****

****

COS 214 Project Report

Group name: idk

(Nickname: Arrays of Sunshine)

u23587832 Hannah Koorbanally u23673941 Louise Bruwer u23530996 Kiara Hodgson

u23605376 Brendan du Plooy u21439631 Euan Botha u21437883 Nolan Kühn u23782219 Ruan le Roux

GITHUB REPOSITORY:

<https://github.com/COS214-Project-2024/idk>

LINK TO DOCUMENT:

<https://docs.google.com/document/d/1w2eHsOXzge7_VIquZCeK1Lx0pm3v5aoK9sUKoSMN8PU/edit?usp=sharing>

**Table of Contents**

[**Section 1 : Research Brief 4**](#_b4emgb8cobev)

[1. Urban Development 4](#_2px69ei0gncb)

[2. Key Principles for City Management 4](#_vb79mweov148)

[**3. Role of Various Components In The City 4**](#_vn3tgkhsuv7x)

[3.1 Buildings 4](#_9xfuuhpzjltm)

[3.2 Utilities 4](#_7lcko08vm71t)

[3.3 Transportation 4](#_7wazopx0crfe)

[3.4 Citizens 5](#_oemdoz6iau4m)

[3.5 Government 5](#_7pzo9oxvnwq)

[3.6 Resources 5](#_3q1d6oa124nb)

[3.7 Taxes 5](#_1vrjuoncqbra)

[3.8 City Growth 5](#_vyzgjovgn3ms)

[4. Influence on design: 6](#_x3v1e08v5mb9)

[4.1 Utilities 6](#_nlhiodtpwo4)

[4.2 Transportation 6](#_z3kaceb1kx7c)

[**Section 2: Design Pattern Application Report 7**](#_w7zlttprrytt)

[1. Iterator 8](#_34smz2fzlwre)

[2. Composite 9](#_cvm65trm6u7h)

[3. Factory Method 10](#_3ovgbl9x3feb)

[4. Prototype 10](#_xwqdd5h6y7wm)

[5. Mediator 10](#_n1s3q9rs1z7g)

[6. Template Method 10](#_bbqb4sllf962)

[7. Strategy 11](#_wy10l1kvf7ox)

[8. State 11](#_qv57jj2ntigr)

[9. Chain of Responsibility 12](#_g0tgtkmsz0ev)

[10. Command 13](#_2m7rch26xvn5)

# 

# Section 1 : Research Brief

### Urban Development

Urban development transforms cities to drive economic & social growth. It focuses on improving living standards while addressing environmental concerns that arise from high population density. Key goals include enhancing quality of life through accessibleservices, housing & efficient transportation. Government, S. (2024)

### Key Principles for City Management

We found particularly relevant the principles of Acquiring sufficient funding (which includes financing running costs, as well as levying taxes) as well as that of Accountable governance, as per Nusca, A. (2024). Our government will be made to be as efficient and have as positive of an impact on citizens as possible.

### Role of Various Components In The City

### Buildings

Buildings serve multiple roles in a city. Residential properties, for example, house citizens, influencing their satisfaction and contributing to economic development. While each building type has a primary function (e.g., providing space for business, production, or residence), it also impacts the city's economy by contributing to GDP (Architecture Courses, 2024).

### Utilities

According to the Council of Scientific and Industrial Research (CSIR), residential utilities support domestic needs by providing resources like electricity and materials, while collective services handle waste management and sewer systems. Maintaining these utilities is crucial for citizen satisfaction and efficient public services.

### Transportation

Many factors can influence the method of transport citizens take. These factors include traffic, cost of vehicles/travel & the time taken to travel. The time taken for travel, in particular, may impact the citizen satisfaction of the city. Both private and public transportation varies depending on the given scenario. Factors such as traffic, cost of vehicle for public and private ownership and the time of travel influence which method is most efficient to travel across the city. Transit, P. (2021)

The transportation component is used within our system to influence factors such as citizen satisfaction, impact from a given scenario (car accident) and the generation of income from public modes of transport (impact on Bank and Government). The transportation section therefore has an economic, functional and qualitative role based on the effective management of the city.

### 

### Citizens

The concept of “citizen-centricity” came up in our research. Thisconcerns the prioritising of citizens’ demands in the designing and delivery stages of public services (Berntzen et al., 2016; Kamalia & Nor, 2017)

### Government

From our research, the government manages all aspects of the city. Examples are managing citizen satisfaction and welfare by ensuring residents have services & adequate housing, as well as Resource Management, where local governments are key managers of essential services like water & energy. Hoeflich de Duque, S. (2023)

### Resources

Urban areas depend heavily on resources to support their populations and activities. However, rapid urbanisation drives up resource consumption, complicating efforts to uphold sustainability principles. Therefore, effective resource management is essential for achieving sustainable urban development, focusing on maximising resource use while minimising waste and environmental impact. Zucaro, A., Maselli, G. and Ulgiati, S. (2021)

### Taxes

Taxes play a key role in how residents feel about their government and their own finances. Changes in tax rates affect people's bank accounts and their satisfaction with public services. There’s often a gap between what citizens want from their local government and what they’re willing to pay in taxes. When people see improvements in services that match their tax dollars, it boosts their overall happiness and financial situation. Glaser, M. A., & Hildreth, W. B. (1999)

### City Growth

Population growth significantly impacts taxes by increasing the demand for essential services like housing. As cities expand, they may need to raise tax rates or introduce new taxes to fund these services. On the other hand, a larger population can also broaden the tax base, potentially allowing for lower rates. Bahl, R., Holland, D. & Linn, J., (2015)

### Influence on design

Our government was made to be as efficient and have as positive of an impact on citizens as possible.

References

* Government, S. (2024) *Urban development*, *Portal GOV.SI*. Available at: https://www.gov.si/en/policies/environment-and-spatial-planning/prostor-2/urban-development/ (Accessed: 29 October 2024).
* Korppoo, K. (2015) *Designing game analytics for a city-builder game*, *Trepo*. Available at: https://trepo.tuni.fi/handle/10024/97480 (Accessed: 22 October 2024).
* Nusca, A. (2024) Four principles of Effective City Management, ZDNET. Available at: https://www.zdnet.com/article/four-principles-of-effective-city-management/ (Accessed: 22 October 2024).
* Villagomez, E. (2023) *S101s - describing building types: Why they matter*, *Spacing Vancouver*. Available at: https://spacing.ca/vancouver/2023/09/11/s101s-describing-building-types-why-they-matter/ (Accessed: 22 October 2024).
* Transit, P. (2021) Public vs. private transportation: Advantages & Disadvantages, Safe and Reliable Bus Charter Rentals in Ventura County. Available at: http://www.pegasustransit.com/post/public-vs-private-transportation-advantages-disadvantages (Accessed: 28 October 2024).
* Hoeflich de Duque, S. (2023) *Sustainable development through local governments*, *Urbanet*. Available at: https://www.urbanet.info/local-governments-sustainable-urbanisation/ (Accessed: 29 October 2024).
* Zucaro, A., Maselli, G. and Ulgiati, S. (2021) *Insights in Urban Resource Management: A comprehensive understanding of unexplored patterns*, *Frontiers*. Available at: https://www.frontiersin.org/journals/sustainable-cities/articles/10.3389/frsc.2021.807735/full (Accessed: 29 October 2024).
* Glaser, M. A., & Hildreth, W. B. (1999). Service Delivery Satisfaction and Willingness to Pay Taxes: Citizen Recognition of Local Government Performance. *Public Productivity & Management Review*, *23*(1), 48–67. <https://doi.org/10.2307/3380792>
* Bahl, R., Holland, D. & Linn, J., (2015). *Urban growth and local taxes in less developed countries*. Available at:<https://scholarspace.manoa.hawaii.edu/server/api/core/bitstreams/4b852e95-4d3b-4c3c-a9ba-e166f883d452/content> [Accessed 29 Oct. 2024]
* Architecture Courses (2024) *Building Types*. Available at:<https://www.architecturecourses.org/learn/building-types> (Accessed: 2 November 2024)

# Section 2: Design Pattern Application Report

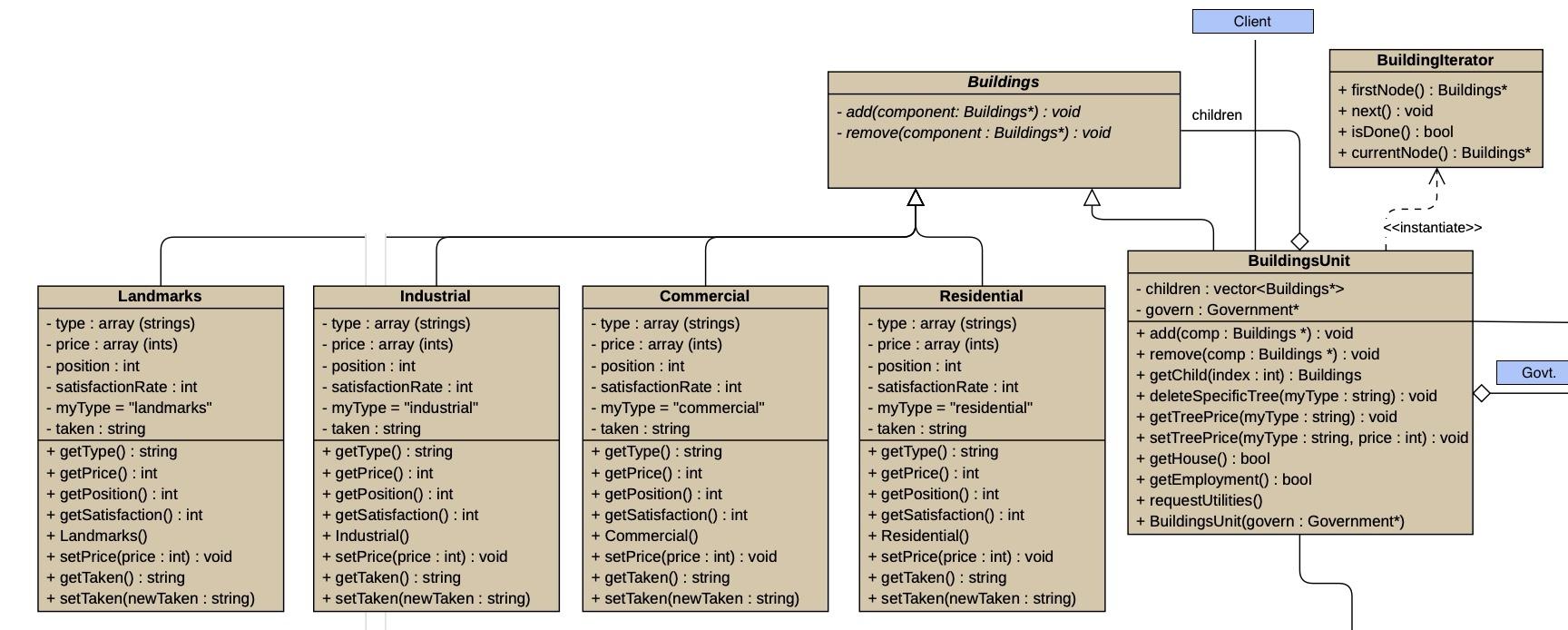
## **Iterator**

The Iterator Design Pattern provides a structured way to traverse elements of an aggregate object sequentially without exposing its underlying structure. This pattern is particularly useful for components that need to access or process a sequence of elements, such as methods, fields, or functions, without needing to understand how those elements are organised internally.

We used this pattern for our **Buildings** component to efficiently locate and access specific types of building units.

In this context, the Iterator pattern is implemented within our BuildingIterator class, providing the ability to navigate through a list of BuildingUnit objects, thus enabling the retrieval of a specific unit based on set requirements.

This approach makes it straightforward for other classes or functions to find and retrieve the necessary BuildingUnit object, ensuring that the process remains clear and unobtrusive, without adding complexity to the existing code structure.

**

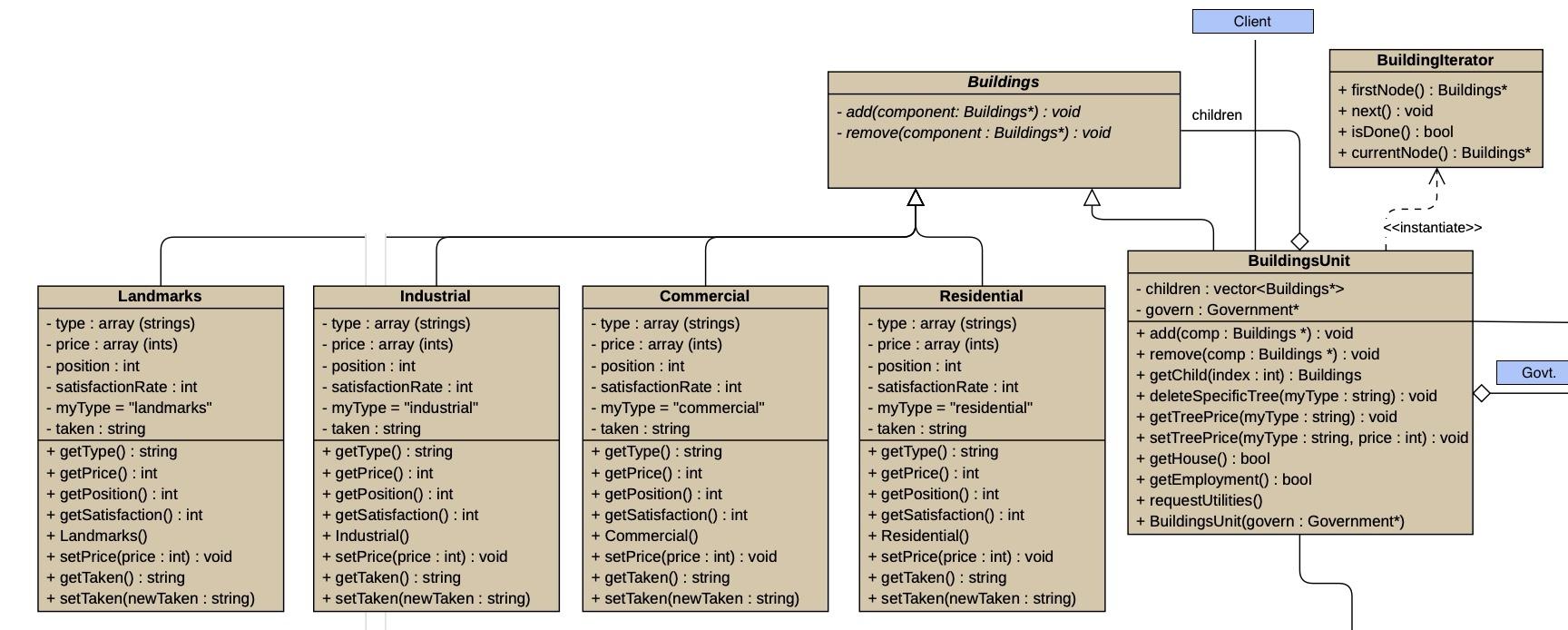
## **Composite**

The Composite Design pattern is intended to organise objects into tree-like structures to represent part-whole hierarchies. Since this pattern allows clients to handle single objects and groups of objects uniformly, it was beneficial to our system as it enabled us to handle all specific building types in the same way as a generalised building unit.

The Composite Design Pattern was chosen to support the modular structure of the Buildings component, enabling both individual building units and collections of units to be **treated uniformly.** This design pattern is particularly suited for managing the diverse requirements of different building types—such as Residential, Industrial, Commercial, and Landmarks—while allowing each type to function as an **independent unit or as part of a larger composite structure**.

Within this pattern, BuildingUnit serves as the composite, while each building type (Residential, Industrial, Commercial, Landmarks) acts as a leaf, providing the specific functionalities needed by citizens. The primary component, Buildings, provides the generalised common ability to add or remove buildings, which is then implemented accordingly in the BuildingsUnit hierarchy.

By structuring buildings and units through the Composite pattern, the system ensures that every building type can be handled consistently by other parts of the system, regardless of whether they are individual units or grouped structures. Each BuildingUnit is supplied with the necessary utilities to function fully, enabling residential spaces, business areas, factories, and landmarks to contribute equally to citizen satisfaction and city development. This uniformity allows buildings to interact with other classes, such as those representing government functions or banking, without needing special handling. Furthermore, the BuildingIterator enhances the BuildingUnit functionality by allowing efficient traversal of building lists, aiding in the timely retrieval of specific units as needed by the system. Overall, the Composite pattern, along with the iterator, establishes a clear, consistent, and functional structure for managing buildings within the city simulation.

**

## 

## **Factory Method (done)**

The Factory Method pattern provides an interface for creating an object but allows subclasses to decide which specific class to instantiate. This enables classes to delegate instantiation to subclasses, promoting flexibility by decoupling the client from the concrete products being created

In this system, a Factory object is instantiated in the main, responsible for continuously creating Citizen objects without additional parameters. Once a citizen is created, it can be copied repeatedly to represent scenarios like twins. Each Citizen can then request housing or employment, which triggers the Buildings unit to check a list for availability, where available entries are marked as taken = false. When a match is found, the status updates to taken = true, and the system notifies the citizen that housing or employment has been secured.

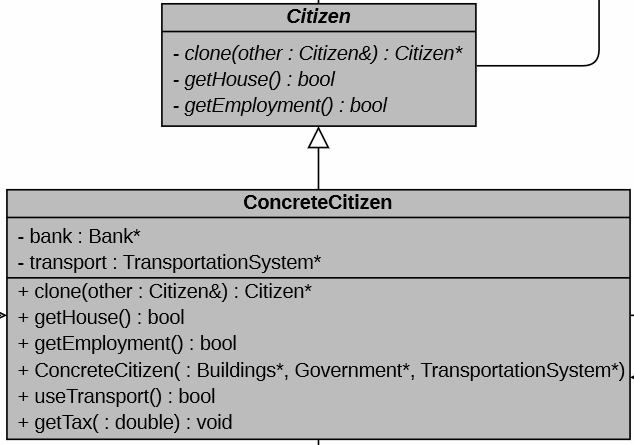
The Factory Method pattern simplifies this process by organising citizen creation in a centralised and manageable way, decoupling Citizen creation from specific details and allowing Buildings to handle housing and job assignments. This separation reduces the complexity in the main application logic, enabling efficient resource allocation and tracking, while keeping object life cycles consistent and orderly across the system.

## 

## **Prototype (done)**

Seeing that all citizens share the same fundamental properties such as a place of residence as well as making use of resources such as transportation, it would be beneficial to make use of the Prototype design pattern to ensure that the citizens created come from a prototypical instance. However, along with the use of the Factory Method design pattern, the properties instantiated for an existing Citizen can be copied onto the newly created citizen, emulating the concept of a family where family members would share the same properties such as the same house.

In the creation of new citizens, it would be timely and un-effective to create citizens that derive from each other from scratch through another design pattern, however, with the Prototype design pattern, one is able to ensure that uniform creation is possible. Seeing that all citizens share the same fundamental makeup (their members: to make use of a bank, as well as a form of transport), making a copy of a prototype and its uniquely populated members (through the Factory Method) would be most efficient in the creation of citizen objects that share the same properties. This makes use of DRY coding, to ensure that code is not repeated unnecessarily.

******

## **Mediator**

Define an object that encapsulates how a set of objects interact with each other by promoting loose coupling through keeping objects from explicit referral and vary interactions independently.

Seeing that in the real world, most processes pass through the government to ensure that all processes and members are tracked and recorded, the use of using the Mediator design pattern as a central connection between all the various components ensured that the Government could complete its intention of being the

Channel between all components. The channel will ensure loose coupling to ensure that various components are not explicitly referring to one another without making the Government aware but rather interacting with each other through the Government instead.

The government forms the central connection between all components in the system ensuring that all components can interact with each other through itself. This ensures that the government is aware of all processes occurring within the system. The needs of the government is to ensure that this interaction and communication does not occur independently from the government, by ensuring that it encapsulates the party’s communication and ensuring that explicit referral is not possible. Thus maintaining its centralism and its awareness of all processes affecting the city.

* 1. ***Screenshot of this pattern in the big class UML***

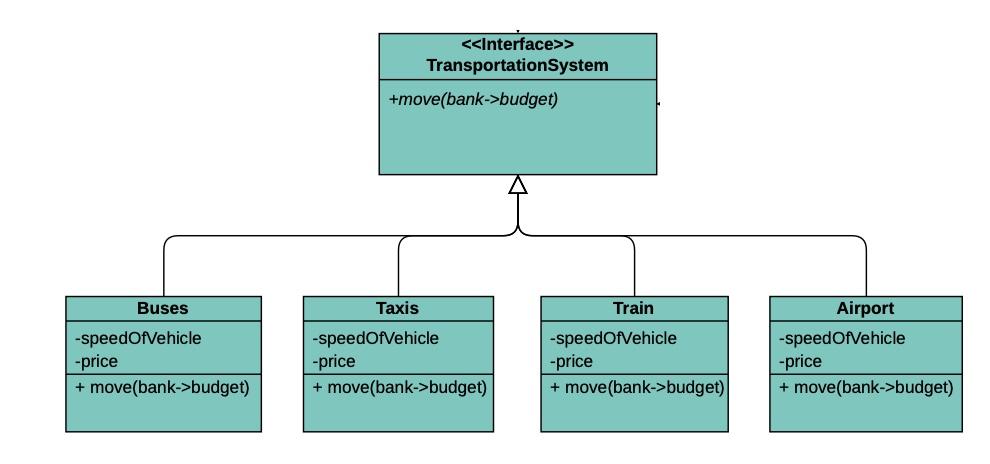
## 

## **Template Method (done)**

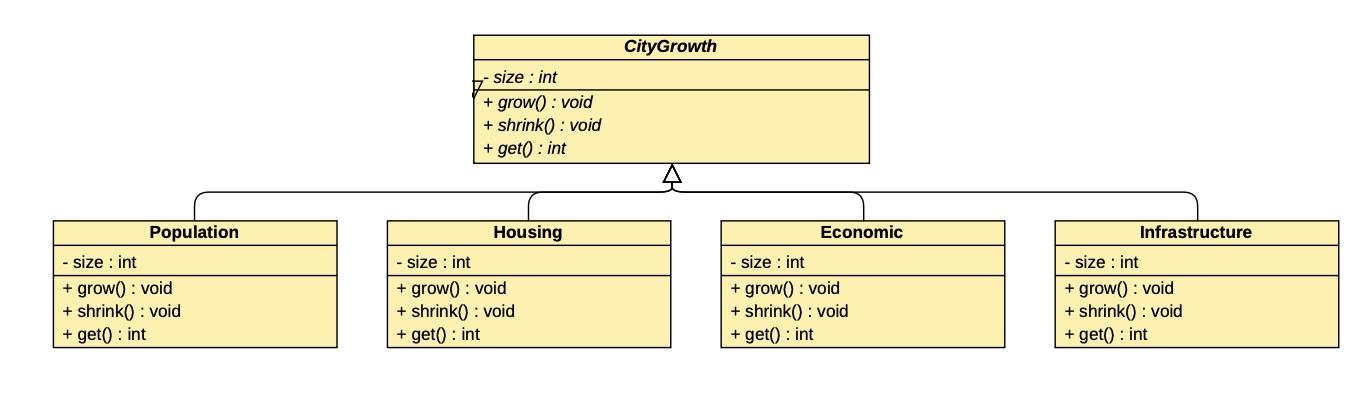
Defines the skeleton of an algorithm in operation by deferring some steps to its subclasses which will then redefine the steps of the algorithm without changing it structure

For the transport component, this approach works best because all the different modes of transport share a common method with similar implementations. Defining a primitive operation in a common abstract class, therefore, enhances code reuse and reduces duplication. Additionally, if a mode of transport needs to be added or removed in the future, this will be much easier with the template method implemented. This reasoning also applies to the city growth component. Different factors affect city growth, but they all have similar methods with similar implementations. Therefore implementing the template method is most efficient.

Transport

******

## City Growth

******

## **Strategy (done)**

The Strategy pattern defines a group of related algorithms, encapsulates each one, and makes them interchangeable. This allows the algorithm to vary independently from the clients using it, providing flexibility and reducing dependencies on specific implementations.

In the tax strategy component, the Strategy pattern allows the government to manage various tax calculations through an abstract taxStrategy class. Three derived classes - propertyTaxStrategy, incomeTaxStrategy, and salesTaxStrategy - each implement the calculateTax() method, enabling different tax types to be calculated without modifying government.

This approach simplifies the design by removing complex conditionals and enables the government to set or change tax strategies dynamically via setStrategy(). Encapsulating each tax calculation ensures robustness, as any changes or additions to tax algorithms don’t impact the government class, keeping the system flexible and easier to maintain.

* 1. ***Screenshot of this pattern in the big class UML***

## **State (done)**

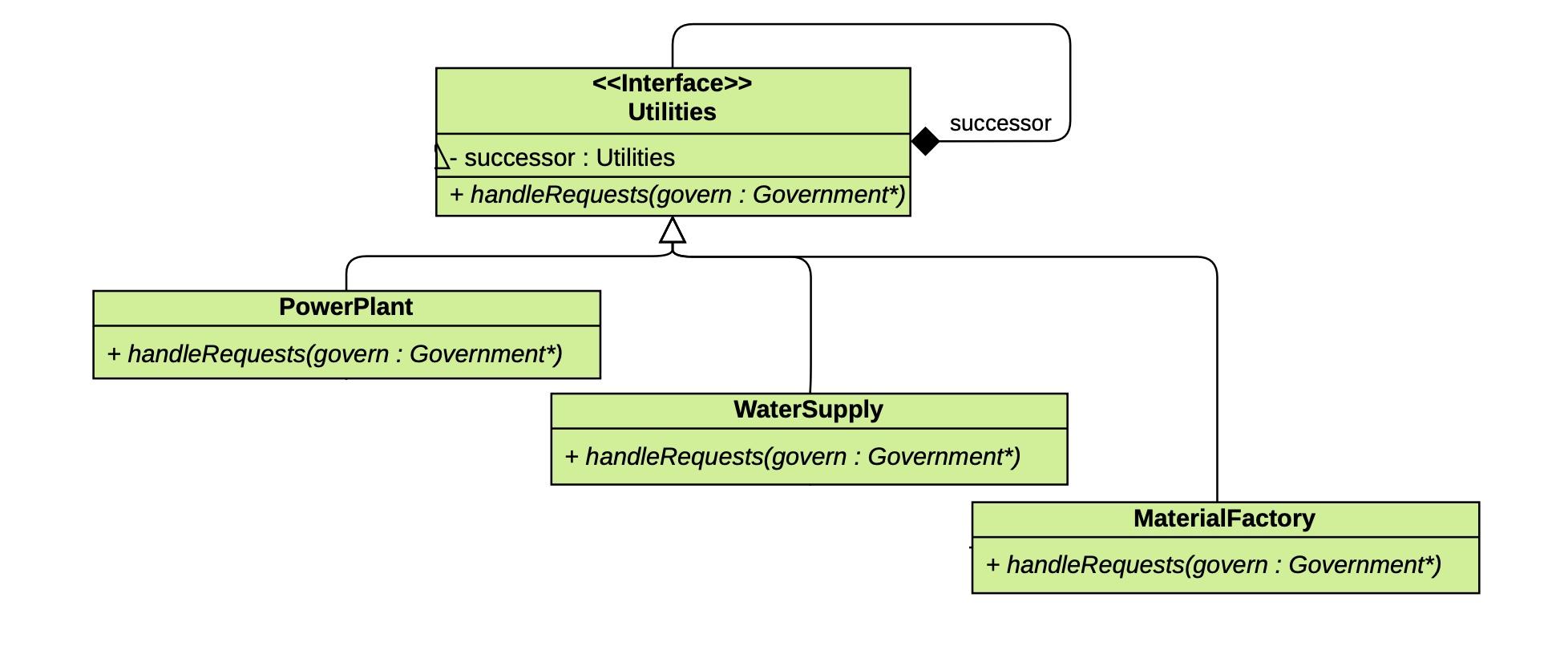
The State pattern allows an object to alter its behaviour when its internal state changes, making the object appear to change its class. This pattern was chosen because it enables resources to dynamically modify their behaviour based on their current state. For resources, particularly in managing their usage, the State pattern allows interactions to adapt according to the quantity available. Since the amount of a resource affects how other classes utilise it, this pattern enables resources to adjust behaviour in response to their quantity state.

This pattern was implemented to manage the resource flow more effectively: depending on the quantity of a specific resource, certain methods—such as generateResource() and useResource()—are called to meet the demands of other components, like Utilities. When a utility requests a specific quantity of a resource, the request is processed by the Government, which in turn requests the exact quantity from the resource. Based on availability, different methods are triggered: if the resource quantity is insufficient, generateResource() is called to increase stock, followed by useResource() to utilise the available amount. If enough of the resource is available, generateResource() is bypassed. The various states of full, empty, or normal dictate how a resource behaves, making the state diagram an appropriate structure for this functionality.



## **Chain of Responsibility (done)**

This design pattern helps avoid tightly coupling the sender of a request to its receiver by allowing multiple objects to handle the request, passing it along a chain until one takes responsibility. In our city system, this pattern is particularly well-suited because the Utilities class must supply each BuildingUnit with resources such as power and water. By decoupling the sender (BuildingUnit) from the receiver (the specific utility), each resource request can be routed along a chain, allowing only the utility responsible for generating that resource to handle the request. This approach supports modular design and efficient processing of varied resource requests.

Each BuildingUnit needs to be equipped with specific resources, and this is achieved by calling the responsible utility, which manages both the creation and distribution of the requested resource. The decoupling achieved with this pattern means that each request is directed only to the relevant utility, ensuring that each resource is provided by the correct handler. This allows the system to manage multiple requests independently and efficiently, with each utility taking responsibility only for the resources it controls.

## 

## **Command**

The Command pattern encapsulates a request as an object, allowing clients to be parameterized with different requests, enabling queuing or logging of requests, and supporting undoable operations. By wrapping requests in command objects, this pattern separates the object that initiates the operation from the one that performs it, offering greater flexibility and control over when and how requests are executed.

The Command pattern enhances the system’s flexibility by allowing the Government to issue different commands - such as handling accidents, managing loadshedding, or addressing crime - through a common interface. This setup allows for easier modification of operations in response to new requirements or changes in command handling. Encapsulating each situation as a command also allows the government to log, queue, or even undo specific operations if needed. By decoupling the Government from the specifics of each action, the design keeps the code modular, making it easy to add new commands in the future while maintaining clear boundaries between request initiation and execution.

## 

## 

Presentation

Components

1. Buildings

* Requirements: The task involves creating a simulation that includes a variety of building types, each with distinct purposes and roles. These building types include *residential buildings*, such as houses; *commercial buildings*, including shops and offices; *industrial buildings* like factories, and plants; and *landmarks*, such as parks and monuments.

Each building type should have specific attributes and behaviours that contribute uniquely to the simulation, impacting factors such as citizen satisfaction, economic growth, and resource consumption.

* ~~Solution: We made use of the Composite design pattern to ensure uniform treatment of each type of building as well at each building unit to create tree-like, part-whole hierarchies.~~

1. Utilities

* Requirements: The task requires us to model essential utilities that are critical to the city’s functioning. These utilities include power plants, a water supply system, waste management, and sewage systems. Each of these utilities should interact dynamically with buildings and citizens in the simulation, influencing both the functionality of the city and the satisfaction of the citizens.
* ~~Solution: Chain of responsibility was used to allow for the building units to have their requests handled by the appropriate utility receiver to equip each unit with the needed resources to be a functional unit.~~

1. Resources

* Requirements: The simulation must include resource management to ensure sustainable city operations. Key resources to track include *materials*, such as wood, steel, concrete; as well as energy, water, and the city’s budget. Efficient management of these resources will support city expansion and the effective provision of services to its citizens.
* ~~Solution: Seeing that the resources’s quantity state was a determining factor to if more needed to be created before the use of the resource, the state pattern was used to ensure~~

1. Citizens

* Requirements: Citizens are central to the city’s vitality, creating demand for housing, jobs, and services. The simulation should model population growth, reflecting a dynamic increase based on various influencing factors. Employment opportunities, driven by industrial and commercial buildings, will play a role in citizens’ livelihoods, while essential services such as healthcare, education, security, and entertainment meet their daily needs. Citizen satisfaction, influenced by taxes, amenities, and overall quality of life, should also be tracked. Additionally, citizens will react to government policies, economic shifts, and infrastructure changes, creating a responsive and evolving city environment.
* ~~Solution: We made use of a combination of the Prototype design pattern and the Factory Method for both a uniform creation of citizen objects but also for the transferral of uniquely assigned members to other members to create family-like structures.~~

1. Transport

* Solution: Template pattern was used to allow various forms of transport to redefine steps of the move() method without changing the structure of the move() algorithm

1. Government

* Requirements: The government system within the simulation is responsible for overseeing city governance, encompassing several key functions. This includes taxation, where the government sets and collects taxes from both citizens and businesses; budget management, which involves allocating financial resources for various city services and projects; and policy implementation, where laws and regulations are established to shape city dynamics. Additionally, the government manages public services such as healthcare, education, and law enforcement. The decisions made by the government should significantly influence citizen satisfaction, economic growth, and the overall development of the city, creating a complex interplay between governance and community well-being.
* Solution: Government should be the central mediator for all processes in the system to promote loose coupling, hence making the government aware of all processes and communications that occur within the city

1. Taxes

* Requirements: Taxation serves as a vital mechanism for funding city services and infrastructure within the simulation. It should model various aspects of taxation, including adjustable tax rates for different categories, such as income, property, and sales. The simulation must also incorporate mechanisms for tax collection from both citizens and businesses, as well as the allocation of collected taxes to various city departments and projects. Furthermore, the impact of tax changes on citizen satisfaction and economic activity should be analysed, highlighting how these financial decisions influence the overall functioning and prosperity of the city.
* Solution

1. City Growth

* Requirements: City growth in the simulation should be portrayed as a dynamic process shaped by various interrelated factors. Population growth will be driven by birth rates, migration patterns, and economic opportunities. As the population increases, housing needs will necessitate the expansion of residential buildings to accommodate residents. Economic development will also play a crucial role, with the growth of commercial and industrial sectors leading to the creation of more jobs. Additionally, infrastructure expansion, including the development of utilities and transportation systems, is essential to support this growth. The mechanics of growth should ensure a cascading effect, where each aspect of the city influences the others, resulting in a realistic simulation of urban development.
* Solution