CASE STUDY

101名・148試合の大会を、ひとつのデータから組み立てる

スポレック大会「ODD Fes vol.1」／抽選・タイムスケジュール・審判割り当て・17種の帳票を自動生成し、当日のスコア共有までを内製した記録

2026.08.15 ／ ODD Fes vol.1 実行委員会

2つのシステムでできている

大会前は「紙をつくる」、当日は「紙をなくす」。目的が違うので、つくり方も分けた。

大会前 帳票生成パイプライン

Python 3.12 + pandas ／ HTML+CSS → WeasyPrint → PDF

エントリーCSV
- ↓ 抽選エンジン　年齢バランスのポット分け
- ↓ スケジューラ　対戦順・コート・時刻
- ↓ 審判アサイナ　制約充足による割り当て
- ↓ 帳票ジェネレータ

17種類のPDF（配布・掲示・運営）

大会当日 スコア共有システム

Google Apps Script + スプレッドシート ／ 静的HTML（Netlify）

本部が入力
- ↓ 承認　承認済みのみ集計に反映
- ↓ 自動集計　順位表・星取表・トーナメント
- ↓ JSON API

参加者のスマホで閲覧（60秒ごとに更新）

設計原則　仕様変更が起きたら、データかロジックを1箇所だけ直して全帳票を再生成する。個別ファイルは手直ししない。

大会前：すべてが属人的だった

経験のある人の頭の中にしか手順がなく、直すたびに全ファイルを触っていた。

領域	従来の課題
組み合わせ抽選	手作業。年齢バランスの考慮が困難
タイムスケジュール	Excel手組み。連戦や空き時間の偏りが発生
コート割り当て	目視調整。同時進行の整合性チェックが困難
審判割り当て	当日その場で調整。トラブルの原因
帳票作成	個別にWord/Excelで作成。修正のたびに全ファイル手直し

> **規模**　参加101名 ／ 82ペア ／ 4部門 ／ 18グループ ／ 8コート ／ プール戦148試合。 当日までに**40回以上の仕様変更**が発生した。

大会当日：本部席に人が張り付かないと回らない

紙のスコアカードが本部に集まり、そこで詰まる構造になっていた。

審判
紙のスコアカードに記入

▶

選手
勝ったチームが本部へ持参

▶

運営
Excelのリーグ表へ手入力

▶

運営
結果を見てトーナメント表を手書き

運営7名は全員が選手兼任

本部が空くと紙が溜まる

参加者にシニア層が多い

審判による直接入力は困難

ボランティア・観客がいない

外部リソースを頼れない

> 「誰が入力するか」に、最後まで答えがなかった。

くじの偶然性と、公平さを両立させる

リーグ分けは当日その場のくじ引き。ただし完全ランダムだと、年齢もチームも偏る。

POT 1
合計年齢が
若いペア

→

POT 2
少し上の
ペア

→

POT 3
…

→

POT 5
合計年齢が
高いペア

部門	参加数	ポット	グループ
混合A	20ペア	5	4
混合B	20ペア	5	4
男子ダブルス	26ペア	5	6
女子ダブルス	16ペア	4	4

混合を2部門に分けた理由

40組と参加が多いため、ペアの合計年齢で2分割した。年齢の近いペア同士で戦えるようにし、どちらにも活躍の機会をつくる意図。 A・Bに上下や強弱の関係はなく、それぞれ独立して優勝を決める。

> 各ポットから1ペアずつ配るので、どのグループにも若手からベテランまでバランスよく入る。ワールドカップの抽選会と同じ考え方。

ルールは2段構え、そして「必ず終わる」ことを先に確かめる

運営の判断が入る余地をなくし、途中で行き詰まらないことを事前に検証した。

かならず守る

同じチームどうしは同じグループにしない。ひとつのチームから何組か出ている場合は必ず別々のグループへ。リーグ戦でのチーム内対戦を避けるため。

できる範囲で

同じ都道府県はできるだけ散らす。特定の県から多くの方が参加しているため、すべてを分けるのは数の上でむずかしい。こちらは心がけとして扱う。

ぶつかったら　次の入れるグループへ順番にずらす（ワールドカップと同じ運用）。運営の判断は入らず、途中で止まることもない。

100%	0件	0%	全部門
最後まで完成する確率	同じチームの同居	途中で詰まる確率	引き順を全通り試行

連戦を、構造的にゼロにする

5ペアの総当たり10試合を、同じペアが連続しない順に並べ替える。

試合	対戦	試合	対戦
第1	**1 - 2**	第6	**1 - 3**
第2	**3 - 4**	第7	**2 - 4**
第3	**1 - 5**	第8	**3 - 5**
第4	**2 - 3**	第9	**1 - 4**
第5	**4 - 5**	第10	**2 - 5**

	変更前	変更後
連戦	最大2試合（40ペア中24ペア）	**0**
最大空き時間	45分	**30分**

4ペア・6試合では不可能

各ペアが3試合に出るため、鳩の巣原理により必ず連戦が発生する。 実装前に全順列探索で確認し、5ペア編成のときだけ連戦ゼロが成立することを確かめてから組んだ。

> 仕様の実現可能性は、実装前に小規模な全探索で確認する。10要素程度なら数秒で結論が出る。

12チームの変則ブラケットと「敗者審判」

同一グループ同士が早期に当たらないよう、2位と3位を別グループで組ませる。

1位（4組）は準々決勝シード
2位・3位（8組）がplay-inで4枠を争う

```
QF1: A1位 vs [ D2位 vs B3位 ]
QF2: B1位 vs [ C2位 vs A3位 ]
QF3: C1位 vs [ B2位 vs D3位 ]
QF4: D1位 vs [ A2位 vs C3位 ]
```

男子16チームは、5組グループが上位3位、4組グループが上位2位＋3位の成績上位2組。

ラウンド	審判を担当するのは
1回戦	混合・女子＝各プール4位／男子＝C・Dプールの4位・5位
2回戦以降	前ラウンドの敗者
決勝	準決勝で敗れた2チームが主審・線審

> 「前の試合をしたペアが次の審判」というルールも検討したが、総当たりの構造上 **148試合中68試合で成立しない**ことが分かり、実装前に不採用とした。

審判割り当ては、制約充足問題として解く

場当たり的なロジックでは条件を足すたびに破綻した。制約を並べ、スコア関数に落として初めて安定した。

#	制約	種別
C1	審判は同時刻に試合をしていない	ハード
C2	同時刻に同じ組を複数コートへ割り当てない	ハード
C3	部門をまたがない	ハード
C4	コートリーダー・運営スタッフを避ける	ソフト
C5	担当回数を平準化する	ソフト
C6	移動距離が短い（コート番号の近さ）	ソフト

```
def score(candidate, context):
    hard = 1 if (is_leader(c) or is_staff(c)) else 0
    return (hard, load[c], distance(c, court), c)
```

ハードとソフトを分ける

絶対に破れない条件と、優先度でしかない条件を明示的に分けてスコアリングする形へ整理した。「複数条件を満たす割り当て」が出てきたら、最初から制約リストとスコア関数で設計する。

母数が足りない時間帯をどう乗り切るか

8試合が同時に進む時間帯があり、部門ごとに割り当て方式を変える必要があった。

部門	1試合あたり	理由
混合A・B／女子	2組（4名から2名）	空きペアが十分にある
男子	1組（2名）	8試合同時進行の時間帯がある

```
# 男子で1組にした根拠
26組中、8試合同時進行時は16組が試合中 → 空き10組
2組ずつ = 16組必要 → 6組不足
1組ずつ = 8組で足りる → 男子内で完結
```

2コート同時進行グループの借用ルール

男子グループC・D（5組編成）は2コートを同時に使うため、自グループの休み組1組では足りない。

- Cの不足分 → A・Bグループの休み組から借用
- Dの不足分 → E・Fグループの休み組から借用

固定的な供給元を先に決めておくことで、当日の混乱を防いだ。

処理順序で枯渇したバグ　Cを先に処理するとDが使うはずの候補を先取りしてしまう。 各時刻ごとに自グループ分を先に予約する仕組みを入れて解決した。

見つけにくかったバグと、最終的な結果

エラーにならず、候補が見つからないという形でだけ現れるバグだった。

```
# 誤り: play に審判割り当て済みの組を足していた
play[(session, t)] |= assigned_referees

# 正: 実際の出場者だけを持つ play0 を分離
play0 = {k: set(v) for k, v in play.items()}
```

この汚染により、13:35 時点で「試合中」と判定される組が 16組 → 24組に膨れ上がり、条件を満たす候補が消えていた。

検証項目	結果
全148試合の審判割当	完了（充足率100%）
部門をまたぐ借用	0件
同時刻の重複起用	0件
審判が試合中	0件
コートリーダー起用	0件
担当回数	1〜5回（平均3.1回）

17種類のPDFを、常に全部つくり直す

差分更新はやらない。データかロジックを1箇所直して、全帳票を再生成する。

配布用

- 大会プログラム表紙（A4／30部）
- 大会プログラム 29ページ（A4／30部）

掲示・コート設置用

- 掲示用コート別進行表（A3）
- 掲示用トーナメント表（A3）
- コート設置用星取表（A3）

運営用

- 大会運営マニュアル 11ページ
- 開閉会式の進行表とカンペ 10ページ
- コートリーダー用星取表とMatchOrder 18ページ
- 本部席シフト表／コート配置図／コートリーダー一覧／コートマップ

技術スタック

- Python 3.12 + pandas
- HTML+CSS → WeasyPrint → PDF
- 作図はインラインSVG（座標をPythonで計算）
- データはCSV＋JSON

CSS Paged Mediaに対応しており、@pageでのA3/A4指定やページ分割制御が既存のHTML/CSS知識で書けることが決め手。

> 40回以上の仕様変更に耐えられたのは、「常に全生成」を前提にしたから。差分更新は整合性バグの温床になる。

刷って、貼って、書き込まれるところまで設計する

PDFは画面で完成ではない。印刷所に渡り、モノクロで掲示され、鉛筆で書き込まれる。

ファイル名に印刷指示を埋める

```
【A4縦_30部_印刷】大会プログラム_V1.pdf
【A3横_各1部_印刷】コート設置用星取表_V1.pdf
```

受け渡し時に、口頭説明なしで意図が伝わる。

モノクロ印刷への対応

- 記入用（トーナメント表・星取表）＝記入欄を完全に白、枠線は黒で統一
- 閲覧用（コート別進行表）＝部門を白／グレーの濃淡で区別

色違いの背景は、モノクロだと同系グレーになって判別できなくなる。

出力を数値で検証する

```
# 枠の重なり検出
print(f'ペア内隙間: {SIN - BH}pt')
# 負なら重なり
```

テキスト抽出で旧データの残存を確認し、ラスタライズして目視。SVGの座標ミスは目では気づけない。

環境依存文字　参加者名に異体字（U+5721）を含む姓があった。表示は正しく出たが、PDFのテキスト抽出では通常字体として取り出されるため、grep等での検証には注意が必要。

当日システムは、やらないことを決めてから作った

「審判が直接入力」を諦めたから、5日で作れた。

今回やらなかった

審判がスマホで直接入力する。参加者層のリテラシー差が大きく、一律の運用にできないと判断した。

今回やった

本部入力の効率化に集中する。入力 → 承認 → 自動集計 → 自動公開を、一本のフローにした。

フェーズ	やること	手段
今回	本部が入力・承認する	GAS + スプレッドシート ／ 5日で構築
次回以降	審判が直接入力する	申込システムと連携し、アカウント基盤をつくる

> スコープを削ることが、最大の設計判断だった。制約を認めることが前に進む条件になる。

入力と承認を分け、表示は切り離す

「ダブルチェックはどうするの」という現場の一言が、そのまま仕様になった。

承認待ち
順位表に出ない

▶

承認済
順位表へ反映

▶

差戻
再入力できる

> 画面もURLで分けた（/exec が入力、?mode=approve が承認）。 GASでは厳密な権限管理ができないため、**URLを知る人だけが承認できる**という運用上の分離に留めた。

入力系はGAS、表示系は静的HTML

- GAS + Sheets ＝ 入力・承認・データ保持、?mode=api でJSONを返す
- 静的HTML（Netlify）＝ 順位表・星取表・トーナメント表・試合順・審判担当、ペア名で検索
- 60秒ごとに自動更新。ビルド不要で、ファイルを置くだけ

ハマったところと、触ってもらって直したUI

いずれもエラーメッセージからは原因が分からず、切り分けに時間を取られた。

GASでハマった4点

試合番号は計算で出せない
対応が部門ごとに違い、紙のプログラムをそのままシート化して解決

NaN を返せない
NaN・undefined・Date が混ざるとクライアントには null が渡る

"9:00" が 1899年の Date になる
書き込み時に文字列書式、読み取り時にも正規化する二段構え

デプロイしないと反映されない
当日いちばん事故りやすい

自動遷移をやめた

得点を選ぶと自動で次のセットへ進む設計だったが、「7を押したあと誤って3を押すと確定してしまう」。明示的な確定ボタンに変更。

「戻る」の行き先を書いた

一覧に戻ると誤解されるため、状況に応じて「選んだ数字を消す」「セット1を取り消す」と文言を変えた。

押せない理由を見せた

ボタンを消さず、進捗バーで「承認済 10 / 40 試合（残り30）」と残数を表示した。

タップ数は増えるが、訂正のコストの方が高い。

運用結果

転記作業がなくなり、本部席を離れられるようになった。

101	148	37	17	100%	5
参加者	予選試合	決勝T 試合	生成したPDFの種類	審判割り当ての充足率	当日システムの開発日数

転記作業がなくなった

入力すると、順位表とトーナメント表が自動で更新される

本部を離れられるようになった

溜まった紙をまとめて処理でき、常駐が不要に

参加者が自分で確認できる

順位・試合順・審判担当をスマホから見られる

順位の理由を説明できる

星取表とワイルドカード順位を公開し、問い合わせを減らした

良かった点：判断が当たったところ

うまくいったのは、機能を足したからではなく、先に決めておいたことによる。

01 手作業の再現性を消した

抽選・スケジュール・審判・帳票をひとつのデータから生成する形にしたため、 **40回以上の仕様変更**を、全帳票の整合性を崩さずに吸収できた。

02 当日の運営負荷が実際に下がった

転記作業がなくなり、本部席への常駐が不要に。運営7名が全員選手兼任という条件のまま、 大会を止めずに回せた。

03 公平性を「説明できる」状態にした

ポット方式と事前検証の結果を抽選前に参加者へ公開し、当日は星取表とワイルドカード順位も公開。 決め方への問い合わせを減らせた。

04 詰まる可能性を先に潰した

連戦ゼロ・審判充足100%・抽選が途中で行き詰まる確率0%を、いずれも実装前後の検証で確認してから当日を迎えられた。

> **最大の勝因はスコープ判断**　「審判が直接入力」を切ったことで、当日システムを5日で立ち上げられた。 機能を足す判断より、削る判断のほうが効いた。

反省点：妥協と、後半に寄った負荷

動いたが、仕組みとして残らなかったもの・終盤に集中したものがある。

設計上の妥協

権限管理をURL分離で代替した
GASの制約とはいえ、承認権限が運用依存のまま残った

マスタをコード内の定数で持った
チーム登録・試合割当の変更にコード修正が必要

反映は60秒ポーリング止まり
承認と同時には見えず、問い合わせの余地が残る

入力の起点が本部のまま
会場にWi-Fiがなく、オフライン前提の設計に踏み込めなかった

進め方の反省

現場レビューが後半に寄った
ブザー時の処理や所要時間の実測など、仕様の核心が終盤に判明した

重いバグが実装終盤に出た
出場者判定の汚染・処理順序による枯渇。いずれもエラーにならず切り分けに時間を取られた

印刷・文字の問題は出力後に発覚
モノクロでの判別不能、異体字の扱い。刷ってから気づく類のもの

スタッフの試合と役割の競合
決勝トーナメント以降のシフトは読めず、当日調整に委ねた

改善点：次回、この順で解く

上から順に、解くと下の問題が自動的に軽くなる並びにしてある。

#	やること	解ける問題
1	**申込時にアカウントを作る**	審判の直接入力と権限管理が同時に解ける。承認者・入力者を記録として残せる
2	**スコア入力をオフラインで動かす**	会場にWi-Fiがない前提へ。ローカルHTML（IndexedDB）か、帳票生成部を流用した入力UIを検討
3	**マスタをデータに出す**	チーム登録・試合割当をシート／CSVへ。定数のコード修正をなくす
4	**現場レビューを設計初期に複数回**	運営メンバーの一言がそのまま仕様になる。早く聞くほど再生成の回数が減る
5	**出力の自動検証を標準手順にする**	座標検証・テキスト抽出・ラスタライズ・モノクロ確認を、生成のたびに必ず通す
6	**承認と同時に反映する**	ポーリングをやめ、順位表の遅延をなくす（優先度は上の5つより低い）

> 1つ目を解けば、2〜3は設計の前提として自然に片づく。次期システムの最優先はアカウント基盤。

残した負債と、引き継ぐ設計

今回の妥協点を、次に解くべき順に並べる。

残した負債

権限管理
URL分離という運用上の回避策に留めた

マスタ管理
チーム登録・試合割当をコード内の定数で持った

リアルタイム性
60秒ポーリング。承認と同時には反映されない

スコア入力の起点
会場にWi-Fiがなく、オフライン動作が前提になる

引き継ぐ設計

承認フロー
入力と承認を分け、承認済みのみ集計に使う

マスタとトランザクションの分離
構成・スケジュールと、結果を分けて持つ

APIファースト
表示を疎結合にしたことで作り込めた

単一データソース＋全生成
1箇所直して、全部つくり直す

> 最優先はアカウント基盤。申込時にアカウントを作れば、審判の直接入力も権限管理も自然に解ける。

TAKEAWAYS

この事例から持ち帰れること

01　スコープを削ることが、最大の設計判断

「審判が直接入力」を諦めたから5日で作れた。制約を認めることが、前に進む条件だった

02　現場の指摘が、そのまま仕様になる

承認フローも確定ボタンも、運営メンバーの一言から生まれた。ブザー時の処理やラジオ体操の所要時間も実測で直した

03　できないことは、実装前に確かめられる

連戦ゼロも相互審判も、小規模な全探索で可否が出た。10要素程度なら数秒で結論が出る

04　壊れたときにどうするかまで設計する

シートを直接編集できる、公開ページが落ちても予備画面がある。逃げ道が運用の安心につながる