

```
{  
[  
    ..  
]  
}  
[]
```

```
def calculate_trajectories():
```

# ІНФОРМАЦІЙНО-АНАЛІТИЧНІ СИСТЕМИ В ОПЕРАТИВНОМУ ПЛАNUВАННІ

Архітектура даних та алгоритмічні стратегії пошуку

```
class IntelParser:  
    data_stream = json.loads(response)  
    for coordinate in grid_points:
```

```
[]
```

```
}
```

```
def calculate_grid_adv():  
    data_stream = json.loads(response)
```

```
]
```

Від паперових карт до цифрової переваги

# Інформація — це зброя, а не лише забезпечення

## ВИКЛИК

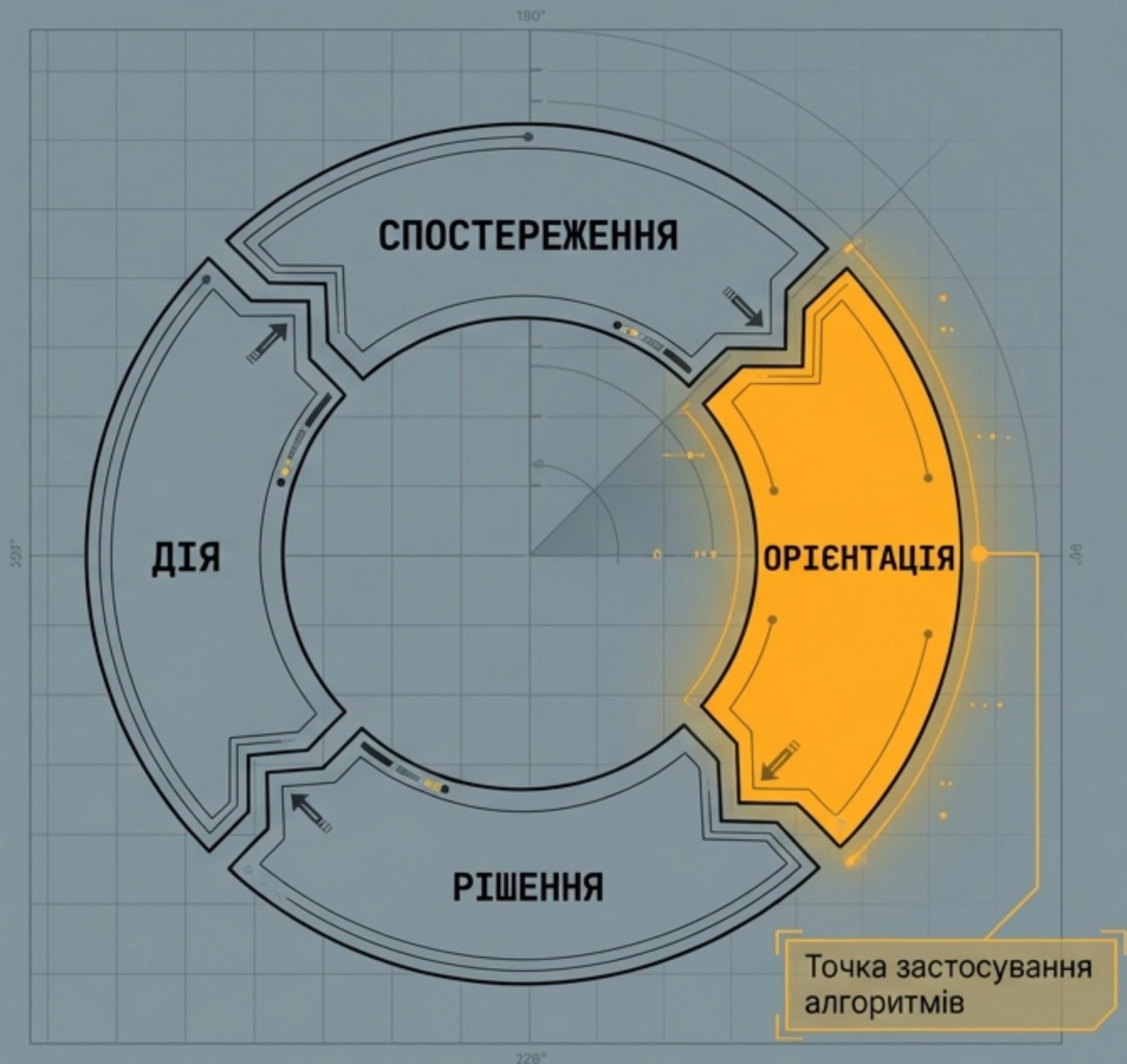
Обсяг вхідних повідомлень (розвідка, логістика, сенсори) перевищує когнітивні можливості людини.

## МЕТА

Автоматизація циклу **OODA**. Перемагає той, хто швидше проходить етап «Орієнтація».

## РІШЕННЯ

**IAC** (Інформаційно-аналітичні системи) як “цифровий скелет” для Загальної операційної картини (**COP**).



# Типологія військових даних: від порядку до хаосу

## Структуровані дані



Чітка схема (SQL).  
Фундамент сумісності.

- Геокоординати (MGRS)
- Часові мітки (DTG)
- NSN коди

## Напівструктуроні дані



Гнучкість для динамічних структур.

- JSON
- XML
- GeoJSON
- ORBAT (Бойовий порядок)

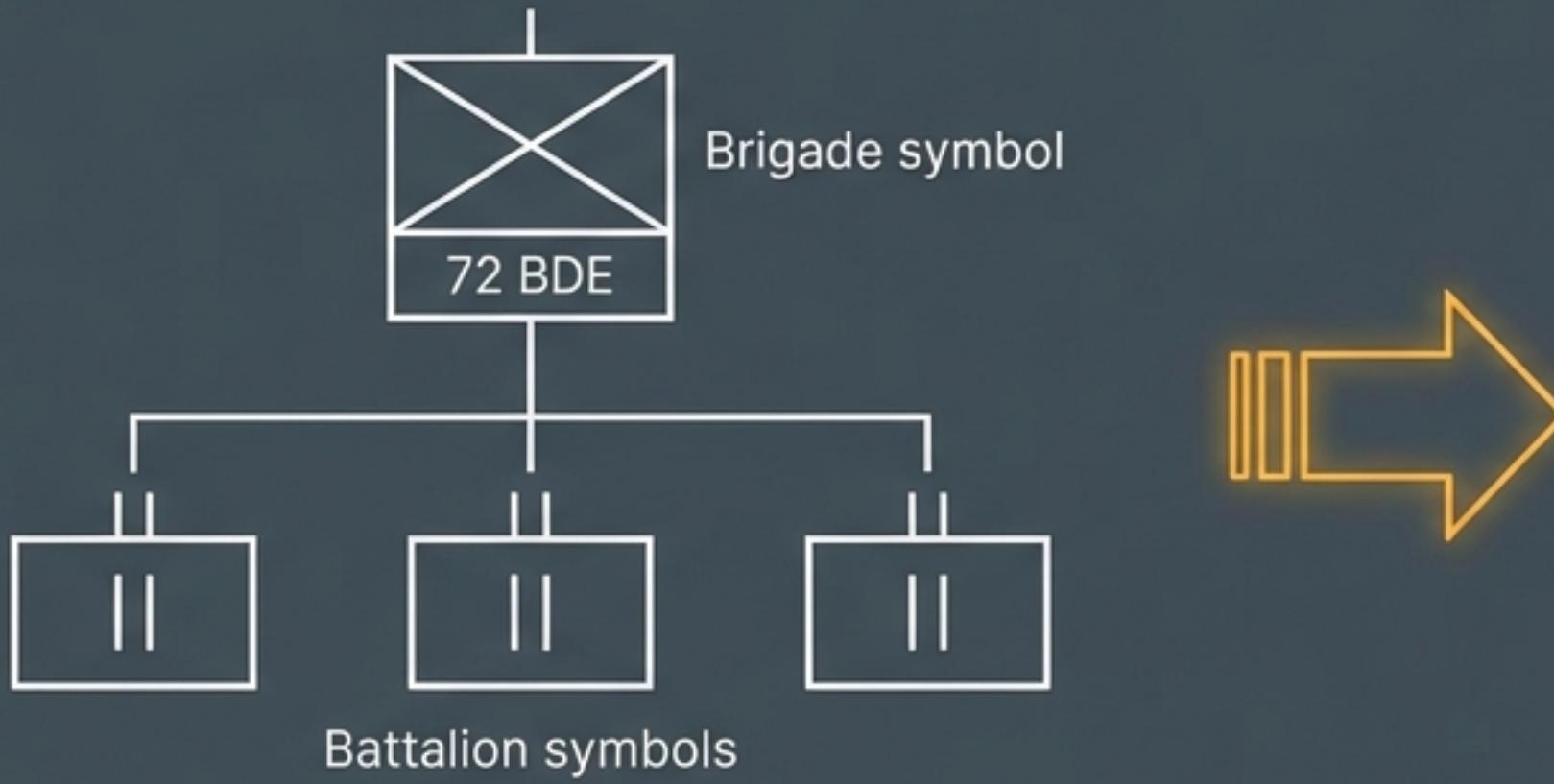
## Неструктуровані дані



80% всієї розвідінформації.  
“Океан інформації”.

- SITREP (текст)
- Радіоперехоплення
- Відео з БПЛА

# Моделювання сил: ORBAT як код

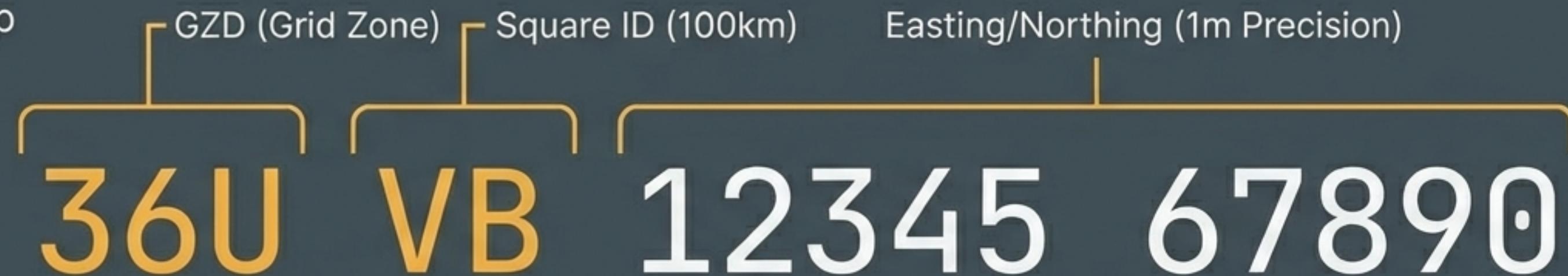


```
orbat_data = {  
    "unit_id": "UA-72-BDE",  
    "name": "72-га окрема механізована бригада",  
    "location": {  
        "mgrs": "36U VB 1234 5678"  
    },  
    "subunits": [...]  
}
```

**Insight:** JSON дозволяє моделювати глибоку вкладеність і динамічно додавати нові параметри без перебудови бази даних.

# Кодування простору: MGRS та GeoJSON

MGRS Розбір



GeoJSON Structure

```
{  
  "type": "Feature",  
  "geometry": {  
    "type": "Point",  
    "coordinates": [37.550, 48.450]  
  },  
  "properties": { "name": "Unit Alpha" }  
}
```



Валідація даних критична  
для вогневої підтримки.

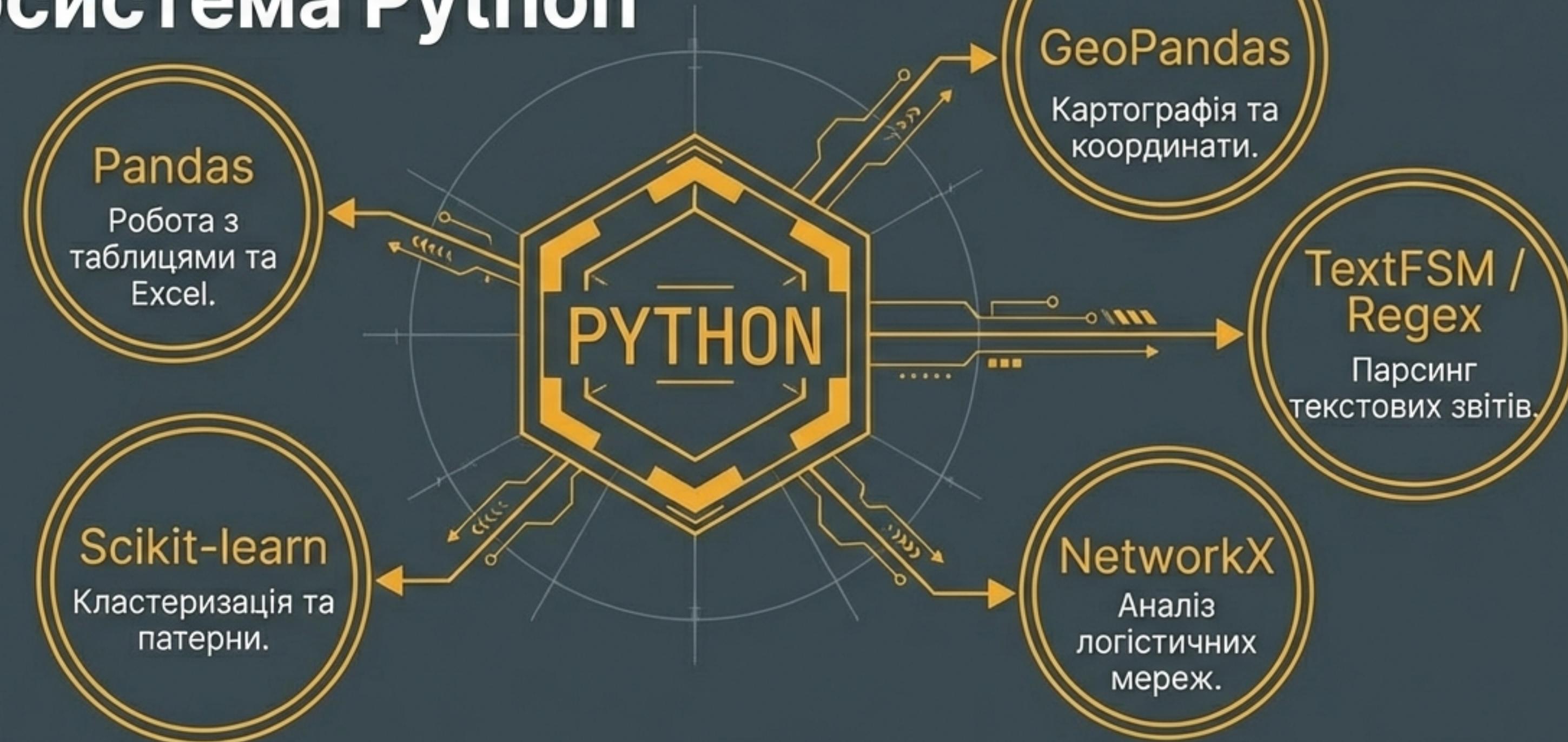


# Логістика як дані: Класи постачання NATO

Клас	Опис	Структура даних (Python Dict)
Class I	Продовольство	{'type': 'MRE', 'qty': 5000}
Class III	Паливо	{'type': 'Diesel', 'qty': 20000, 'unit': 'liters'}
Class V	Боєприпаси	{'type': '155mm HE', 'qty': 300, 'fuzes': 300}
Class IX	Запчастини	{'nsn': '2530-01-123-4567', 'qty': 10}

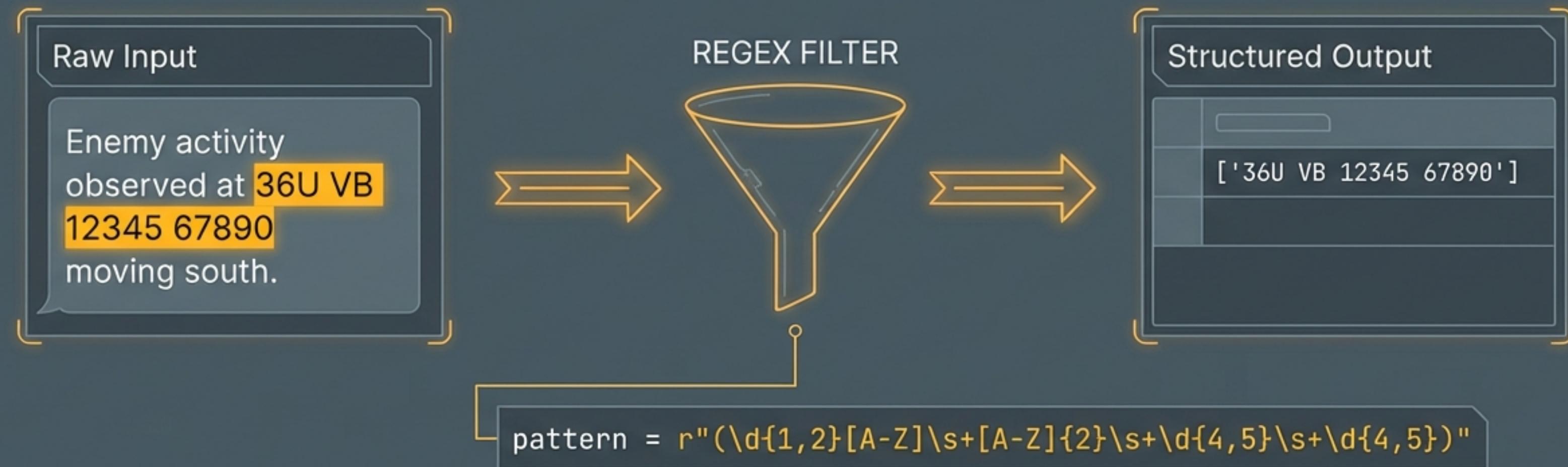
Структурування дозволяє агрегувати потреби та застосовувати алгоритмії оптимізації потоків.

# Арсенал аналітика: Екосистема Python





# Обробка тексту: від SITREP до бази даних



Автоматичне витягування координат з тисяч повідомень Telegram.



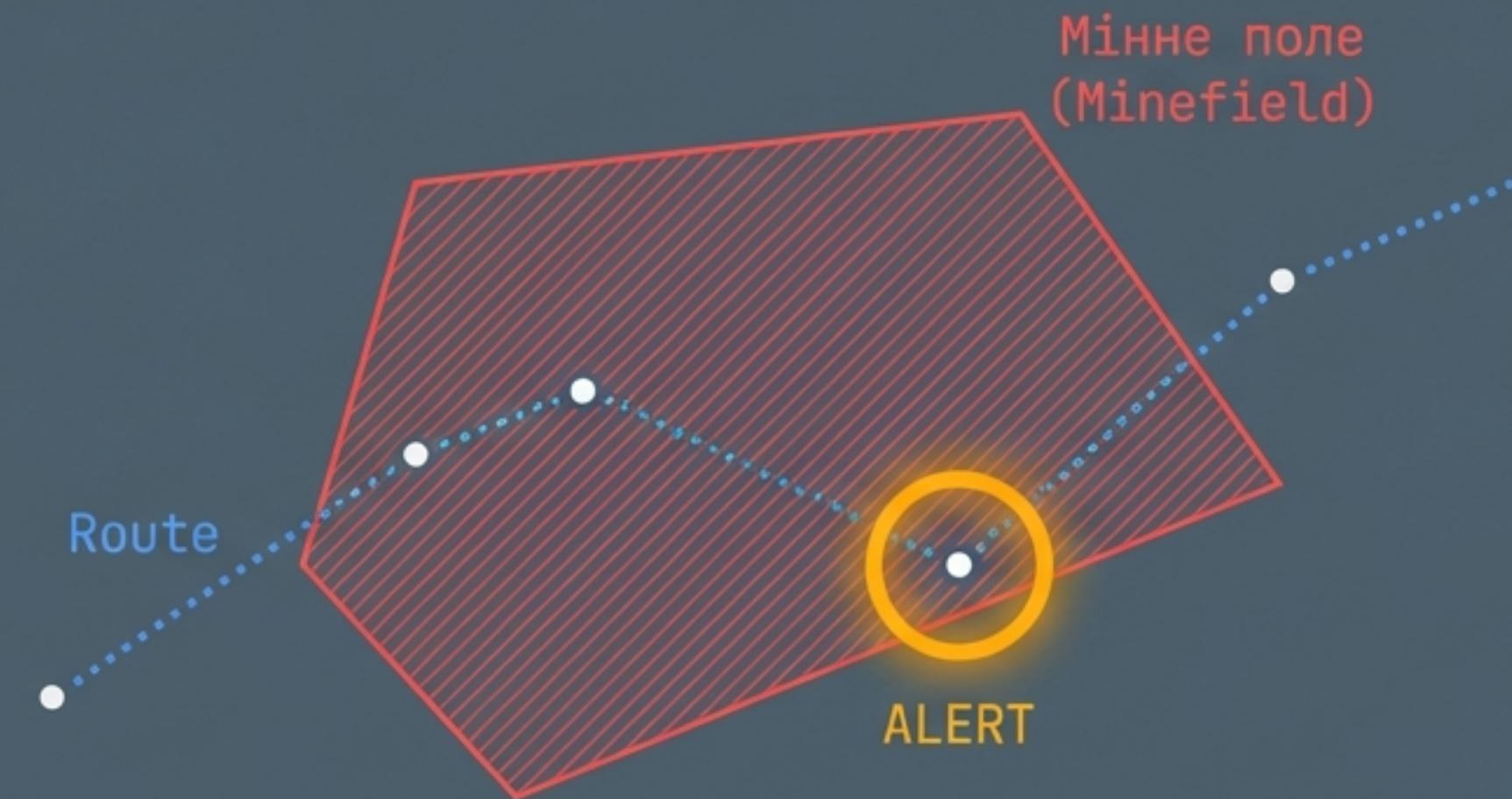
# Нечіткий пошук (Fuzzy Search): Подолання людського фактора



Алгоритм: Відстань Левенштейна.  
Вирішує проблему  
транслітерації (Kyiv/Kiev) та  
одруків у радіожурналах.  
Бібліотека: `thefuzz`.



# Геопросторовий аналіз: Алгоритм Point-in-Polygon

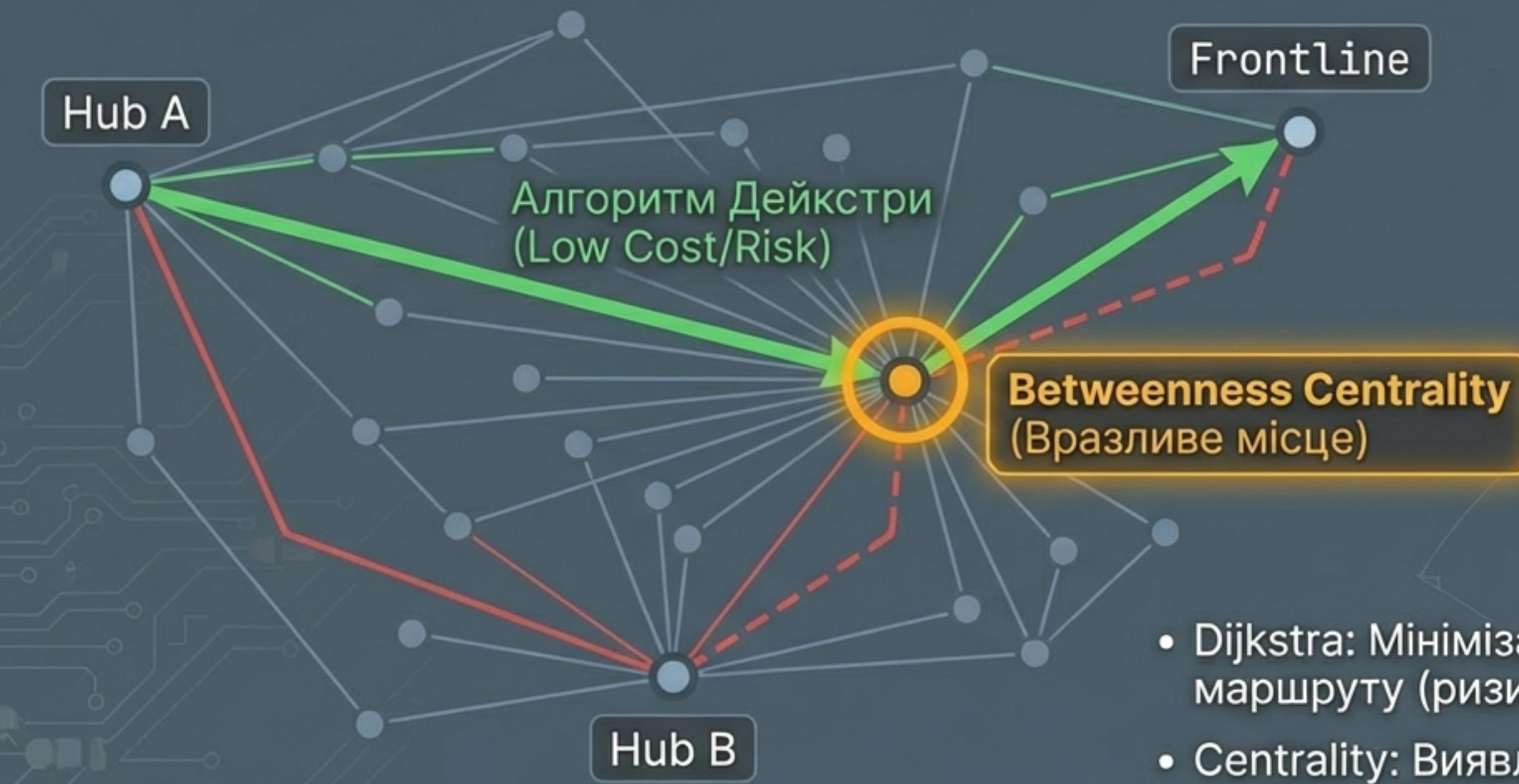


```
danger_mask = patrols.within(minefield)
if not patrols_in_danger.empty:
    trigger_alarm()
```

Автоматична перевірка безпеки маршрутів.



# Графові алгоритми: Логістика та мережева війна





# Семантичний пошук: Векторний аналіз сенсу



Запит знаходить 'танки', навіть якщо слово 'бронетехніка' відсутнє.



# ChatGPT як асистент: Генерація даних



Конвертуй цей текстовий ORBAT у вкладений JSON. Додай поля 'location' та 'commander'.



```
{  
    "unit_designation": "1-ва БТГр",  
    "echelon": "Баталійне тактична група",  
    "tecation": "п.п. Аерізан, Донецька обл.",  
    "commander": "Підполковник Іван Петренко",  
    "subordinate_units": [  
        {  
            "unit_designation": "1-ша ТР",  
            "echelon": "Танкове рога",  
            "tecation": "Окремій Аерізан",  
            "commander": "Капітан Олексій Сидоренко"  
        },  
        {  
            "unit_designation": "2-га НР",  
            "echelon": "Механізоване рога",  
            "location": "Примислове селище",  
            "commander": "Старший підстепант Дондре Коєане"  
        }  
    ]  
}
```



Очисти цей SITREP та витягни координати MGRS у список.



```
# Report re  
sitrep_text = """  
SITREP 232S60Z OCT 24  
...  
Сностереоено рух ворожої бронетехніки біля 360UT 1234 5678.  
Артилерійський обстріл по наявних позиціях в районі 360UT 9876 5432.  
...  
# Регулярний вираз для MGRS (поумка, може вимагати уточнення)  
mgrs_pattern = r'\b(\d{2})[A-Z](\d{2})\w{2}(\d{2})\w{2}(\d{2})\w{2}'  
  
mgrs_coordinates = re.findall(mgrs_pattern, sitrep_text)  
  
print('Виведені координати MGRS: ')  
for coord in mgrs_coordinates:  
    print(coord)  
  
# Вивід:  
# ['360UT 1234 5678', '360UT 9876 5432']
```



# ChatGPT як асистент: Алгоритмічні рішення



Напиши скрипт на Python використовуючи `scikit-learn` для кластеризації (K-Means) точок падіння снарядів. Мета: виявити позиції артилерії.



```
from sklearn.cluster import KMeans
import numpy as np

# Приклад даних: [широта, довгота]
X = np.array([[48.12, 37.55], [48.14, 37.57],
              [48.13, 37.56], [48.20, 37.60],
              [48.21, 37.61], [48.22, 37.62]])

# Ініціалізація KMeans (припустимо 2 кластери)
kmeans = KMeans(n_clusters=2,
                 random_state=0, n_init=10)

# Навчання подвій
kmeans.fit(X)

# Отримання центрів кластерів (потенційні позиції)
cluster_centers = kmeans.cluster_centers_

print("Центри кластерів (потенційні позиції):")
print(cluster_centers)
```

Аналітик формулює логіку задачі, AI генерує синтаксис коду.



# Інформаційна перевага (Information Dominance)



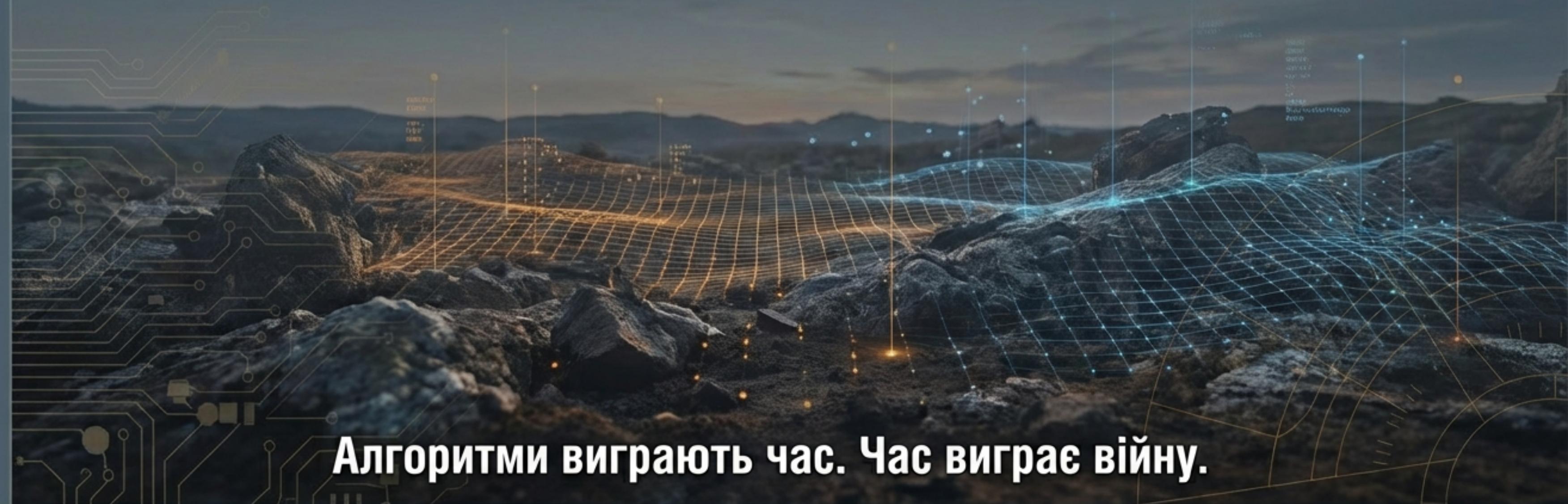
Структурування: Хаос  
перетворюється на дані.



Алгоритми: Швидкість  
'Орієнтації' перемагає.



Еволюція: Від Excel  
до архітектури коду.



**Алгоритми виграють час. Час виграє війну.**