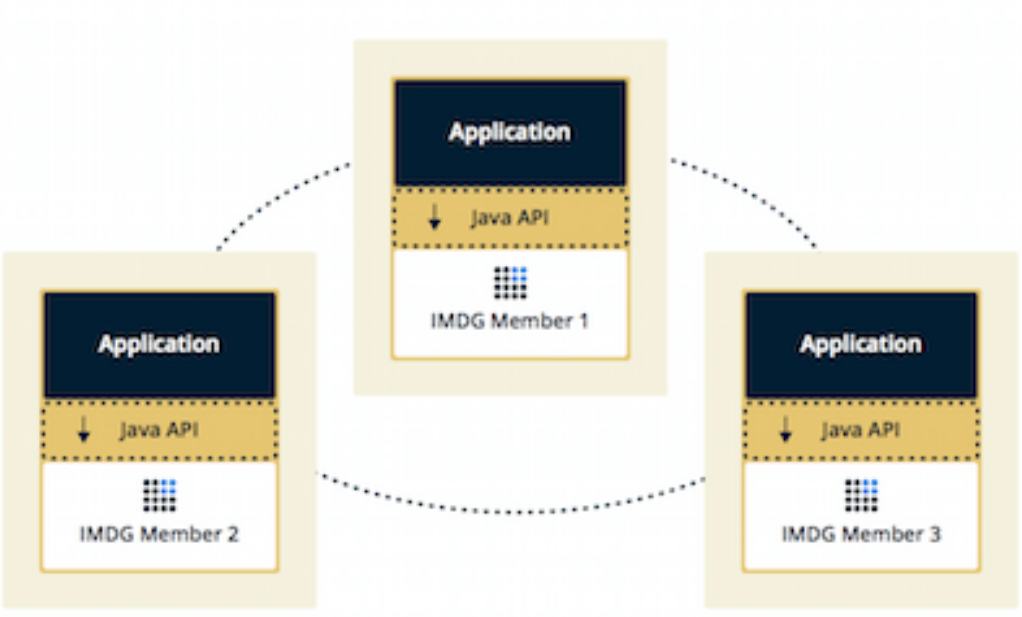
Task 7
Розгортання і робота з distributed in-memory data structures на основі Hazelcast: Distributed Map

Hazelcast є розподіленим сховище даних в оперативній пам'яті. "Розподілене" означає те, що на кожній з нод (серверів) системи запускається свій екземпляр Hazelcast, які потім об'єднуються в загальний кластер. В рамках цього кластера, через API можна створювати різні розподілені структури даних: Map, Queue, Topic, Lock, ...

Запуск Hazelcast з Java-застосування
Ноду можна створювати та запускати напряму з Java-додатку, який буде працювати на тому же сервері де запущений екземпляр Hazelcast. Додаток матиме доступ через API доступ до цих розподіленим структурам даних, і зможе писати/читати в/з них.

```
HazelcastInstance hzInstance = Hazelcast.newHazelcastInstance();
Map<String, String> distributedMap = hzInstance.getMap( "capitals" );
    capitalcities.put( "1", "Tokyo" );
    capitalcities.put( "2", "Paris" );
```

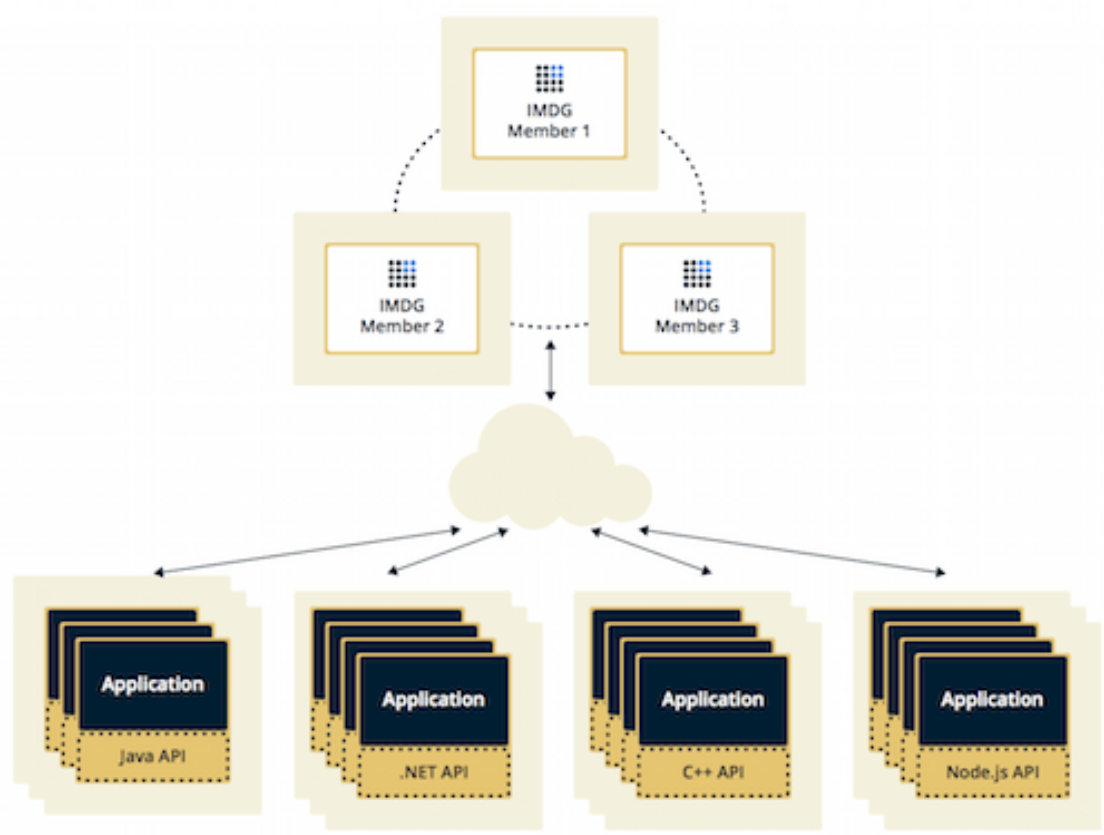
При цьому, інші додатки підключені до інших нод кластеру Hazelcast будуть також бачити зміни в розподіленим структурам даних, і також можуть писати/читати в/з них.
https://docs.hazelcast.com/imdg/4.1.1/overview/topology.html



Запуск нод Hazelcast окремо

Іншим способом є запуск нод кластеру, як окремих застосувань через командний рядок (https://hazelcast.org/imdg/get-started/) та підключення до нього за допомгою клієнтів, які існують під різні мови програмування

```
HazelcastInstance hzInstance = Hazelcast.newHazelcastClient();
Map<String, String> distributedMap = hzInstance.getMap( "capitals" );
    capitalcities.put( "1", "Tokyo" );
    capitalcities.put( "2", "Paris" );
```



<u>Завдання:</u>

1) Встановити і налаштувати Hazelcast http://hazelcast.org/download/

2) Сконфігурувати і запустити 3 ноди (інстанси) об'єднані в кластер або як частину Java-застосування, або як окремі застосування https://hazelcast.org/getting-started-with-hazelcast/

3) Продемонструйте роботу Distributed Map
http://docs.hazelcast.org/docs/latest/manual/html-single/index.html#map
- використовуючи API створіть Distributed Map
- запишіть в неї 1000 значень з ключем від 0 до 1к
- за допомогою Management Center (https://hazelcast.org/imdg/get-started/) подивиться на розподіл значень по нодах
- подивитись як зміниться розподіл якщо відключити одну ноду/дві ноди. Чи буде втрата даних?

4) Продемонструйте роботу Distributed Map with locks
- використовуючи 3 підключення (чи підключення з 3х клієнтів) одночансо запустіть приклади за посиланням з підрахунку значень в циклі: а) без блокування; б) з песимістичним; в) з оптимістичним блокуванням
http://docs.hazelcast.org/docs/latest/manual/html-single/index.html#locking-maps
- порівняйте результати кожного з запусків

5) Налаштуйте Bounded queue
- на основі Distributed Queue (http://docs.hazelcast.org/docs/latest/manual/html-single/index.html#queue) налаштуйте Bounded queue (http://docs.hazelcast.org/docs/latest/manual/html-single/index.html#setting-a-bounded-queue)
- з однієї ноди (клієнта) йде запис, а на двох інших читання
- перевірте яка буде поведінка на запис якщо відсутнє читання, і черга заповнена
- як будуть вичитуватись значення з черги якщо є декілька читачів

6) Налаштуйте Distributed topic (https://docs.hazelcast.com/hazelcast/5.0/data-structures/topic):
- первірте поведінку з налаштуванням *globalOrderEnabled* та без нього