) to denote different levels:

```plaintext

G1: Operational safety is verified in Pacemaker System

- S1: Argument over the satisfaction of all specs over Pacemaker System

- C1: All specs are given by PacemakerSystem.spec.allInstances()

- G2: All operational hazards are mitigated

- C2: Operational hazards are given by pacemakerSystem.operationalHazard.allInstances()

- S2: Argument over reliability in all suitable levels of Pacemaker System

- S3: Argument over operational hazards

- G3: Electrical hazard is mitigated

- S4: Argument over the applied scenarios of Electrical hazard

- C3: All related scenarios are given by ElectricalHazard

- G4: Electrical hazard is mitigated under high voltage scenario

- S5: Argument over all specs related to high voltage scenario

- C4: All related specs are given by highVoltageScenario.spec.allInstances()

- G5: Voltage tolerance spec is appropriate for high voltage scenario

- S6: Argument the appropriateness of Voltage tolerance spec over properties

- C5: Properties are given by VoltageToleranceSpec.property.allInstances()

- G6: Voltage tolerance property is appropriate for Voltage tolerance spec

- S7: Argument over the source of the Voltage tolerance property definition

- C6: Source is given by VoltageToleranceProperty.source

- G7: Manufacturer specification is appropriate and trustworthy

- Sn1: Manufacturer certification

- G8: Voltage tolerance property definition is sufficient

- Sn2: Voltage tolerance specification document

- G3: Software malfunction hazard is mitigated

- S4: Argument over the applied scenarios of Software malfunction hazard

- C3: All related scenarios are given by SoftwareMalfunctionHazard

- G4: Software malfunction hazard is mitigated under unexpected input scenario

- S5: Argument over all specs related to unexpected input scenario

- C4: All related specs are given by UnexpectedInputScenario.spec.allInstances()

- G5: Input validation spec is appropriate for unexpected input scenario

- S6: Argument the appropriateness of Input validation spec over properties

- C5: Properties are given by InputValidationSpec.property.allInstances()

- G6: Input validation property is appropriate for Input validation spec

- S7: Argument over the source of the Input validation property definition

- C6: Source is given by InputValidationProperty.source

- G7: Testing protocol is appropriate and trustworthy

- Sn1: Testing protocol documentation

- G8: Input validation property definition is sufficient

- Sn2: Input validation specification document

```

This hierarchical structure fulfills the given pattern by instantiating specific goals, strategies, contexts, and solutions relevant to a hypothetical pacemaker system. Each level represents a deeper layer in the assurance case, where strategies support goals, and goals are contextually framed. The documented evidence supports claims about the system's safety properties.