

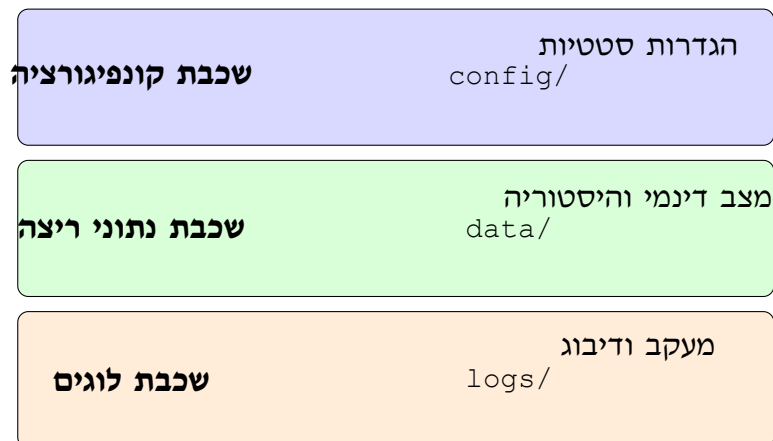
9 פרוטוקול נתוני הליגה

9.1 מבוא: הקוד הגנטי של חברת הסוכנים

כאשר אנו בונים חברה של סוכנים אוטונומיים – שחקנים, שופטים ומנהלי ליגה – אנו למעשה יוצרים תרבות דיגיטלית חדשה. כמו בכל חברה אנושית, גם כאן נדרשים שלושה יסודות קריטיים:

1. **חוקים משותפים** – הפרוטוקול שהגדרנו בפרקים הקודמים.
 2. **זיכרון קולקטיבי** – היכולת לשמור ולשחזר מידע היסטורי.
 3. **קוד גנטי** – הקונפיגורציה שמגדירה את ה-DNA של כל סוכן.
- נספח זה מתאר את "בסיס הנתונים על קובצי JSON" – ארכיטקטורת שלוש שכבות שמאפשרת למערכת לצמוח לקנה מידה של אלפי סוכנים וליגות.

9.2 ארכיטקטורת שלוש השכבות



9.2.1 עקרונות מנחים

כל קובץ במערכת עומד בעקרונות הבאים:

- **מזהה ייחודי (id):** כל אובייקט ראשי מקבל מזהה חד-ערכי.
- **גרסת סכמה (schema_version):** מאפשר מיגרציות עתידיות.
- **חיתמת זמן (last_updated):** בפורמט UTC/ISO-8601.
- **תאימות לפרוטוקול:** כל השדות תואמים ל-league.v2.

9.3 שכבת הקונפיגורציה – config/

שכבה זו מכילה את ה"קוד הגנטי" של המערכת – הגדרות סטטיות שנקראות בעליית הסוכנים.

9.3.1 קובץ מערכת גלובלי - config/system.json

- **מטרה:** פרמטרים גלובליים לכל המערכת.
 - **משתמשים:** כל הסוכנים, Orchestrator עליון.
 - **מיקום:** SHARED/config/system.json
- קובץ זה מגדיר את ערכי ברירת המחדל עבור:
- הגדרות רשת (network) – פורטים וכתובות.
 - הגדרות אבטחה (security) – טוקנים ו-TTL.
 - זמני המתנה (timeouts) – תואמים להגדרות הפרוטוקול בפרק 2.
 - מדיניות ניסיון חוזר (retry_policy) – תואמת להגדרות הפרוטוקול.

דוגמה: מבנה system.json (קטע)

```
{
  "schema_version": "1.0.0",
  "system_id": "league_system_prod",
  "protocol_version": "league.v2",
  "timeouts": {
    "move_timeout_sec": 30,
    "generic_response_timeout_sec": 10
  },
  "retry_policy": {
    "max_retries": 3,
    "backoff_strategy": "exponential"
  }
}
```

9.3.2 רישום סוכנים - config/agents/agents_config.json

- **מטרה:** ניהול מרכזי של אלפי סוכנים.
 - **משתמשים:** מנהל הליגה, כלי Deployment.
 - **מיקום:** SHARED/config/agents/agents_config.json
- קובץ זה מכיל את "ספר האזרחים" של חברת הסוכנים:
- league_manager – פרטי מנהל הליגה.
 - referees[] – רשימת כל השופטים הרשומים.
 - players[] – רשימת כל השחקנים הרשומים.

9.3.3 קונפיגורציית ליגה – config/leagues/<league_id>.json

- **מטרה:** הגדרות ספציפיות לליגה.
 - **משתמשים:** מנהל הליגה, שופטים.
 - **מיקום:** SHARED/config/leagues/league_2025_even_odd.json
- כל ליגה היא "מדינה" עצמאית עם חוקים משלה:

דוגמה: קונפיגורציית ליגה (קטע)

```
{
  "league_id": "league_2025_even_odd",
  "game_type": "even_odd",
  "status": "ACTIVE",
  "scoring": {
    "win_points": 3,
    "draw_points": 1,
    "loss_points": 0
  },
  "participants": {
    "min_players": 2,
    "max_players": 10000
  }
}
```

9.3.4 רישום סוגי משחקים – config/games/games_registry.json

- **מטרה:** רישום כל סוגי המשחקים הנתמכים.
 - **משתמשים:** שופטים (לטעינת מודול חוקים), מנהל ליגה.
 - **מיקום:** SHARED/config/games/games_registry.json
- המערכת תומכת בסוגי משחקים מרובים. כל משחק מגדיר:
- game_type – מזהה ייחודי.
 - rules_module – מודול החוקים לטעינה.
 - max_round_time_sec – זמן מקסימלי לסיבוב.

9.3.5 ברירות מחדל לסוכנים – config/defaults/

- **מטרה:** ערכי ברירת מחדל לפי סוג סוכן.
 - **קבצים:** referee.json, player.json
 - **מיקום:** SHARED/config/defaults/
- קבצים אלה מאפשרים לסוכן חדש להתחיל לפעול עם הגדרות סבירות מבלי להגדיר כל פרמטר בנפרד.

9.4 שכבת נתוני הריצה – data/

אם שכבת הקונפיגורציה היא ה"קוד הגנטי", שכבת נתוני הריצה היא ה"זיכרון ההיסטורי" של החברה. כאן נשמרים כל האירועים שקורים במערכת.

9.4.1 טבלת דירוג – data/leagues/<league_id>/standings.json

- **מטרה:** מצב הדירוג העדכני של הליגה.
- **מעדכן:** מנהל הליגה (אחרי MATCH_RESULT_REPORT).
- **מיקום:** SHARED/data/leagues/league_2025_even_odd/standings.json

דוגמה: מבנה טבלת דירוג

```
{
  "schema_version": "1.0.0",
  "league_id": "league_2025_even_odd",
  "version": 12,
  "rounds_completed": 3,
  "standings": [
    {
      "rank": 1,
      "player_id": "P01",
      "display_name": "Agent Alpha",
      "wins": 4, "draws": 1, "losses": 1,
      "points": 13
    }
  ]
}
```

9.4.2 היסטוריית מחזורים – data/leagues/<league_id>/rounds.json

- **מטרה:** תיעוד כל המחזורים שהתקיימו.
- **מעדכן:** מנהל הליגה (אחרי ROUND_COMPLETED).
- **מיקום:** SHARED/data/leagues/league_2025_even_odd/rounds.json

9.4.3 נתוני משחק בודד – data/matches/<league_id>/<match_id>.json

- **מטרה:** תיעוד מלא של משחק בודד.
- **מעדכן:** השופט שניהל את המשחק.
- **מיקום:** SHARED/data/matches/league_2025_even_odd/R1M1.json

קובץ זה הוא ה"תעודת זהות" של המשחק ומכיל:

- lifecycle – מצב המשחק וזמנים.

- transcript[] – כל ההודעות שהוחלפו (היסטוריית מהלכים).
- result – התוצאה הסופית (תואם ל-GAME_OVER).

9.4.4 היסטוריית שחקן – data/players/<player_id>/history.json

- **מטרה:** "זיכרון אישי" של השחקן.
 - **משתמש:** השחקן עצמו (לבניית אסטרטגיה).
 - **מיקום:** SHARED/data/players/P01/history.json
- שחקן חכם יכול להשתמש בקובץ זה כ"זיכרון" לשיפור האסטרטגיה שלו:

דוגמה: היסטוריית שחקן

```
{
  "player_id": "P01",
  "stats": {
    "total_matches": 20,
    "wins": 12, "losses": 5, "draws": 3
  },
  "matches": [
    {
      "match_id": "R1M1",
      "opponent_id": "P02",
      "result": "WIN",
      "my_choice": "even",
      "opponent_choice": "odd"
    }
  ]
}
```

9.5 שכבת הלוגים – logs/

שכבה זו היא "מערכת העצבים" של החברה – היא מאפשרת לנו לראות מה באמת קורה במערכת המבוזרת.

9.5.1 לוג ליגה מרכזי – logs/league/<league_id>/league.log.jsonl

- **פורמט:** JSON Lines (כל שורה אובייקט JSON נפרד).
- **משתמשים:** DevOps, תמיכה טכנית.
- **מיקום:** SHARED/logs/league/league_2025_even_odd/league.log.jsonl

דוגמה: רשומת לוג ליגה

```
{
  "timestamp": "2025-01-15T10:15:00Z",
  "component": "league_manager",
  "event_type": "ROUND_ANNOUNCEMENT_SENT",
  "level": "INFO",
  "details": {"round_id": 1, "matches_count": 2}
}
```

9.5.2 לוג סוכן – logs/agents/<agent_id>.log.jsonl

- **מטרה:** מעקב פר-סוכן לדיבוג.
 - **משתמשים:** מפתחי הסוכן.
 - **מיקום:** SHARED/logs/agents/P01.log.jsonl
- כל סוכן מתעד את ההודעות שהוא שולח ומקבל, מה שמאפשר End-to-End Trace של כל אינטראקציה במערכת.

9.6 טבלת סיכום קבצים

9.7 שימוש בקבצים המשותפים

כל קובצי הדוגמה המתוארים בנספח זה זמינים בתיקיה המשותפת:

L07/SHARED/

סטודנטים מוזמנים להשתמש בקבצים אלה כבסיס למימוש הסוכנים שלהם. הקבצים כוללים:

- דוגמאות מלאות לכל סוג קובץ.
- נתונים תואמים לפרוטוקול league.v2.
- מבנה תיקיות מומלץ לפרויקט.

טבלה 18: סיכום קובצי הקונפיגורציה והנתונים

שכבה	נתיב	מטרה	משתמש
קונפיג	config/system.json	פרמטרים גלובליים	כל הסוכנים
קונפיג	config/agents/	רישום סוכנים	מנהל ליגה
קונפיג	config/leagues/	הגדרות ליגה	מנהל ליגה
קונפיג	config/games/	רישום משחקים	שופטים
קונפיג	config/defaults/	ברירות מחדל	סוכנים
ריצה	data/.../stand-ings.json	טבלת דירוג	כולם
ריצה	data/.../rounds.json	היסטוריית מחזורים	מנהל ליגה
ריצה	data/matches/	פרטי משחק	אנליטיקה
ריצה	data/players/	היסטוריה אישית	שחקן
לוגים	logs/league/	לוג מרכזי	DevOps
לוגים	logs/agents/	לוג סוכן	מפתחים

9.8 סיכום

ארכיטקטורת שלוש השכבות שהצגנו – קונפיגורציה, נתוני ריצה, ולוגים – מספקת את התשתית הנדרשת לבניית מערכת סוכנים בקנה מידה גדול.

כמו בחברה אנושית, גם כאן:

- **הקונפיגורציה** היא ה"חוקה" – הכללים היסודיים שכולם מכירים.

- **נתוני הריצה** הם ה"ארכיון ההיסטורי" – הזיכרון הקולקטיבי.

- **הלוגים** הם ה"עיתונות" – תיעוד מה שקורה בזמן אמת.

מבנה זה מכין את המערכת לצמיחה של אלפי סוכנים וליגות, תוך שמירה על סדר, עקביות, ויכולת מעקב.