

ソースファイル「20251118__風の村管理本部総務部給与.m4a」の内容に基づき、総務部給与担当者へのヒアリング詳細メモを作成しました。
このヒアリングでは、給与計算の実務における「手作業」の多さと、それを生み出す「複雑な規定」および「システム機能の限界」が具体的に語られています。

ヒアリング詳細メモ: 総務部 給与担当

日時: 2025年11月18日 対象: 総務部 給与担当職員 インタビュアー: 上田 (コンサルタント)

1. 業務の全体像とスケジュール感

- 業務サイクル:
- 毎月5日: 現場(事業所)での勤怠締め切り。
- 5日~: 勤怠データのチェック、修正、確定。
- 15~16日頃: 給与計算処理のピーク(振込3日前までに確定が必要)。
- 20日: 給与支給日。同日、経理へデータを渡す。
- 翌月25日: 全社の月次決算確定(経理側)。
- 使用システム:
- 施設系勤怠: 「金次郎」
- 訪問系勤怠・給与: 「NDソフトウェア」
- 本部給与計算: 「MJS(ミロク情報サービス/魅力)」
- これらが連携しておらず、データ連携(CSV取込や転記)が発生している 1。

2. 抱えている具体的な課題(ボトルネック)

① システムで計算しきれない「手当」の迷宮(手作業の温床)

給与規定や手当のルールが複雑すぎて、パッケージソフト(MJS)の標準機能では対応できず、**「手計算」して「強制入力」**しているケースが多発している。

- 「早遅(はやおそ)手当」の問題: 2, 4
- 早番と遅番を通して勤務する(中抜け時間が長い)シフトに付く手当。
- 急な欠勤対応などでシフトパターンを変更しても、システム上のマスタ設定(パターン)を変えられない(変行すると手当がつかなくなる等の制約がある)ため、手入力で手当を付与している。
- 「宿直・夜勤手当」の単価違い: 5, 6
- 事業種別(介護、児童用語、障害など)によって、同じ「宿直」「夜勤」でも手当の単価が異なる。
- システムには1つの単価しか設定できないため、単価が異なる事業所(例: 児童養護)の分は、差額を手計算して入力している。
- 法定休日・割増計算の複雑さ: 6, 7
- 「16時間以上の夜勤」と「13時間以下の夜勤」でルールが違う。
- 法定休日の夜勤など、複雑な割増率(0.35など)がかかるケースはシステムで自動計算されず、データを抽出して手計算し、強制転記している。

②「人件費安分」による業務負荷と決算遅延 8-10

- 兼務の多さ: 1人の職員が複数の事業所(例: 介護ステーションと定期巡回など)を兼務することが多く、最大で20箇所以上に費用を安分(配賦)する必要がある。
- 変動する安分率: 安分率は固定ではなく、その月の勤務実績(時間)によって毎月変動するため、給与計算が完全に固まらなると経費が確定せず、これが月次決算の遅れ(翌月25日)の主因となっている。

③ 現場(事業所)とのコミュニケーションとガバナンス不全

- 締め切りが守られない: 毎月5日の締め切りを守らない事業所、全く締めていない所長が存在する 11, 12。
- 修正の嵐: 締め切り後も「有給の入力忘れ」「勤怠間違い」などの修正依頼が給与確定直前(あるいは確定後)まで続き、手戻りが発生している 13, 14。
- 問い合わせ対応: 現場からは「年末調整の書き方」「システムの操作方法」「就業規則の確認」など、マニュアルや掲示板(Kintone)を見ればわかる内容の電話問い合わせが多い 15, 16。

④ その他: 形骸化したマニュアル「風グラム」 17-20

- 過去に作成した業務マニュアル「風グラム」は、「お茶の出し方」レベルまで細分化しすぎた(大項目・中項目・小項目と分け、小項目だけで数千個に及ぶ)ため、メンテナンス不能に陥り、現在は誰も使っていない。
- 結果、職員は個人のメモや「頭の中」にルールを持っており、属人化している。

3. 解決に向けた示唆(AI活用の可能性など)

- 問い合わせ対応へのAI導入:
- 就業規則やマニュアル、過去のQ&AをAIに学習させ、チャットボットで一次対応させることで、電話対応を削減できる可能性が高い 21, 22。
- 「処遇改善加算」計算の自動化:
- 毎月、国への報告のために複雑な計算(誰にいくら配分したか等)を行っているが、これもロジックが決まっているため、自動化の余地がある 23-25。
- 知識の体系化:
- 「風グラム」の失敗を教訓に、AIを活用して散在する情報を整理し、検索可能な「知識体系」として再構築する必要がある 26-28。

4. インタビュアー(上田氏)のコメント・視点

- 給与計算の複雑さは「過去の経緯(M&Aや事業拡大)」と「制度の複雑さ」に起因するが、「個人の頭の中にあるルール(知識)」をシステム(AIやルールエンジン)に外出しできれば、属人化は解消できる。
- 現状の「現場入力→本部修正→システム入力」という多重入力・バケツリレー方式をやめ、一気通貫のデータフローを設計すべきである。