Version: 1.0.0 **Target:** 0.134.1-Beta

Implementation Plan - Phase 2: Text Report Header Standardization

This plan applies the standard "Three-Line Title" and "Metadata Header" pattern to the remaining 8 text reports, completing the standardization mandated by the CLA Reports Style Guide.

1. contest_tools/reports/text_rate_sheet.py

Rationale: This report currently lacks a global header and relies on manual string centering for table titles. We will inject the standard header generator into the `generate` method.

Changes:

1. **Import:** Add `format_text_header`, `get_cty_metadata`, `get_standard_title_lines` from `._report_utils`.
2. **Generate Method:**
   - Calculate `modes_present` for smart scoping.
   - Generate `title_lines` and `meta_lines`.
   - Calculate `table_width` from the first block's width.
   - Generate `header_block` and prepend it to `full_content`.

CODE_BLOCK python --- BEGIN DIFF --- @@ -26,7 +26,7 @@ import pandas as pd import os from ..contest_log import ContestLog from .report_interface import ContestReport -from ._report_utils import create_output_directory # (Note: clean up unused import if necessary, but focusing on footer removal) +from ._report_utils import format_text_header, get_cty_metadata, get_standard_title_lines from ..data_aggregators.time_series import TimeSeriesAggregator

class Report(ContestReport): @@ -94,6 +94,9 @@ if not available_modes: available_modes = ["QSO"] # Fallback

- `        # Calculate modes present for smart scoping`
- `        modes_present = set(available_modes)`
- `        report_blocks = []`

`        # --- BLOCK 1: Overall Summary ---`

@@ -141,8 +144,15 @@ display_net = gross_qsos if include_dupes else net_qsos footer = f"Gross QSOs={gross_qsos} Dupes={dupes} Net QSOs={display_net}"

- `        # --- Generate Standard Header ---`

```
        # Measure width from the first block
        block1_lines = block1.split('\n')
        table_width = len(block1_lines[3]) if len(block1_lines) > 3 else 80

        title_lines = get_standard_title_lines(self.report_name, [log], "All Bands", Non
        meta_lines = ["Contest Log Analytics by KD4D", get_cty_metadata([log])]
        header_block = format_text_header(table_width, title_lines, meta_lines)
        # --- Assembly ---
        full_content = "\n\n".join(report_blocks) + "\n\n" + footer + "\n"
        full_content = "\n".join(header_block) + "\n\n" + "\n\n".join(report_blocks) + "

        os.makedirs(output_path, exist_ok=True)
        filename = f"{self.report_id}_{callsign}.txt"
```

--- END DIFF --- **CODE_BLOCK**

2. contest_tools/reports/text_continent_summary.py

Rationale: Replace the manual title centering logic with the standard utility.

Changes:

1. **Import:** Add header utilities.
2. **Generate Method:** Remove legacy title construction lines and replace with `get_standard_title_lines` and `format_text_header`.

CODE_BLOCK python --- BEGIN DIFF --- @@ -21,7 +21,7 @@ import pandas as pd import os from ..contest_log import ContestLog from .report_interface import ContestReport +from ._report_utils import format_text_header, get_cty_metadata, get_standard_title_lines

class Report(ContestReport): """ @@ -82,16 +82,13 @@ table_width = len(table_header) separator = "-" * table_width

```
        title1 = f"--- {self.report_name} ---"
        title2 = f"{year} {contest_name} - {callsign}"
        report_lines = []
        if len(title1) > table_width or len(title2) > table_width:
             header_width = max(len(title1), len(title2))
             report_lines.append(f"{title1.ljust(header_width)}")
             report_lines.append(f"{title2.center(header_width)}")
```

- `        else:`
- `             header_width = table_width`
- `             report_lines.append(title1.center(header_width))`
- `        report_lines.append(title2.center(header_width))`
-
- `        # --- Standard Header ---`
- `        modes_present = set(df['Mode'].dropna().unique())`
- `        title_lines = get_standard_title_lines(self.report_name, [log], "All Bands", Non`
- `        meta_lines = ["Contest Log Analytics by KD4D", get_cty_metadata([log])]`
-
- `        header_block = format_text_header(table_width, title_lines, meta_lines)`
- `        report_lines.extend(header_block)`

```
        report_lines.append("")
        report_lines.append(table_header)
```

--- END DIFF --- **CODE_BLOCK**

3. contest_tools/reports/text_comparative_continent_summary.py

Rationale: Similar to single summary, replace manual centering with standard comparative header generation.

Changes:

1. **Import:** Add header utilities.
2. **Generate Method:** Replace legacy title block. Note: Comparative report passes `self.logs` (list) to utilities.

CODE_BLOCK python --- BEGIN DIFF --- @@ -24,7 +24,7 @@ from ..contest_log import ContestLog from ..contest_definitions import ContestDefinition from .report_interface import ContestReport -from ._report_utils import _sanitize_filename_part +from ._report_utils import _sanitize_filename_part, format_text_header, get_cty_metadata, get_standard_title_lines

class Report(ContestReport): """ @@ -92,23 +92,16 @@ contest_name = first_log.get_metadata().get('ContestName', 'UnknownContest') year = first_log.get_processed_data()['Date'].iloc[0].split('-')[0] if not first_log.get_processed_data().empty else "----"

- `    title1 = f"--- {self.report_name} ---"`
- `    title2 = f"{year} {contest_name} - {', '.join(all_calls)}"`

```
•
•     title1_width = len(title1)
•     title2_width = len(title2)
•
•     if title1_width > table_width or title2_width > table_width:
•         header_width = max(title1_width, title2_width)
•         report_lines = [
•             f"{title1.ljust(header_width)}",
•             f"{title2.center(header_width)}",
•             ""
•         ]
•     else:
•         header_width = table_width
•         report_lines = [
•             title1.center(header_width),
•             title2.center(header_width),
•             ""
•         ]
•     # --- Standard Header ---
•     modes_present = set()
•     for log in self.logs:
•         df = log.get_processed_data()
•         if 'Mode' in df.columns:
•             modes_present.update(df['Mode'].dropna().unique())
•
•     title_lines = get_standard_title_lines(self.report_name, self.logs, "All Bands", Non
•     meta_lines = ["Contest Log Analytics by KD4D", get_cty_metadata(self.logs)]
•
•     header_block = format_text_header(table_width, title_lines, meta_lines)
```

- ```
  report_lines = header_block + [""]

  report_lines.append(table_header)
  report_lines.append(separator)
  ```

--- END DIFF --- **CODE_BLOCK**

4. contest_tools/reports/text_continent_breakdown.py

Rationale: This report uses a complex manual insertion logic for titles. We will simplify this by generating the header block first and then appending the grid data.

Changes:

1. **Import:** Add header utilities.
2. **Generate Method:** Remove the "MOVED BLOCK" logic (lines 114-123) and replace with standard generation at the start of the report construction.

CODE_BLOCK python --- BEGIN DIFF --- @@ -21,6 +21,7 @@ import pandas as pd import os from ..contest_log import ContestLog from .report_interface import ContestReport +from ._report_utils import format_text_header, get_cty_metadata, get_standard_title_lines

class Report(ContestReport): """ @@ -114,15 +115,14 @@ if grid_layout: table_width = col_width * len(grid_layout[0])

- ```
  # --- THIS BLOCK MOVED TO AFTER WIDTH CALCULATION ---
  title1 = f"--- {self.report_name} ---"
  title2 = f"{year} {contest_name} - {callsign}"
  header_width = max(table_width, len(title1), len(title2))

  if len(title1) > table_width or len(title2) > table_width:
      report_lines.insert(0, f"{title2.center(header_width)}")
      report_lines.insert(0, f"{title1.ljust(header_width)}")
  else:
      report_lines.insert(0, f"{title2.center(header_width)}")
      report_lines.insert(0, title1.center(header_width))
  report_lines.insert(2, "")
  # --- END MOVED BLOCK ---
  # --- Standard Header ---
  ```

- `            modes_present = set(df['Mode'].dropna().unique())`
- `            title_lines = get_standard_title_lines(self.report_name, [log], "All Bands",`
- `            meta_lines = ["Contest Log Analytics by KD4D", get_cty_metadata([log])]`
-
- `            header_block = format_text_header(table_width, title_lines, meta_lines)`
-
- `            # Insert header at the top`
- `            for line in reversed(header_block):`
- `                report_lines.insert(0, line)`
- `            report_lines.insert(len(header_block), "")`

`            for grid_row in grid_layout:`

--- END DIFF --- **CODE_BLOCK**

5. contest_tools/reports/text_multiplier_summary.py

Rationale: Standardize the header. Handle the "no multipliers found" edge case to ensure it still gets a proper header.

Changes:

1. **Import:** Add header utilities.
2. **Generate Method:**
   - Calculate `table_width` earlier or use a default if no data.
   - Replace manual title construction.
   - Ensure header is applied even in the "No multipliers found" branch.

CODE_BLOCK python --- BEGIN DIFF --- @@ -32,7 +32,7 @@ import logging import hashlib from ..contest_log import ContestLog from ..data_aggregators.multiplier_stats import MultiplierStatsAggregator from .report_interface import ContestReport +from ._report_utils import format_text_header, get_cty_metadata, get_standard_title_lines

class Report(ContestReport): """ @@ -107,17 +107,24 @@

`    sorted_mults = sorted(list(unique_mults))`

- `    # --- Report Header Setup ---`
- `    first_row_date = combined_df[0]['Date'] if combined_df else "----"`
- `    year = first_row_date.split('-')[0]`
- `    mode_title_str = f" ({mode_filter})" if mode_filter else ""`
- `    title1 = f"--- {self.report_name}: {mult_name}{mode_title_str} ---"`

- title2 = f"{year} {contest_def.contest_name} - {', '.join(all_calls)}"
- # --- Header Preparation ---
- modes_present = {mode_filter} if mode_filter else set()
- if not modes_present:
- # Try to derive from logs if not explicit
- for log in self.logs:
- df = log.get_processed_data()
- if 'Mode' in df.columns:
- modes_present.update(df['Mode'].dropna().unique())
-
- title_lines = get_standard_title_lines(f"{self.report_name}: {mult_name}", self.logs
- meta_lines = ["Contest Log Analytics by KD4D", get_cty_metadata(self.logs)]
- report_lines = []

# --- Gracefully handle cases with no multipliers ---
if not pivot_data:

- report_lines.append(title1)
- report_lines.append(title2)
- report_lines.append(f"\nNo '{mult_name}' multipliers found to summarize.")
- # Default width for empty report
- header_block = format_text_header(80, title_lines, meta_lines)
- report_lines.extend(header_block)
- report_lines.append(f"\n\nNo '{mult_name}' multipliers found to summarize.")

# --- Save the clean (but empty) report and exit ---
report_content = "\n".join(report_lines) + "\n"
os.makedirs(output_path, exist_ok=True)

@@ -164,13 +171,9 @@ table_header = f"{first_col_header:<{first_col_width}}
+ "".join(header_parts) table_width = len(table_header)
separator = "-" * table_width

- header_width = max(table_width, len(title1), len(title2))
- if len(title1) > table_width or len(title2) > table_width:
- report_lines.append(f"{title1.ljust(header_width)}")

- `        report_lines.append(f"{title2.center(header_width)}")`
- `    else:`
- `        report_lines.append(title1.center(header_width))`
- `        report_lines.append(title2.center(header_width))`
- `    report_lines.append("")`
-
- `    # Generate standard header`
- `    header_block = format_text_header(table_width, title_lines, meta_lines)`
- `    report_lines.extend(header_block)`
- `    report_lines.append("")`
- `    report_lines.append(table_header)`
  `    report_lines.append(separator)`

--- END DIFF --- **CODE_BLOCK**

6. contest_tools/reports/text_wae_score_report.py

Rationale: Standardize WAE single-log score report header.

Changes:

1. **Import:** Add header utilities.
2. **Generate Method:** Remove manual title block and replace.

CODE_BLOCK python --- BEGIN DIFF --- @@ -31,7 +31,7 @@

from ..contest_log import ContestLog from .report_interface import ContestReport -from ._report_utils import get_valid_dataframe, create_output_directory, save_debug_data +from ._report_utils import get_valid_dataframe, create_output_directory, save_debug_data, format_text_header, get_cty_metadata, get_standard_title_lines from ..data_aggregators.wae_stats import WaeStatsAggregator

class Report(ContestReport): @@ -99,19 +99,16 @@])

`    # --- Build Final Report String ---`

- `    # Note: We use qsos_df['Date'] for year to preserve exact logic,`
- `    # but technically we could parse it from headers. Sticking to dataframe for metadata`
- `    year = qsos_df['Date'].iloc[0].split('-')[0]`
- `    contest_name = metadata.get('ContestName', 'UnknownContest')`
- `    title1 = f"--- {self.report_name} ---"`

- `    title2 = f"{year} {contest_name} - {callsign}"`
- `    table_str = table.get_string()`
  `    table_width = len(table_str.split('\n')[0])`
- `    header_width = max(table_width, len(title1), len(title2))`
-
- `    # Standard Header`
- `    modes_present = set(qsos_df['Mode'].dropna().unique())`
- `    title_lines = get_standard_title_lines(self.report_name, [log], "All Bands", None, m`
- `    meta_lines = ["Contest Log Analytics by KD4D", get_cty_metadata([log])]`
- `    header_block = format_text_header(table_width, title_lines, meta_lines)`
- `    report_lines = [`
- `        title1.center(header_width),`
- `        title2.center(header_width),`
- `        "",`
- `    report_lines = header_block + [`
- `        "",`
  `        f"Operating Time: {metadata.get('OperatingTime', 'N/A')}",`
  `        "",`
  `        table_str,`

--- END DIFF --- **CODE_BLOCK**

7. contest_tools/reports/text_wae_comparative_score_report.py

Rationale: Standardize WAE comparative score report header.

Changes:

1. **Import:** Add header utilities.
2. **Generate Method:** Prepend standard header before appending content.

CODE_BLOCK python --- BEGIN DIFF --- @@ -43,7 +43,7 @@ from ..contest_log import ContestLog from ..contest_definitions import ContestDefinition from .report_interface import ContestReport -from ._report_utils import get_valid_dataframe, create_output_directory, _sanitize_filename_part +from ._report_utils import get_valid_dataframe, create_output_directory, _sanitize_filename_part, format_text_header, get_cty_metadata, get_standard_title_lines from ..data_aggregators.wae_stats import WaeStatsAggregator

class Report(ContestReport): @@ -137,13 +137,18 @@ final_score_lines.append(f"{label:<{max_label_width
: {score_str}") report_lines.extend(final_score_lines)

```
    # --- Prepend Titles ---
    first_log = self.logs[0]
    year = first_log.get_processed_data()['Date'].iloc[0].split('-')[0]
    contest_name = first_log.get_metadata().get('ContestName')
    title1 = f"--- {self.report_name} ---"
    title2 = f"{year} {contest_name} - {', '.join(all_calls)}"

    final_report_str = "\n".join([title1, title2] + report_lines) + "\n"
    # --- Generate Standard Header ---
    modes_present = set()
    for log in self.logs:
        df = log.get_processed_data()
        if 'Mode' in df.columns:
            modes_present.update(df['Mode'].dropna().unique())

    title_lines = get_standard_title_lines(self.report_name, self.logs, "All Bands", Non
    meta_lines = ["Contest Log Analytics by KD4D", get_cty_metadata(self.logs)]
    # Calculate roughly width from one of the tables or default
    header_block = format_text_header(80, title_lines, meta_lines)

    final_report_str = "\n".join(header_block + report_lines) + "\n"

    # --- Save to File ---
    create_output_directory(output_path)
```

--- END DIFF --- **CODE_BLOCK**

8. contest_tools/reports/text_comparative_score_report.py

Rationale: Standardize generic comparative score report header.

Changes:

1. **Import:** Add header utilities.

2. **Generate Method:** Remove legacy title generation logic and replace with standard block.

CODE_BLOCK python --- BEGIN DIFF --- @@ -43,7 +43,7 @@ from ..contest_log import ContestLog from ..contest_definitions import ContestDefinition from .report_interface import ContestReport -from ._report_utils import _sanitize_filename_part +from ._report_utils import _sanitize_filename_part, format_text_header, get_cty_metadata, get_standard_title_lines

class Report(ContestReport): """ @@ -127,18 +127,16 @@ final_line = " " * padding + line report_lines.append(final_line)

```
    # --- Prepend Centered Titles ---
    contest_name = first_log.get_metadata().get('ContestName', 'UnknownContest')
    year = first_log.get_processed_data()['Date'].iloc[0].split('-')[0] if not first_log

    title1 = f"--- {self.report_name} ---"
    title2 = f"{year} {contest_name} - {', '.join(all_calls)}"

    header_width = max(table_width, len(title1), len(title2))
    final_header = [
        title1.center(header_width),
        title2.center(header_width),
        ""
    ]
    report_lines = final_header + report_lines
    # --- Generate Standard Header ---
    modes_present = set()
    for log in self.logs:
        df = log.get_processed_data()
        if 'Mode' in df.columns:
            modes_present.update(df['Mode'].dropna().unique())

    title_lines = get_standard_title_lines(self.report_name, self.logs, "All Bands", Non
    meta_lines = ["Contest Log Analytics by KD4D", get_cty_metadata(self.logs)]
```

-
- `    header_block = format_text_header(table_width, title_lines, meta_lines)`
- `    report_lines = header_block + report_lines`

```
    # --- Save to File ---
    report_content = "\n".join(report_lines) + "\n"
```

--- END DIFF --- **CODE_BLOCK**