prd06 plugin and extensibility

PRD 06 — Plugin and Extensibility System

1. Product Vision
The Plugin and Extensibility System allows the translator and tools to evolve with new file types, game versions, and advanced transformations without requiring core engine rewrites. Plugins extend the functionality in a controlled, safe manner and integrate with diagnostics.

2. Plugin Types
- File Type Adapters: Support new or custom resources inside mods.
- Game Version Packs: Provide patch-specific validation rules and schemas.
- Transformation Packs: Encapsulate higher-level logic (e.g., autonomy rules, balance passes).

3. Functional Requirements
3.1 Plugin Interface
- R1.1: Define a stable interface for plugins that includes:
  Input: raw resource data and optional IR fragments.
  Output: IR additions or modifications, along with diagnostics.
- R1.2: Ensure plugins can register for specific resource types or processing stages.

3.2 Safety and Isolation
- R2.1: Run plugins in a way that minimizes risk to stability (e.g., controlled execution or sandboxing where practical).
- R2.2: Guard against plugin failures: a crashing plugin should not crash the entire engine.
- R2.3: Capture plugin errors as structured diagnostics using EngineError.

3.3 Plugin Management
- R3.1: Maintain a registry of available plugins with metadata (name, version, author, supported game versions).
- R3.2: Allow projects to enable and disable plugins.
- R3.3: Present compatibility warnings when plugins do not match the current engine or game version.

4. Non-Functional Requirements
- Stability: Poorly behaving plugins must be contained.
- Versioning: Plugins should declare compatible engine versions and game patches.
- Longevity: Design plugin APIs with forward compatibility in mind.
- Observability: Capture plugin performance and error metrics.