| **Step** | **Description** | **Protégé Action / TTL Snippet** |
| --- | --- | --- |
| 1 | Add **ConsistencyPrinciple** as subclass of **UsabilityPrinciple** | **Protégé:** Classes → select **UsabilityPrinciple** → Add subclass **ConsistencyPrinciple** **TTL:** ttl<br>:ConsistencyPrinciple a owl:Class ;<br> rdfs:subClassOf :UsabilityPrinciple ;<br> rdfs:label "Consistency Principle"@en ;<br> rdfs:comment "Ensures uniform UI behavior and appearance across platforms."@en .<br> |
| 2 | Define **CrossPlatformEnterpriseContext** under **TaskContext** | **Protégé:** Classes → select **TaskContext** → Add subclass **CrossPlatformEnterpriseContext** **TTL:** ttl<br>:CrossPlatformEnterpriseContext a owl:Class ;<br> rdfs:subClassOf :TaskContext ;<br> rdfs:label "Cross-Platform Enterprise Context"@en ;<br> rdfs:comment "Enterprise app context spanning desktop, mobile, etc."@en .<br> |
| 3 | Create object-property **appliesToContext** linking principles to contexts | **Protégé:** Object Properties → “+” → Name **appliesToContext** • Domain: **DesignPrinciple** • Range: **TaskContext** • Inverse: **isContextForPrinciple** **TTL:** ttl<br>:appliesToContext a owl:ObjectProperty ;<br> rdfs:domain :DesignPrinciple ;<br> rdfs:range :TaskContext ;<br> rdfs:label "applies to context"@en ;<br> rdfs:comment "Links a design principle to applicable contexts."@en ;<br> owl:inverseOf :isContextForPrinciple .<br> |
| 4 | Assert restriction: **ConsistencyPrinciple** must apply in **CrossPlatformEnterpriseContext** | **Protégé:** Classes → select **ConsistencyPrinciple** → SubClassOf → Add restriction: appliesToContext some CrossPlatformEnterpriseContext **TTL:** ttl<br>:ConsistencyPrinciple rdfs:subClassOf [<br> a owl:Restriction ;<br> owl:onProperty :appliesToContext ;<br> owl:someValuesFrom :CrossPlatformEnterpriseContext<br>] .<br> |
| 5 | Query for consistency principles in the cross-platform enterprise context | **SPARQL:** sparql<br>PREFIX onto: <http://your-ontology#><br>SELECT ?principle<br>WHERE {<br> ?principle a onto:ConsistencyPrinciple ;<br> onto:appliesToContext onto:CrossPlatformEnterpriseContext .<br>}<br> |

o4-mini

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX owl: <http://www.w3.org/2002/07/owl#>

PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX onto: <http://www.semanticweb.org/asifcomputer/ontology/2025/ui\_ux\_automation#>

SELECT ?principle ?context

WHERE {

?principle a onto:ConsistencyPrinciple ;

onto:appliesToContext ?context .

?context a onto:TaskContext .

}

**How G4-A2 Drives UI/UX Automation & Solves Client Requirements**

Answering **G4-A2** ("What consistency principles should be applied when designing cross-platform enterprise applications?") bridges the gap between design theory and automation by:

1. **Automating Compliance Checks**:
   * Codifies consistency rules (e.g., uniform button sizes, navigation patterns) into the ontology.
   * LLM-driven tools can query these rules via SPARQL to validate designs against client requirements *automatically*.
2. **Ensuring Cross-Platform Uniformity**:
   * Links principles (e.g., ConsistencyPrinciple) to specific contexts (e.g., CrossPlatformEnterpriseContext).
   * Automation tools use this mapping to enforce identical UI behavior across web, mobile, and desktop, reducing manual audits.
3. **Aligning with Client Needs**:
   * Clients demand seamless user experiences across platforms. By embedding these principles into the ontology, automation tools ensure designs adhere to enterprise standards (e.g., WCAG, brand guidelines).
   * Example: Auto-generate platform-specific UI code while maintaining visual/functional consistency.
4. **Reducing Development Time**:
   * SPARQL queries retrieve context-specific principles, enabling tools to auto-suggest design components (e.g., "Use 24px buttons for touch targets in enterprise mobile apps").