Rules Matrix Specification v1

1. Purpose

The Rules Matrix defines a **formal, machine-readable representation of captioning rules** derived from standards such as the BBC Subtitle Guidelines.

It provides:

- A structured source of truth for all rules
- Direct input for the Standards Rules Engine
- A reference layer for validation, scoring, and repair
- A consistent interface for Claude-based analysis
- Alignment between editorial standards and system behaviour

---

2. Scope

The Rules Matrix includes all rules required to enforce:

- Readability
- Timing and synchronisation
- Segmentation and line breaking
- Speaker identification
- Visual safety and positioning
- Typography and layout constraints
- Controlled expressive behaviour
- System-specific enhancement constraints

---

3. Design Principles

3.1 Explicitness

All rules must be defined explicitly. No rule should exist only as prose.

3.2 Traceability

Each rule must reference its source (e.g. BBC section).

3.3 Machine-Checkability

Rules must be structured so they can be:

- evaluated programmatically
- tested automatically
- logged in decision logs

3.4 Classification

Each rule must belong to a defined rule class:

- Class A: Hard accessibility rules
- Class B: Preferred presentation rules
- Class C: Controlled expressive rules
- Class D: Project-specific advanced rules

3.5 Separation of Concern

Rules define **constraints**, not behaviour.
Behaviour is determined by the Policy Engine.

---

4. Rule Structure (JSON Schema)

```
{
  "rule_id": "string",

  "source": {
    "document": "BBC Subtitle Guidelines",
    "section": "string",
    "description": "string"
  },

  "category": "timing | segmentation | layout | typography | speaker |
visual | enhancement",

  "rule_class": "A | B | C | D",

  "description": "human-readable explanation",

  "condition": {
    "type": "expression",
    "logic": "string or structured condition"
  },

  "thresholds": {
```

```
    "max_wpm": 180,
    "min_duration_per_word": 0.33
  },

  "evaluation": {
    "method": "deterministic | heuristic | ML-assisted"
  },

  "failure_condition": "description of violation",

  "severity": "low | medium | high | critical",

  "auto_fix": {
    "available": true,
    "strategy": "extend_duration | split_caption | resegment |
suppress_feature"
  },

  "applies_to": [
    "speech",
    "sound",
    "music"
  ],

  "dependencies": [
    "other_rule_id"
  ],
"constraints_generated": [
  "READABILITY_PRIORITY",
  "TIMING_LIMIT"
]
}
```

Each rule must explicitly declare the constraints it produces when evaluated. These constraints are consumed by the Policy Engine for conflict resolution.

Severity must reflect impact on readability, accessibility, or system stability, not implementation difficulty.

---

5. Rule Categories

Rules should be grouped into the following categories:

Category	Description
timing	reading speed, duration, sync
segmentation	line breaks, phrase structure

Category	Description
layout	number of lines, positioning
typography	font size, scaling, contrast
speaker	identification, colour, labels
visual	safe regions, obstruction
enhancement	constraints on advanced features

---

6. Example Rules

6.1 Timing Rule (BBC-TIM-001)

```
{
  "rule_id": "BBC-TIM-001",
  "category": "timing",
  "rule_class": "A",
  "description": "Caption must allow sufficient reading time",

  "thresholds": {
    "max_wpm": 180
  },

  "evaluation": {
    "method": "deterministic"
  },

  "failure_condition": "WPM exceeds threshold",

  "severity": "critical",

  "auto_fix": {
    "available": true,
    "strategy": "extend_duration"
  }
}
```

6.2 Line Break Rule (BBC-SEG-001)

```
{
  "rule_id": "BBC-SEG-001",
  "category": "segmentation",
  "rule_class": "A",
  "description": "Avoid splitting tightly bound grammatical units",
```

```
  "evaluation": {
    "method": "heuristic"
  },

  "failure_condition": "line break within syntactic unit",

  "severity": "high",

  "auto_fix": {
    "available": true,
    "strategy": "resegment"
  }
}
```

6.3 Enhancement Constraint Rule

```
{
  "rule_id": "SYS-ENH-001",
  "category": "enhancement",
  "rule_class": "D",
  "description": "Enhancements must not reduce readability",

  "evaluation": {
    "method": "deterministic"
  },

  "failure_condition": "enhancement causes reflow or instability",

  "severity": "critical",

  "auto_fix": {
    "available": true,
    "strategy": "suppress_enhancement"
  }
}
```

---

7. Rule Evaluation Flow

Caption Input
→ Apply relevant rules
→ Evaluate conditions
→ Record pass/fail/warning
→ Trigger auto-fix or policy decision
→ Log decision

---

8. Integration with Decision Logs

Each rule evaluation must generate a corresponding decision log entry containing:

- rule_id
- evaluation result
- applied thresholds
- decision outcome
- reasoning

---

9. Integration with Policy Engine

The Rules Matrix does not determine behaviour directly.

Instead:

- Rules define constraints
- Policy Engine resolves conflicts
- Renderer executes final outcome

---

10. Validation and Scoring

Rules must support scoring models:

- pass → full score
- warning → partial penalty
- fail → full penalty

Scores may be aggregated into:

- readability score
- timing score
- layout score
- overall compliance score

---

11. Claude Integration

Claude can use the Rules Matrix to:

- extract missing rules

- validate caption behaviour
- generate test cases
- identify gaps in implementation
- propose rule improvements

---

12. Maintenance and Versioning

Rules must support:

- version tracking
- change history
- backward compatibility
- deprecation markers

---

13. Future Extensions

- language-specific rule variations
- adaptive thresholds (context-aware)
- ML-assisted segmentation rules
- user-specific readability tuning

Summary

The Rules Matrix:

- Converts BBC guidance into executable structure
- Feeds directly into the Rules Engine
- Integrates with Decision Logs
- Enables Claude to operate at a system level