<<5/19>>會議記錄

會議日期	2014/5/19
時間	2014/5/19, 12:30~18:00
地點	R440
主持人	劉宗瑋
記錄者	吳佳倫
目的	討論 Homework andClass diagram

參與者

姓名	E-mail	角色
蔡宗翰	b99902066@ntu.edu.tw	組長
林映孜	d98944002@ntu.edu.tw	組員
范哲誠	d02922030@ntu.edu.tw	組員
黃奕軻	b99902032@ntu.edu.tw	組員
劉宗瑋	b99902100@ntu.edu.tw	組員
呂俊宏	enricolu@gmail.com	組員
吳佳倫	bingo4508@gmail.com	記錄

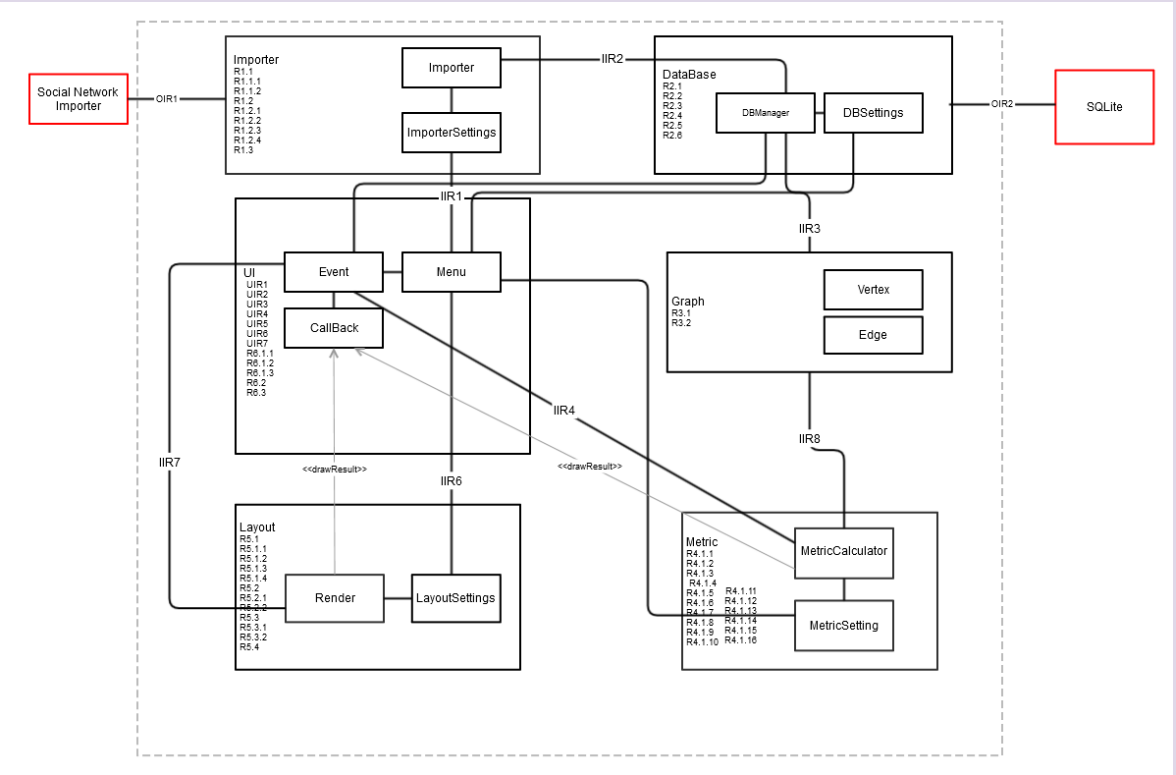
會議議程

1. **Review homework (12:30~13:15)**
2. **Discuss project –Class diagram (13:15~18:00)**

會議討論議題

(2014/5/13, 16:20~19:05 @ R106)

1. **Original Structure**



2. Discussion:

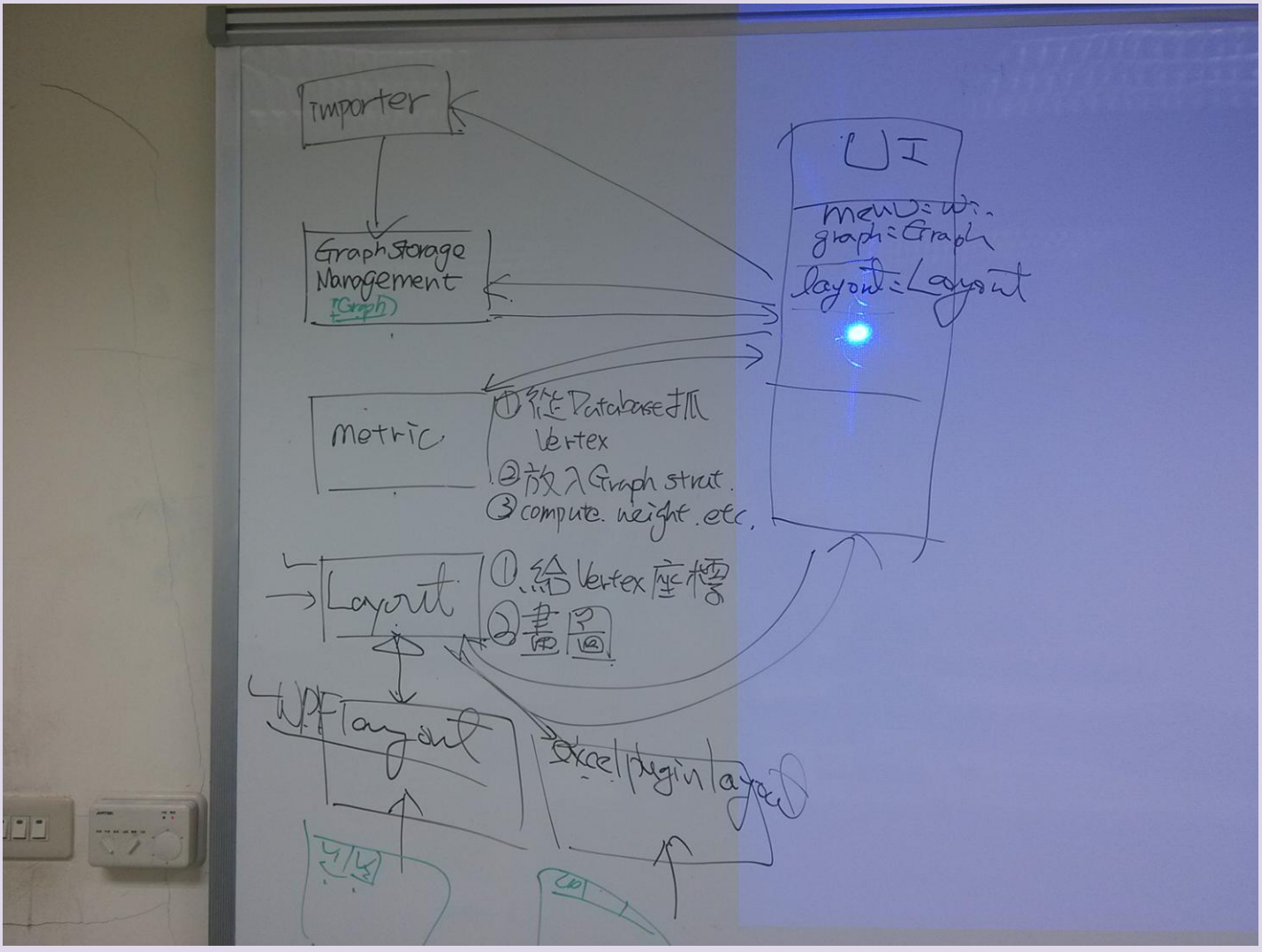
a. 觸發活動的是 Importer or Data Base？
   - 都可以，使用者可以自己透過 UI 來控制

b. Graph 跟 Metric 系統要整合？
   - Graph 的功能分給 Database 跟 Metric, 從系統降級為 Database 中的 public class (所有程式可以去看它)

c. UI 可以控制絕大部分的系統 (原 Architecture 圖沒有連接這個)

d. Data Base Requirement 討論
   - R2.4, R2.6 要過濾兩次嗎？
     - 是的，R2.4 (Filter rows by derived restriction)之後，R2.6 Convert selected data to "Graph" structure 才會發生

3. 新版 Architecture



4. 畫 class diagram / 設計 UI 分工：

Importer –哲誠

GraphStorageManagement –奕軻、佳倫

Metric –宗瑋

Layout – 林映孜

UI – Eric 2

5. 整合

Architecture – 宗翰

(2014/5/15, 19:00 – 21:00 @R439)

1. Review Homework 10 [2hr]

According to requirements, our application design contained two main functionalities. The first is torecognize different type of tokens in RTF document given, so that corresponding routines whichconvert a RTF format token of specified type (e.g. paragraph and character) to another token ofdifferent format can be used properly. The second is the actual work to convert a RTF token, so thatall converted tokens will compose a different format document (e.g. TeX).In initial design, both functions are coded in the same class, which is responsible for tokenizing adocument, calling various convert routines, doing the actual convert work. To provide the firstfunctionality, a switch statement invokes different routines (e.g. convertCharacter()) according tothe token type. The second one is provided through many convertXXX() routines each of whichgenerate a same type token but of different format according to the wanted document format.During the refactoring process, we all agreed on the encapsulation of the second functionality whichincluded may convert routines into a converter class per document format. However discussion thatthe switch statement should be placed in converter class or reader class couldn't reach consensus.Both solutions would be presented probably.

"Pros and Cons of process"

Pros: quality ensurance in specific enviroment

經驗傳承

Cons: lack of flexibility

cost of maintenance

(2014/5/19, 12:30 – 18:00 @R440)

1. 討論 Importer -

1. UI 把 setting 過的資料傳遞給 ImporterSetting，Importer 去 ImporterSetting 抓資料。

2. Importer 遇到的問題：

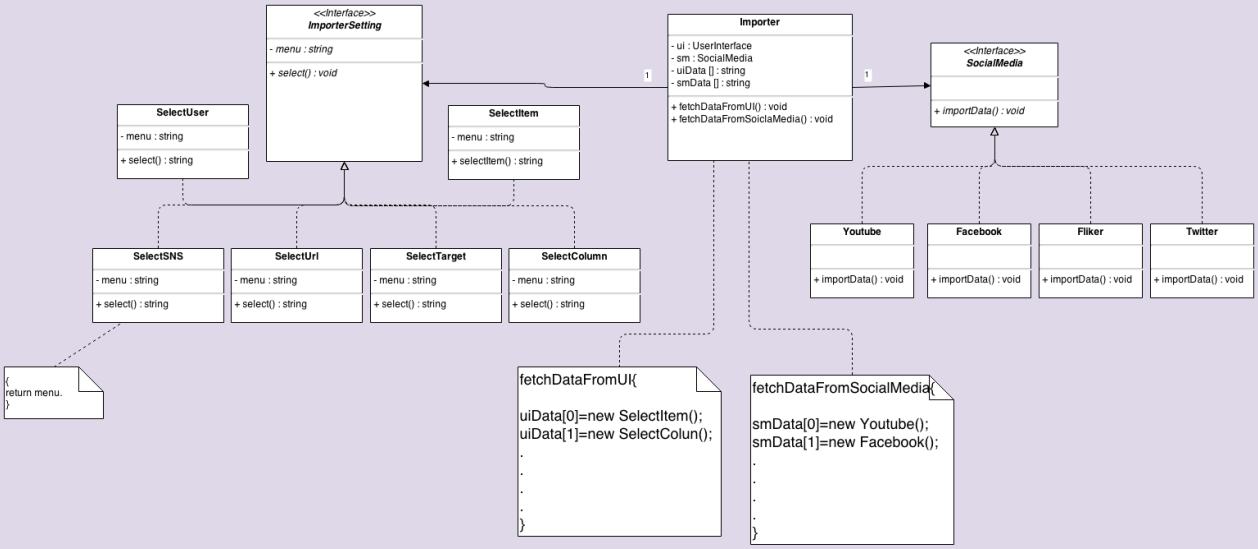
a.從各種 Social Media 抓下來的資料結構不一樣(如 Facebook、Youtube)，因此可以產生各式各樣的 Layout，但整合上有困難。

b.單一個 Social Media 抓下來的資料會產生很多 table，如果要整合所有 Social Media 抓下來的資料處存在 Database 裡，許多資料會被捨棄，或者是需要容量較大的處存空間。

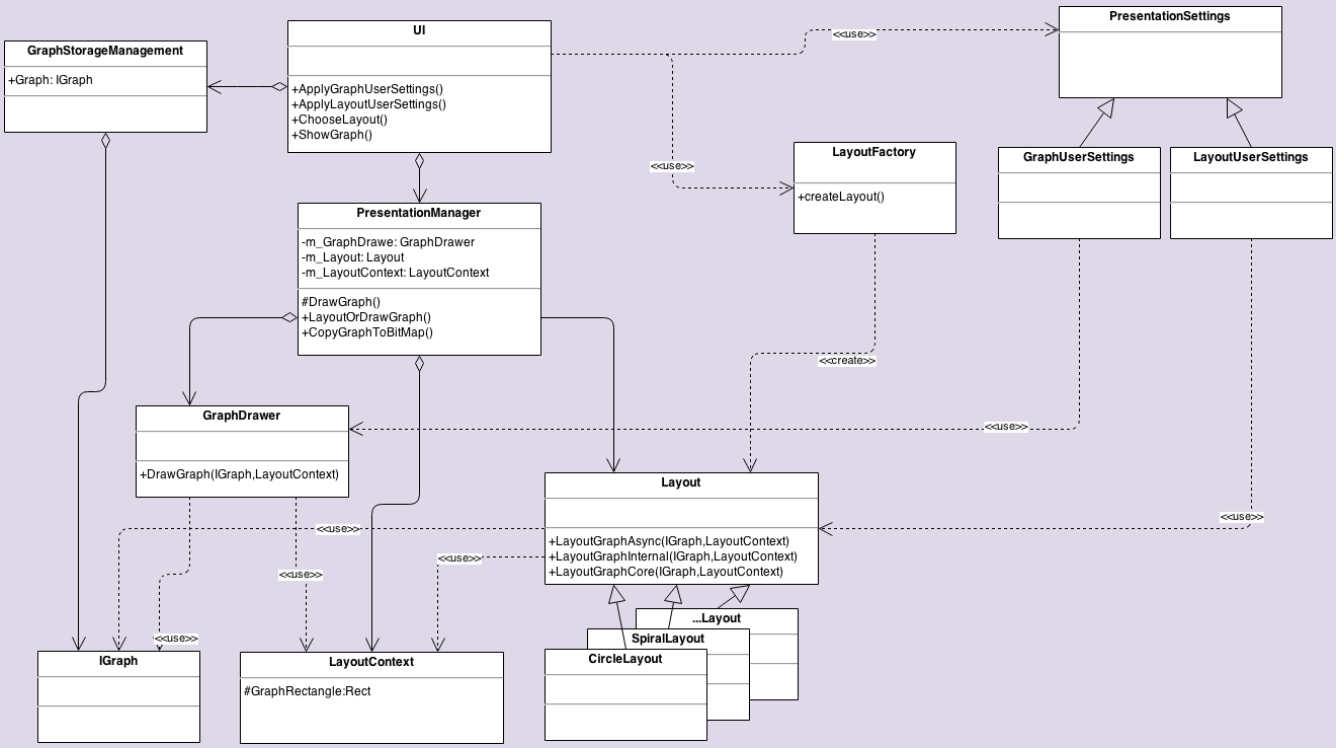
3. Importer 目前解決方案:

a.簡單版：實作 import 每種 Social Media，限制從每個 Social Media 抓下來的資料量，產生較少的 Layout。

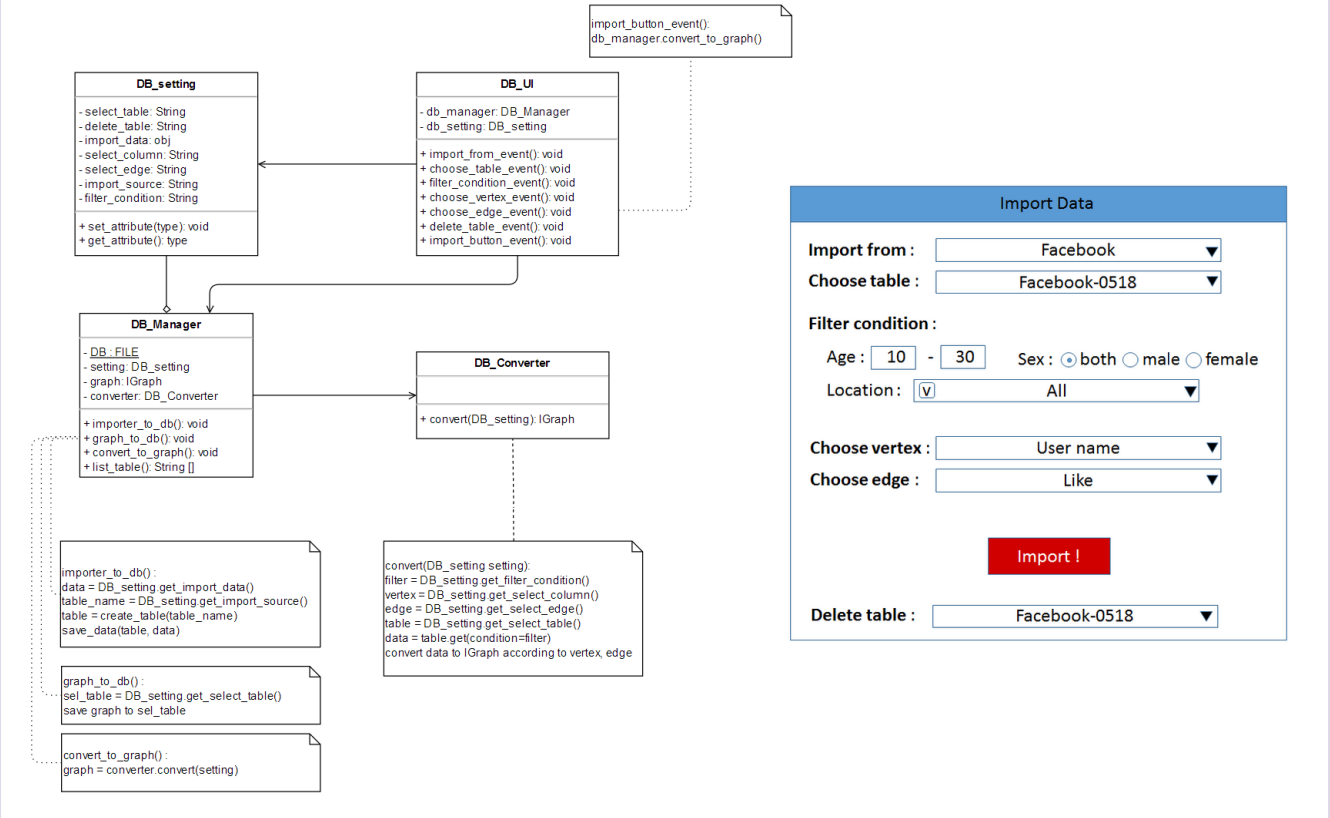
b.深入版：深入一種 Social Media，並且產生出許多不同的 Layout。



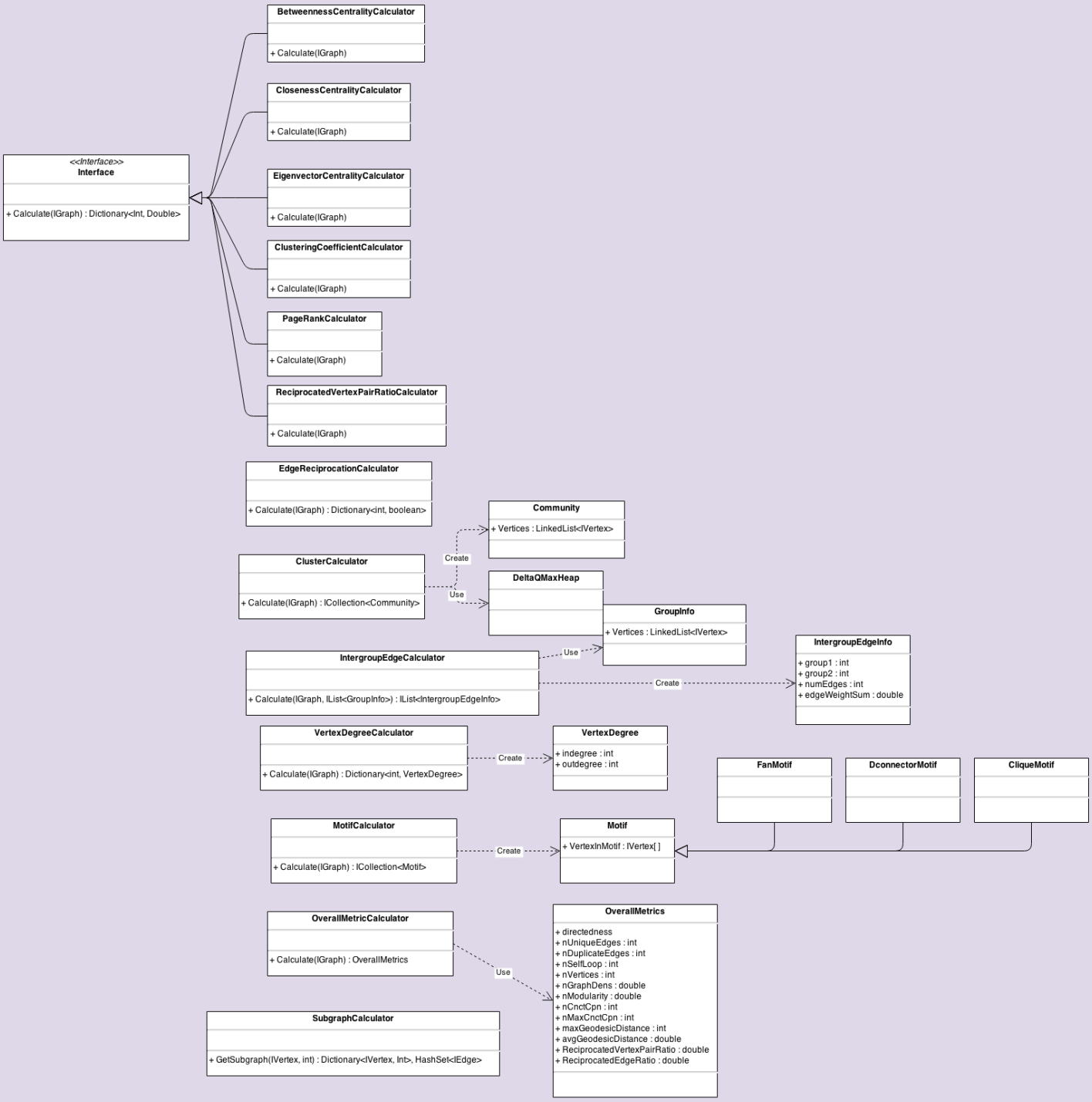
3. DB 宣告一個 db 的 reference, 在 constructer connect to db



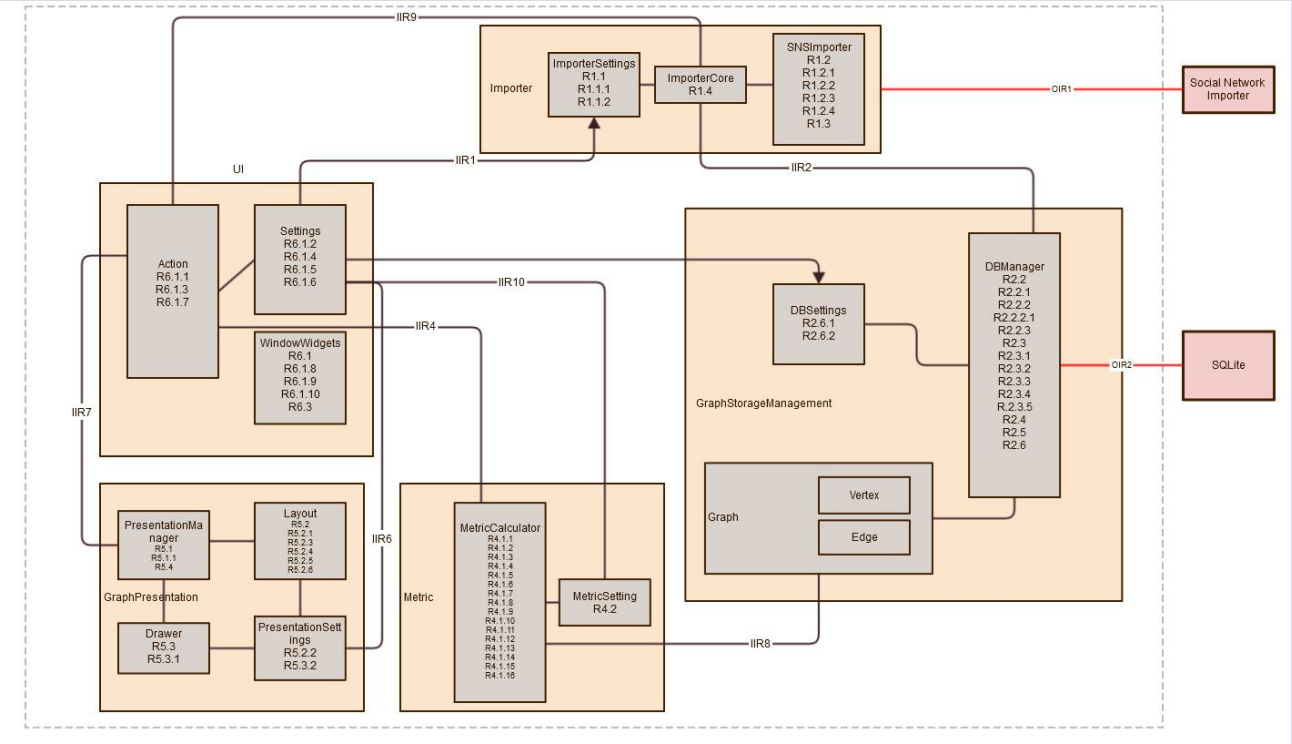
4. GraphStorageManagement



5. Metric


BetweennessCentralityCalculator
+ Calculate(IGraph)
ClosenessCentralityCalculator
+ Calculate(IGraph)
<<Interface>>
Interface
+ Calculate(IGraph) : Dictionary<Int, Double>
EigenvectorCentralityCalculator
+ Calculate(IGraph)
ClusteringCoefficientCalculator
+ Calculate(IGraph)
PageRankCalculator
+ Calculate(IGraph)
ReciprocatedVertexPairRatioCalculator
+ Calculate(IGraph)
EdgeReciprocationCalculator
+ Calculate(IGraph) : Dictionary<int, boolean>
Community
+ Vertices : LinkedList<IVertex>
ClusterCalculator
+ Calculate(IGraph) : ICollection<Community>
Create
Use
DeltaQMaxHeap
GroupInfo
+ Vertices : LinkedList<IVertex>
IntergroupEdgeCalculator
+ Calculate(IGraph, IList<GroupInfo>) : IList<IntergroupEdgeInfo>
Use
Create
IntergroupEdgeInfo
+ group1 : int
+ group2 : int
+ numEdges : int
+ edgeWeightSum : double
VertexDegreeCalculator
+ Calculate(IGraph) : Dictionary<int, VertexDegree>
Create
VertexDegree
+ indegree : int
+ outdegree : int
FanMotif
DconnectorMotif
CliqueMotif
MotifCalculator
+ Calculate(IGraph) : ICollection<Motif>
Create
Motif
+ VertexInMotif : IVertex[ ]
OverallMetricCalculator
+ Calculate(IGraph) : OverallMetrics
Use
OverallMetrics
+ directedness
+ nUniqueEdges : int
+ nDuplicateEdges : int
+ nSelfLoop : int
+ nVertices : int
+ nGraphDens : double
+ nModularity : double
+ nCnctCpn : int
+ nMaxCnctCpn : int
+ maxGeodesicDistance : int
+ avgGeodesicDistance : double
+ ReciprocatedVertexPairRatio : double
+ ReciprocatedEdgeRatio : double
SubgraphCalculator
+ GetSubgraph(IVertex, int) : Dictionary<IVertex, Int>, HashSet<IEdge>

6. Architecture based on Designed Class Diagram



- **Data Importer**

Functional Requirement	
R1.1	UI can save settings in ImporterSettings
R1.1.1	UI can select SNS which can be fetched in ImporterSettings
R1.1.2	UI can select url/target/user which can be fetched in ImporterSettings
R1.1.3	UI can select items/columns which can be fetched in ImporterSettings
R1.2	Importer can import data from selected SNS
R1.2.1	Importer can import data from Facebook
R1.2.2	Importer can import data from Twitter
R1.2.3	Importer can import data from Youtube
R1.2.4	Importer can import data from Flickr
R1.3	Importer can load data from file(graphML)
R1.4	Core can load settings and execute with corresponding concrete importer.

GraphStorageManagement

Non-Functional Requirement	
R2.1	Memory usage should be more efficient than original NodeXL
Functional Requirement	
R2.2	Table Design
R2.2.1	Design Table Naming method
R2.2.2	Design Different Social Media table schema & primary key
R2.2.2.1	Facebook Schema
R2.2.3	Design Index method
R2.3	Table Function

R2.3.1	List Tables
R2.3.2	Add Table
R2.3.3	Delete Table
R2.3.4	Select Table
R.2.3.5	UI Design
R2.4	Filter rows by derived restriction
R2.5	Save graph structure to DB(new table)
R2.6	Convert selected data to "Graph" structure according to Settings
R2.6.1	UI can select which column of the data to be used as vertices and save them in Settings
R2.6.2	User can Match an (weighted) edge according to relationship between vertices (select edge) and save in Settings

- **Metric**

R4.1	Calculate graph metrics according to Settings and generate result.
R4.1.1	Compute ClosenessCentrality of each vertex in a graph
R4.1.2	Compute BetweennessCentrality of each vertex in a graph
R4.1.3	Compute EigenvectorCentrality of each vertex in a graph
R4.1.4	Compute PageRank of each vertex in a graph
R4.1.5	Compute Max/Avg GeodesicDistances of a graph
R4.1.6	Compute Modularity of a graph
R4.1.7	Generate clusters of a graph by Girvan-Newman algorithm
R4.1.8	Generate communities of a graph by Clauset Newman Moore algorithm
R4.1.9	Identify all Cliques of a graph
R4.1.10	Calculate clusters using the Wakita-Tsurumi algorithm
R4.1.11	Calculate in/out degree of each vertex in a graph
R4.1.12	Identify all strongly connected components of a graph
R4.1.13	Calculate intergroup edges of any pair of group and within a group
R4.1.14	Calculates the reciprocated vertex pair ratio for each of the graph's vertices
R4.1.15	Generate a subgraph according to a specified vertex
R4.1.16	Partition the graph into motifs
R4.2	UI can select some algorithm to calculate graph metrics and save in Settings.

- **Graph Representation**

Non-Functional Requirement	
	Unify the Rectangle to the same type: System.Windows.Rect (WPF)
Functional Requirement	
R5.1	NodeXL has a view to show graph on it
R5.1.1	To show graph on the view, we first layout the graph then draw them on the view
R5.2	When layouting, we specify the location of each vertex in the graph according to the layout context and layout type.
R5.2.1	The types of layouts includes: CircleLayout,GridLayout, Random Layout,

	SinusoidLayout, etc.
R5.2.2	The layout type is selected by the user
R5.2.3	We can only plot one type of layout for a graph at one time.
R5.2.4	It is easy to add new types of layouts.
R5.2.5	The layout context describe the width,height of the layout view.
R5.2.6	The location of each vertex contains the x,y coordinates
R5.3	When drawing graph, we draw the vertices and edges in the graph on the view according the layout context and the properties of vertices and edges
R5.3.1	The vertices and edge in the graph can be draw with different color, border
R5.3.2	The used color are depend on user's selection
R5.4	The user is able to Export current layout view into image file

- **UI**

Functional Requirement	
R6.1	There are 3 section in the application : 1. Toolbar on the top. and below the toolbar - 2. Table on the left 3. Graph on the right
R6.1.1	User can choose to load data from file, database or SNS importer.
R6.1.2	After the data is loaded, user can choose which column of the data to be vertices and input restrictions to filter rows.
R6.1.3	User can choose to export the data to image file or graphML file (need to specify the file path)
R6.1.4	User can select multiple metric to be computed at once (check box)
R6.1.5	Once the metrics are computed, user can use one of the metrics to autofill columns(color, width, style, opacity, visibility, label)
R6.1.10	There is a dropdown list widget, which includes the layout that user can choose.
R6.1.6	User can select a layout from dropdown list.
R6.1.7	User can refresh the graph
R6.1.8	User can Show/hide table
R6.1.9	User can Show/hide graph

- **IIR**

IIR1	**Importer read setting info from UI**
IIR2	**Imorted data is stored to DataBase**
IIR3	**Graph can ask data from DataBase**
IIR4	**UI can ask Mertic to compute metrics**
IIR5	**Layout can get data from Graph to show.**
IIR6	**Layout can ask for setting info from UI.**
IIR7	**After computated, Layout can notify UI to render the result.**
IIR8	**Metric can ask data from Graph**
IIR9	UI can ask Importer to Import data and save in DB.
IIR10	UI can select which Metric to be calculated and save in settings.

- **OIR**

OIR1	Importer use "SocialNetworkImporter" to get data from SNS
OIR2	DataBase use "SQLite" to save and fetch data.

Action Item 後續處理項目

編號	處理動作	負責人員	處理期限	狀態	備註
1	WBS draft	黃奕軻	4/1	**Closed 4/1**	蔡宗翰 review
2	提供 Github public key	全員	4/1	**Closed 4/1**	
3	新版 WBS	Eric	4/4	**Closed**	
4	加入 Implementation 的詳細內容	全員	4/7	**Closed**	On google doc
5	繪製 High Level Architecture	Eric	4/8	**Closed**	
6	上傳前次的 Meeting Minutes	吳佳倫	4/8	**Closed 4/1**	
7	WBS—分析、設計階段認領	全員	4/7	**Closed**	
8	設定 Github key, 並確認成功上傳至少一次	全員	4/8	**Closed**	奕軻 review
9	Homework6	吳佳倫	4/8	**Closed 4/08**	
10	Check all Commit	黃奕軻	4/8	**Closed 4/08**	
11	Review System Architecture	全員	4/14	**Closed 4/14**	映孜 review
12	Code Survey	全員	4/14	**Closed 4/14**	
13	Terminology and Naming	呂俊宏	4/14	**Closed 4/14**	
14	Homework 7-1	蔡宗翰	4/15	**Closed 4/15**	
15	Homework 7-2	呂俊宏	4/15	**Closed 4/15**	
16	Homework 8-1	林映孜	4/21	**Closed 4/21**	
17	Homework 8-2	劉宗瑋	4/21	**Closed 4/21**	
18	Homework 9-1	范哲誠	4/28	**Closed 4/28**	
19	Homework 9-2	廖尉棠	4/28	**Closed 4/28**	
20	User requirement	全體	4/28	**Closed 4/28**	
21	System requirement	全體	5/4	**Closed 5/5**	
22	Interface requirement	全體	5/5	**Closed 5/5**	
23	Homework 10	吳佳倫	5/6	**Closed 5/5**	
24	Refine System Architecture	全體	5/13	**Closed 5/12**	
25	Presentation of Requirements & Architecture	蔡宗翰	5/6	**Closed 5/6**	
26	Modifying requirement	全體	5/12	**Closed 5/12**	
27	Homework 11	黃奕軻	5/13	**Closed 5/12**	
28	Re-assign duties for further works	全體	5/13	**Closed 5/13**	
29	Homework 12	劉宗瑋	5/12	**Closed 5/19**	
30	Class Diagram	全體	5/19	**Ongoing**	

下次會議

日期	時間	地點

5/20 Tue.	16:30	R104