

ЛАБОРАТОРНА РОБОТА № 5

"Розробка системи моніторингу стану серверного обладнання та сповіщення про інциденти"

Мета: Набуття практичних навичок щодо побудови та використання діаграм класів з урахуванням використання патернів проектування.

Хід роботи:

Завдання 1: Побудувати діаграму класів з врахуванням використання патернів програмування.



Рис.1. Діаграма класів з урахуванням патернів проектування

					ДУ «Житомирська політехніка».22.121.31.000 – Лр5		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Ярошинський С.В.					
Перевір.		Левківський В.Л.					
Керівник							
Н. контр.							
Зав. каф.							
					Літ.	Арк.	Аркушів
						1	4
					ФІКТ Гр. ІПЗ-22-2[1]		

Звіт з
лабораторної роботи

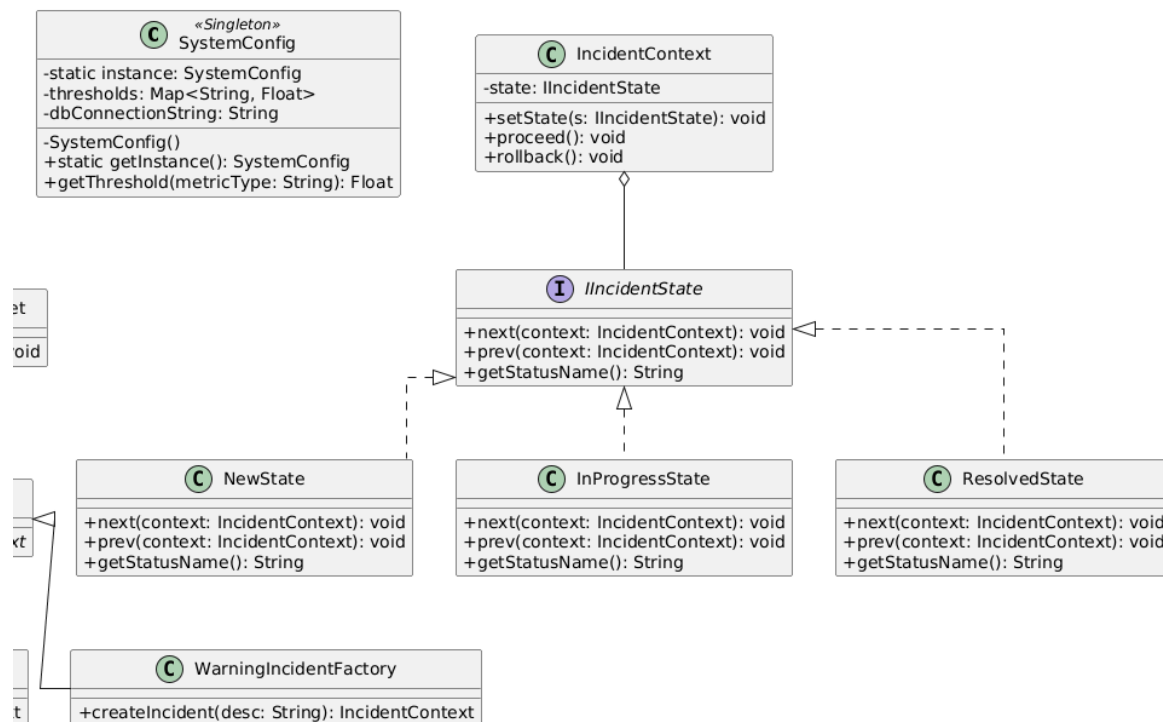


Рис.2. Діаграма класів з урахуванням патернів проектування

У розробленій архітектурі системи моніторингу використано 5 ключових патернів проектування, які вирішують типові проблеми гнучкості та масштабування.

1. Singleton (Одинак)

- **Клас:** SystemConfig
- **Призначення:** Забезпечує існування єдиного екземпляра класу конфігурації на всю програму.
- **Застосування:** У системі моніторингу критично важливо, щоб усі модулі (сенсори, аналізатори) використовували одні й ті самі порогові значення температур та налаштування підключення до БД. Патерн гарантує централізований доступ до цих налаштувань.

2. Observer (Спостерігач)

- **Класи:** ServerNode (Subject), IMetricObserver (Observer), AlertSystem, DashboardWebSocket.

- **Призначення:** Визначає залежність "один-до-багатьох", де зміна стану одного об'єкта автоматично сповіщає всіх залежних.
- **Застосування:** Коли ServerNode отримує нову метрику (наприклад, стрибок навантаження CPU), він автоматично викликає метод update() у підписників. Це дозволяє одночасно оновити графіки на дашборді та перевірити умови для тривоги в AlertSystem, не змінюючи код самого сервера.

3. Strategy (Стратегія)

- **Класи:** INotificationStrategy, EmailStrategy, TelegramStrategy, NotificationContext.
- **Призначення:** Визначає сімейство алгоритмів та дозволяє змінювати їх незалежно від клієнта.
- **Застосування:** Система дозволяє змінювати канал сповіщення "на льоту". Наприклад, вдень інженер отримує Email, а вночі — Telegram. Клас NotificationContext не знає деталей реалізації відправки, він лише делегує це завдання обраній стратегії.

4. State (Стан)

- **Класи:** IncidentContext, IncidentState (та його реалізації New, InProgress, Resolved).
- **Призначення:** Дозволяє об'єкту змінювати свою поведінку при зміні внутрішнього стану.
- **Застосування:** Інцидент проходить складний життєвий цикл. У стані New його можна взяти в роботу, у стані Resolved — закрити. Патерн State інкапсулює логіку переходів і забороняє некоректні дії (наприклад, не можна перевести інцидент з New одразу в Closed без етапу роботи).

5. Factory Method (Фабричний метод)

- **Класи:** IncidentFactory, CriticalIncidentFactory, WarningIncidentFactory.
- **Призначення:** Надає інтерфейс для створення об'єктів, але залишає підкласам рішення про те, який клас інстанціювати.
- **Застосування:** Система може генерувати різні типи інцидентів залежно від серйозності. Фабрика дозволяє стандартизувати процес створення інцидентів.

		Ярошинський С.В.			ДУ «Житомирська політехніка».22.121.31.000 – Лр5	Арк.
		Левківський В.Л.				3
Змн.	Арк.	№ докум.	Підпис	Дата		

ту (наприклад, критичний інцидент одразу отримує високий пріоритет і SMS-сповіщення, а попередження — лише запис у лог), приховуючи цю логіку від клієнтського коду.

Висновок: у ході виконання лабораторної роботи ми набули практичних навичок щодо побудови та використання діаграм класів з урахуванням використання патернів проектування.

		Ярошинський С.В.			ДУ «Житомирська політехніка».22.121.31.000 – Лр5	Арк.
		Левківський В.Л.				
Змн.	Арк.	№ докум.	Підпис	Дата		4