



# **BNP Paribas: Enterprise Architecture**

A Major Qualifying Project Report

Submitted to the faculty of the  
WORCESTER POLYTECHNIC INSTITUTE  
In partial fulfillment of the requirements of the  
Degree of Bachelor of Science

by

**Zhen He**, Computer Science and Actuarial Mathematics

**Joe Servi**, Mathematics

**Qiu Chen**, Computer Science and Actuarial Mathematics

**Siqi Wang**, Management and Mathematics

**Project Sponsor:** BNP Paribas

**Submitted to:**

On- Site Liaisons: Scott Visconti  
Arjun Kohli

Project Advisors: Professor Arthur Gerstenfeld, Department of Management  
Professor Dan Dougherty, Department of Computer Science  
Professor Jon Abraham, Department of Mathematics

**Submitted on**

Wednesday, March 14, 2012

## **Abstract**

The goal of the project, sponsored by the Global Equities and Commodity Derivatives (GECD) at BNP Paribas, was to create a graphical management application that allows the group to better visualize a wide array of system flows. It was required for the application to have bi-directional communication with internal BNP databases and clearly show any possible problems with the data, such as an overloaded server. The outcome of the project is an integrated application that graphically displays and interactively manages business flows and their underlying data and builds performance/capacity dashboards.

## Acknowledgement

There were many people involved with the Enterprise Architecture project both from BNP Paribas and Worcester Polytechnic Institute. Foremost we would like to thank Mr. Scott Visconti, our sponsor, who oversaw the project from the BNP Paribas side. Also Mr. Arjun Kohli, the project manager, who managed the day to day operations of the project. We want to extend our gratitude towards Professor Arthur Gerstenfeld, director of the Wall Street Project Center, for creating such an energizing experience. Along with Professor Gerstenfeld, we want to thank Professors Dan Dougherty and Professor Jon Abraham, our faculty advisors, for their continued support and advice and assistance in our final report and presentation. Finally we thank everyone in the GECD division of BNP Paribas who lent their advice, support and opinions concerning the eventual destination of the project. Thank you.

## Authorship

The academic work of this project consists of 3/3 unit of work in B-Term 2011 (October 2011 – December 2011) and an additional 1/3 unit of work in C-Term 2012 (January 2012 – March 2012). Zhen He and Joe Servi took credit of the work done in B-Term 2011. Later since Zhen He is a double major, she partnered with Qiu Chen and Siqi Wang and finished the rest 1/3 unit of work related to this project in C-Term 2012.

For the work done in B-Term 2011, at the tool evaluating phase of the project, Zhen He took full credit of the development in ApEx and Graphviz and Joe Servi took full credit of the development in Visio and Data Modeler. At the formal development stage of the project, Zhen He and Joe Servi made equal contributions to the development in the final product of this Enterprise Architecture project.

The work done in C-Term 2012 was incorporated into the original report as Appendix D.

The writers are listed to the right of the corresponding sections of this report.

Abstract – Zhen He and Joe Servi

Acknowledgement – Joe Servi

Executive Summary – Zhen He

Introduction – Zhen He and Joe Servi

Background – Zhen He

Requirement – Zhen He

Business Uses – Zhen He

Development Phase I – Zhen He and Joe Servi

Development Phase II – Zhen He and Joe Servi

Results – Zhen He

Conclusion – Joe Servi and Zhen He

Appendix – Zhen He, Qiu Chen, and Siqi Wang

References – Zhen He

## Executive Summary

BNP Paribas' Global Equities and Commodity Derivatives (GECD) division offers a variety of industry leading financial products. To keep its businesses running successfully, the division relies on various internal applications (systems) and hardware that conduct and support many business operations to function correctly and efficiently. IT managers and infrastructure teams need to have a firm control over the health of these applications and hardware within the context of different business flows. Whenever there is a performance problem, they should be informed right away.

For years, however, managers kept track of business flows and performed capacity planning manually. It was extremely costly and inefficient.

To achieve automation and increase efficiency in this area, BNP Paribas IT managers wanted to visualize their entire "enterprise architecture" and perform capacity management accordingly. At the same time, Human Resources managers and Business managers also sought a better visualization of the organization for their own purposes. Obviously, the existing database level of presentation of the data could not meet their need.

With this said, this project aimed to help BNP Paribas to visualize and manage their "enterprise architecture". The goal of the project was to develop a well-integrated application which addresses two key problems – visualization and capacity management. It should have four basic components of implementation. The first component is to automatically extract data from the database and feed it into a data management interface. The second component is to create the visualization. The third and fourth are graph interactivity and capacity analysis respectively. The final component can be further broken down to two parts – to reflect the health of any entity on the graph and to build a detailed performance dashboard for application or hardware.

The project entailed two phases. At the first phase, which is a tool evaluating phase, a variety of tools and technologies were reviewed. Two major visualization tools – Graphviz and Microsoft Visio were assessed at this phase. Oracle Application

Express (ApEx) was tested as a potential application interface. Major components of implementation were explored on each of these tools as thoroughly as possible. Key functions were implemented and demos on dummy data were given to the stakeholders of the project.

Research on the tools showed that Graphviz has its strength in representing structural information (Gansner, Emden R., Eleftherios Koutsofios, and Stephen North). It is powerful in positioning optimization and rich in graphical attribute options. The simplicity of the input file Graphviz takes in to render the graph also made it easier to automate the whole process. These were all crucial merits that would help the final build of the application.

Microsoft Visio, by contrast, is strong in graph interactivity including drag and drop. It was relatively difficult to automate the graphing process with Visio, however, and testers had to optimize positioning explicitly in the development of the tool. While graph interactivity was good to have in the product, it was not a necessity. Complexity in automation and extra work for positioning nodes and links in the graph, on the other hand, represented significant flaws in the tool.

As the single tool tested as an application interface builder, ApEx performed well. It is web-based, specialized in database applications, fast to develop, and easy to deploy and administrator.

At the end of the first phase, testers of the project concluded that the combination of Graphviz and ApEx seemed to be the next step to go given the timeframe. Based on the analysis provided by the testers, the management agreed with this conclusion. The project then moved to the second phase. It was a formal development phase where full capacity of the two tools was explored and the final product was built.

By the end of the project, a well-integrated visualization and data management application was built. The application has the following features: Visualization based on User Selection, Graph Interactivity, Capacity Metrics, Summary View & Detailed View, Processes Sub-graph, and Data management. It also has other functionalities including User Log, Saved Diagrams, and Graphical Attribute Management.

The application is a good blend of different technologies. Three major tools, six programming languages, and other technical concepts were deployed and practiced. Graphviz, SQL Developer, and Apex, as the graphing tool, the programming environment, and the application interface developer respectively, were heavily used and well integrated to function as a unit. Towards the end of the project, the visualization was incorporated into the application interface with all the behind-screen functionalities generated by Graphviz still supported. At this point, Graphviz and Apex were completely merged into one application.

One important part of the project is the process data to be visualized. It consists of two parts – enterprise entities that constitute the business flows and relationships among entities that preserve business logic. There are five types of entity data – Application, Hardware, Organization Unit, People, and Process. These five enterprise entities, together with the intricate relationships among them make up the various hierarchical diagrams the visualization generated.

There are many business uses of the product. In general, the uses can be grouped into three categories.

- Performance Management (Capacity Analysis)
- Personnel Management (Organization Chart)
- Technical Management (Processes Sub-graph)

In the future, the management can conduct efficiency analysis, optimize resources allocation, and expand or retract businesses based on the information.

## Table of Contents

BNP Paribas: Enterprise Architecture A Major Qualifying Project Report.....	
Submitted to the faculty of the .....	
WORCESTER POLYTECHNIC INSTITUTE.....	
In partial fulfillment of the requirements of the .....	
Degree of Bachelor of Science.....	
Abstract.....	i
Acknowledgement .....	ii
Authorship.....	iii
Executive Summary .....	iv
1. Introduction.....	1
2. Background.....	4
2.1. BNP Paribas and GECD.....	4
2.2. Data of Interest.....	5
2.3. Technologies Review .....	7
2.3.1. Tools.....	7
2.3.2. Programming Languages .....	8
3. Requirements .....	10
4. Business Uses.....	13
4.1. Performance Management.....	13
4.2. Personnel Management .....	13
4.3. Technical Management.....	13
5. Development Phase I: Tools Assessment, Prototyping .....	15
5.1. ApEx + Graphviz – in-house development .....	15
5.1.1. Pre-analysis on the tool .....	15
5.1.2. Technology used.....	15
5.1.3. Database and Sample data.....	16
5.1.4. Development in Graphviz .....	16
5.1.5. Development in ApEx.....	20
5.1.6. Two way interaction.....	26
5.1.6. Evaluation .....	29
5.2. Visio – in-house development .....	33



5.2.1.	Hierarchical Design.....	33
5.2.2.	Free Form Design.....	33
5.2.3.	Semi-Free Form Design .....	33
5.2.4.	Methodology .....	34
5.2.5.	Bi-directional Interface .....	37
5.3.	Oracle Data Modeler – in-house development.....	39
5.3.1.	SQL Developer Investigation.....	39
5.3.2.	SQL Developer Conclusions.....	39
6.	Development Phase II: Formal Development of Final Product .....	40
6.1.	Detailed Design Plan.....	40
6.1.1.	User Interface Design.....	40
6.1.2.	Database Design.....	42
6.2.	Technology Used.....	43
6.2.1.	Software .....	43
6.2.2.	Programming Language.....	43
6.2.3.	Technical Support .....	44
6.3.	Architecture.....	44
6.4.	Data .....	44
6.5.	Methodology .....	45
6.5.1.	The implementation of user preferences .....	45
6.5.2.	Improvement on graph interactivity .....	45
6.5.3.	Embedment of graph in the application .....	46
7.	Results.....	48
7.1.	Major functions and capabilities implemented.....	50
7.2.	Summary of functions .....	55
8.	Conclusion .....	57
	Appendix A .....	58
	Appendix B .....	60
	Appendix C .....	61
	Appendix D – Report of C-Term 2012.....	63
	Definition .....	63
	Classification.....	63
	Specification .....	64
	Process .....	69

Environmental Requirements.....	69
Well-established regulatory system.....	70
Accurate analysis of cash flows .....	70
Clear and reliable accounting activities .....	70
Accredited public rating organizations .....	71
Comprehensive Investment Banking Service .....	71
Healthy Treasury Bills and Notes market .....	72
Active secondary market.....	72
Diversified investors .....	72
Valuation and Assessment.....	73
Futures Contracts .....	74
Interest Rate Swaps.....	75
Options.....	78
Credit Default Swaps .....	81
Cash Flow Collateralized Debt Obligations.....	87
Overview.....	89
Key Concepts .....	91
Housing Bubble .....	91
Subprime Mortgages.....	92
Securitization .....	93
Definition.....	93
Historical Review.....	94
Process Illustration.....	94
Tranches .....	95
Rationale for Securitization.....	95
Securitization and 2008 Financial Crisis.....	96
Major Game Players.....	99
The U.S. Government – Over interfered the housing market .....	99
Investment Banks.....	101
Rating Agency.....	104
Global Investors .....	107
References.....	108

## 1. Introduction

BNP Paribas' Global Equities and Commodity Derivatives (GECD) division offers a variety of industry leading financial products and services including equity derivatives and commodity derivatives, indices and funds, and research and brokerage services. To keep its businesses running successfully, the division relies on various internal applications (systems) and hardware that conduct and support business operations to function correctly and efficiently. IT managers and infrastructure teams need to have a firm control over the health of these applications and hardware within the context of different business flows. Not only do they need to get alerts right away when a system or a server signals a performance problem, but also to understand how an over-capacity server can impact any particular flow.

Previously, managers of BNP Paribas kept track of business flows and perform capacity planning manually. A few years ago, right before this project was conceived, the company had once attempted to perform capacity analysis on applications and hardware. In order to draw useful conclusions, process data (data that constitutes a business flow and usually can be visualized as a flow diagram) had to be manually entered into *Casewise*, an external tool BNP used for business process analysis, management, modeling, enterprise architecture etc. This took the company nearly a year for the data to be in place. However, the results went out of date shortly after useful analysis was done. It was too costly and inefficient. After that, the company abandoned the way and sought for more automation in this area.

On May 6th, 2010, the financial world was briefly shaken by what has come to be known as the '2010 flash crash.' In a hectic day of trading, the Dow Jones Industrial Average plunged 998.5 points in a few minutes only to recover its losses several minutes later. No conclusive reason was ever agreed upon as to the cause of this hyper-volatile string of events, but the day sent a reverberating pulse through the financial industry calling for something to be done.

As a consequence, it became more important for BNP Paribas IT managers to

manage their entire “enterprise architecture” to prevent another ‘flash crash’ from happening again. Managers wanted to get a better sense of how the organization, business application, and infrastructure components participate in various business flows. However, the existing database level presentation of the data could not meet their need. Process data was still presented in raw database tables. Relationships between the data were still referenced by index. By scanning rows after rows of a table, one could hardly tell anything useful in terms of how the two business entities are connected, not to say perform any useful analysis on their performance. It was difficult to piece together the entire network of interconnected data and to manage capacity was almost impossible.

With this said, the project aimed to help BNP Paribas visualize and manage their “enterprise architecture”. The goal of the project was to develop a well-integrated application which addresses two key problems: an all-in-one-place visualization, and convenient capacity analysis. The application graphically displays business flows and interactively manages their underlying data. It also builds performance/capacity dashboards on business entities within the context of any flow.

The application has four basic components of implementation. The first component is to automatically extract data from the database and feed it into a data management interface. The process consists of making queries to the database and transforming its raw relational form into a more presentable format displayed in the user interface. The second component is to graph the data. The graph should be able to establish relationships at different levels. For example, if it currently displays a general view with processes as elements, after drilldown, it should reveal what is going on underlying each of the processes. The third component is graph interactivity. The graph should allow for simple user interactions. Based on user selection and preferences, the graph should collapse or expand a node, hide part of the picture, or roll up or roll down at a particular level. The final component is capacity analysis. This component involves two parts. One is to reflect the health of any entity on the graph so managers spot the problem at the first sight of the picture. The other is to

build a detailed performance dashboard. The dashboard lists performance statistics on certain capacity attributes measured against predefined thresholds.

To implement the application, the plan of the project entailed two phases. The first phase is a tool evaluating phase. Two major visualization tools – Graphviz and Microsoft Visio were assessed at this phase. Oracle Application Express (ApEx) was also tested as an application interface. After some analysis, the combination of Graphviz and ApEx proved to be the right solution. The project then moved on to the second phase which is a formal development phase. Full capacity of the two tools was explored at this phase and the final product was built.

## 2. Background

In this section, background of the company, BNP Paribas and its division, GECD are briefly discussed. The data to be visualized in this project is introduced and candidate tools and technologies that were used to build the application are reviewed.

### 2.1. BNP Paribas and GECD

A leader in global banking and financial services, BNP Paribas is one of the six largest banks in the world. The US site of the company has a very strong Corporate & Investment Banking business. BNP Paribas Global Equities & Commodity Derivatives, an arm of BNP Paribas Corporate and Investment Banking, offers custom-made derivatives on equity and commodity underlyings worldwide. It brings together three complementary business lines of Structured Equity, Flow & Financing and Commodity Derivatives.

- **Structured Equity:** provides structured solutions to a broad group of personal and business customers, banking networks, insurance companies and pension funds. It provides customized or exchange-traded structured products to meet their needs in capital protection, yield and diversification.
- **Flow and Financing:** covers products and services required by institutional investors to implement their investment, hedging, and portfolio optimization needs in a multitude of markets and underlyings. These products and services encompass flow derivatives, stock lending, prime brokerage and execution, as well as Asian equities research and brokerage.
- **Commodity Derivatives:** offers a full range of price risk management solutions on underlyings including energy, metals and soft commodities. The OTC (Over-The-Counter) group provides liquidity and market-making services, while the Futures team provides global clearing and financing tools for listed commodity futures and options to the corporate and institutional client base of BNP Paribas.

To support these important businesses, over 1400 front office staff work under GECD, numerous applications (systems) and servers keep running, and hundreds of

business flows are going behind.

## 2.2. Data of Interest

One important part of the project is the process data to be visualized. Process data contains valuable information about business flows. It consists of two parts – enterprise entities that constitute the business flows and relationships among entities that preserve business logic. There are five types of entity data.

- Application – system where business activities are carried out. A typical example is High Volume Trading Platform.
- Hardware – infrastructure unit that supports applications, i.e. switch ports, servers.
- Organization Unit (OU) – business component and department that is either a group of business users or an IT support team of applications or hardware, i.e. GECD, CIB. OU is usually an intermediate node in an organizational hierarchy.
- People – individual employees. They are usually end leaves in an organizational hierarchy.
- Process – collection of Applications, Hardware, and OUs. Process is an integral and independent subpart of a business flow.

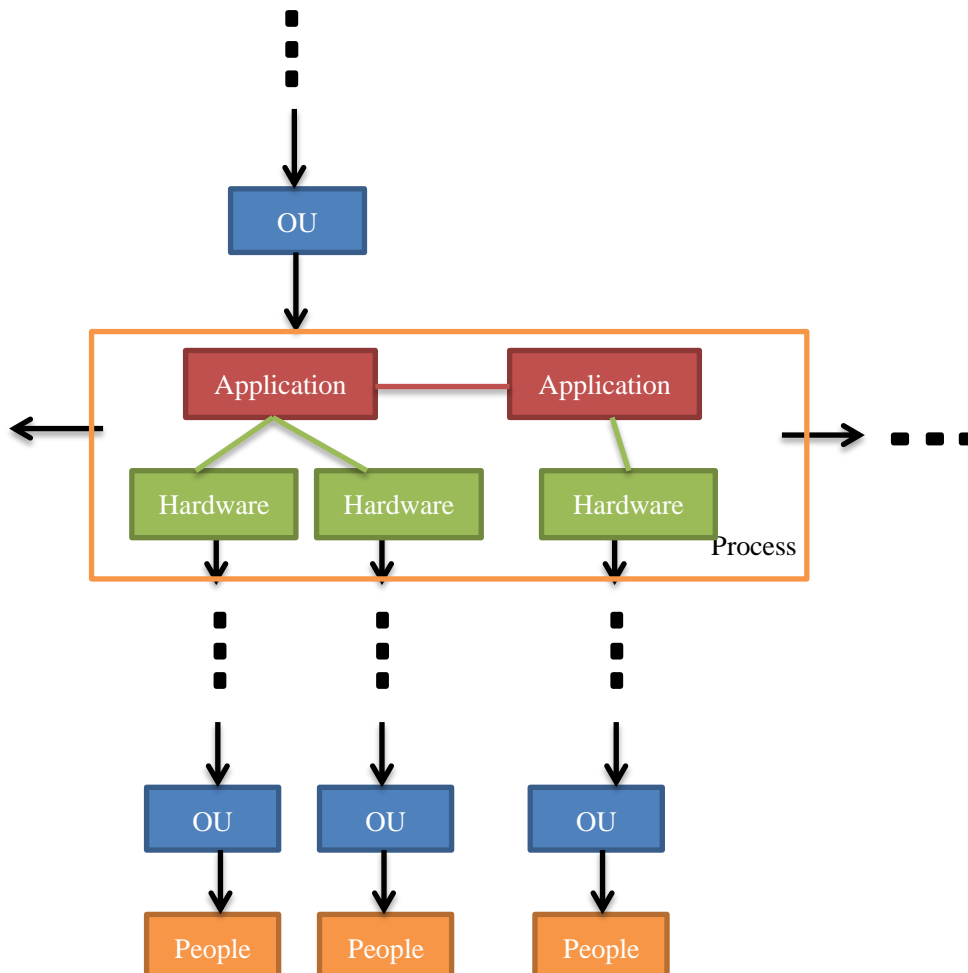
The most important entities are Application (system), Hardware (server), and OU. They make the majority of a business flow. The first two are also main targets for capacity analysis. Capacity dashboards are built around the performance of an *Application* or *Hardware* entity.

Another thing notable about an *Application* or *Hardware* entity is the distinction between its physical instances and the master entity itself. The master entity is a conceptual existence representing one particular type. For example, *MktFeed* is a type of *Exchange Feed* application. However, there is no single application called “MktFeed” running in reality. What actually perform the type of business operation are its physical instances. They are *NYCE feed*, *ARCA Feed*, and *Nasdaq Feed*, each responsible for one market. Each physical instance might be used by a different OU

and participate in a different business flow. Therefore, it makes sense to distinguish them to manage their capacity separately. It also makes sense to look at all instances as one master entity in a summary view.

Process is a relatively special entity among the five. It is a composite entity made up of the interconnected entities. The elements of a process act as a whole for one business purpose.

Various relationships exist between enterprise entities to indicate business logic. Relationships include hardware to application, application to application, hardware to organization unit, and organization unit to organization unit. Relationships make up chains and chains make up networks that are part of the entire business hierarchy. The simplified hierarchy can be illustrated by the following diagram:



**Figure 1: Hierarchy.**

The hierarchy starts with organization units, for example, BNP Paribas or GECD. These *OUs* usually take charge of some business process, which consists of



interconnected applications and hardware that support them. The process can flow into other processes or business entities at the same level. The process can also flow down into another group of *OUs*, usually the infrastructure teams that support the hardware in the process. The hierarchy keeps running down until reaches the ends where individual employees under a particular OU reside.

To see samples of real data, readers can refer to table *Computer System, Infrastructure, ProcessID* (for entities) and *Relation Classification, Map Infrastructure to System* (for relationships) in Appendix C.

## 2.3. Technologies Review

A variety of technologies contributed to the development of the project, either at the tool assessment stage or in the formal development phase. Five tools and six languages are included here. Below is a literature review to each of them.

### 2.3.1. Tools

#### **Graphviz**

Graphviz, short for Graphviz Visualization Software, is a set of open-source tools provided by AT&T Labs Research for drawing graphs and networks. The software takes in the description of the graph in a simple text language called DOT, which can then be rendered into a variety of useful formats, such as JPEG and SVG (for web pages), with different layout options including “spring model”, radial, and circular layouts. In the market, Graphviz is known as industry-standard graph visualization software. It has many important applications in networking, bioinformatics, software engineering, database and other technical areas.

#### **ApEx**

Oracle Application Express (Oracle ApEx) is a fast application building tool based on the Oracle databases (Oracle Application Express). It requires no installation on the user side and can be used as pure web-based development. It can be used for building departmental-style applications with a dozen users, but can also scale up to handle thousands of users (Oracle Application Express). The framework itself adds very little overhead to each page request so the performance is only affected by the

efficiency of the SQL queries built in the application.

### **Microsoft Visio**

Microsoft Visio is a commercial diagramming program for Microsoft Windows that uses vector graphics to create diagrams. It has developer capabilities which allow the application to integrate with Visual Basic to produce automated responses. Visio comes with a large number of shape templates that correspond to different popular designs such as implementing an employee hierarchy. Visio focuses a variety of simplistic measures while maintaining a graphically pleasing display.

### **Oracle Data Modeler**

SQL Developer Data Modeler is a unique, free data and database modeling tool, which provides a full range of utilities to support all data modeling needs. Data Modeler uses the .XML coding language to produce a graphical diagram. Data Modeler displays relationships between various entities, and allows the user to make simple data manipulations from the graphical level.

## **2.3.2. Programming Languages**

### **Python**

Python is a general-purpose, high-level programming language. It is designed to emphasize code readability. Its use of indentation for block delimiters is unique among popular programming languages. It has remarkable programming power which includes large standard library and comprehensive modules. Similar to Scheme, Ruby, and Perl, Python is also a dynamic language. It is often used as a scripting language, although it can be applied to non-scripting contexts. The reference implementation of Python is free and open source software and has a wide community-based development model.

### **DOT Language**

DOT is a plain text graph description language. It is a way to describe graphs. The files that use DOT language usually end with *.gv* or *.dot*. Many programs can process DOT files, including *dot*, *neato*, *fdp*, and *circo*. Most of the programs are part of the Graphviz package.

## **SQL**

SQL (Structured Query Language) is a programming language for managing data in relational database management systems (RDBMS). It has a variety of capabilities including data insert, query, update and delete, schema creation and modification, and data access control. It is the most widely used database language.

## **PL/SQL**

PL/SQL (Procedural Language/Structured Query Language) is Oracle Corporation's procedural extension language for SQL and the Oracle relational database. It is designed specifically for the seamless processing of SQL commands. Server-side PL/SQL is stored and compiled in Oracle Database and runs within the Oracle executable. As one of the three key languages embedded in the Oracle Database, it automatically inherits the robustness, security, and portability of Oracle Database.

## **HTML**

Technically speaking, HTML (Hyper Text Markup Language) is not a programming language. It is a markup language that uses tags to describe web pages. HTML documents are equivalent to web pages. Web browsers read these documents, interpret the content by HTML tags, and display them as web pages.

## **JavaScript**

JavaScript is a prototype-based scripting language. It is primarily used in the client side, implemented as part of a web browser to provide enhanced user interfaces and dynamic websites. Besides web pages, it has other applications such as PDF document.

### 3. Requirements

The requirement of the project came from three major areas:

#### **Dashboard on System (Application) Performance**

- System Monitor

#### **List of Systems**

- System Cartography
- Systems by Function
- Systems by Business Unit

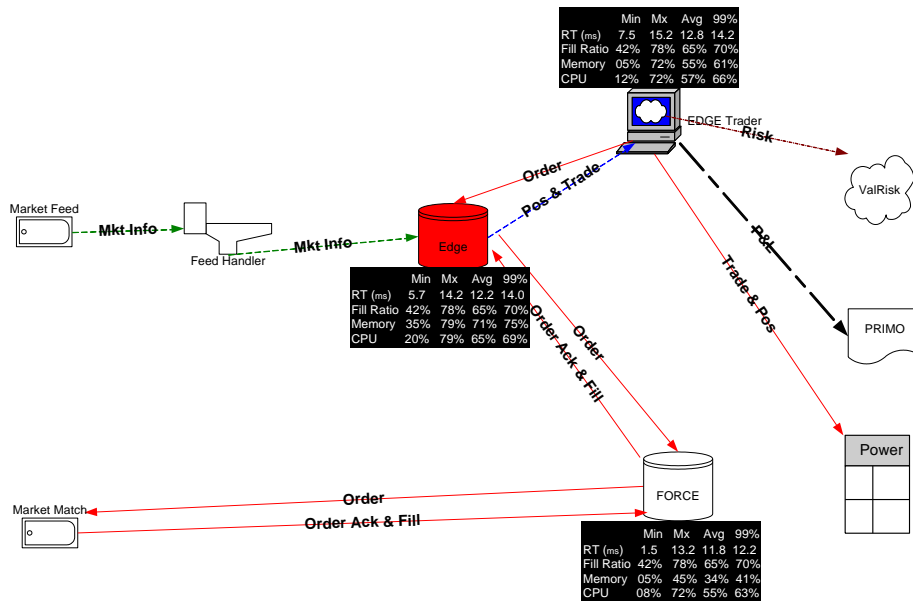
#### **List of Servers**

- Server Cartography
- Servers by System

(BNP Paribas. “Enterprise Architecture Overview.”)

The first requirement was to build dashboards to monitor system (application) performance. This was the step stone to capacity management. In order to build performance dashboards, the systems and the hardware (servers) that support them needed to be understood with the context of business flows. That was where the second and third requirements came in – to visualize the list of systems and servers. The visualization was not necessarily a messy picture with everything included but could be part of the entire organization within certain contexts. Systems could be viewed with the context of business function or business unit while servers with the context of system.

The requirements also included what the visualization ideally should look like. The following picture is a business flow of systems with performance dashboards.

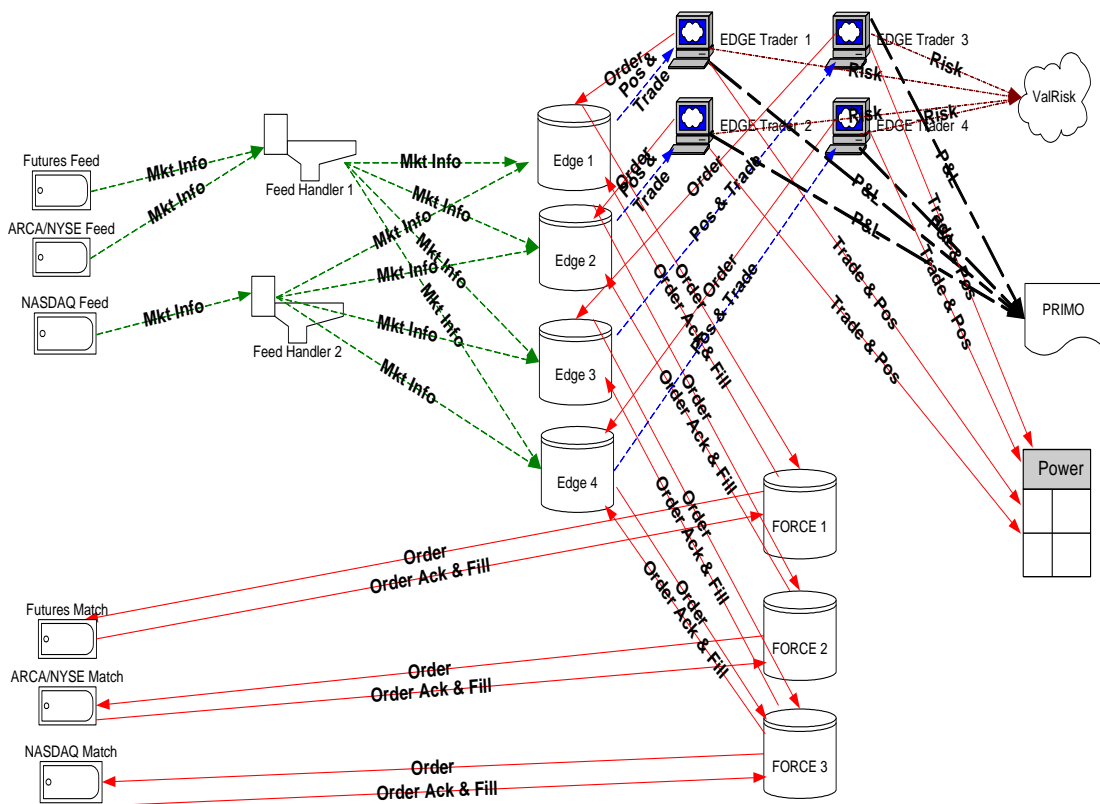


**Figure 2: Graph Example from Requirements.**

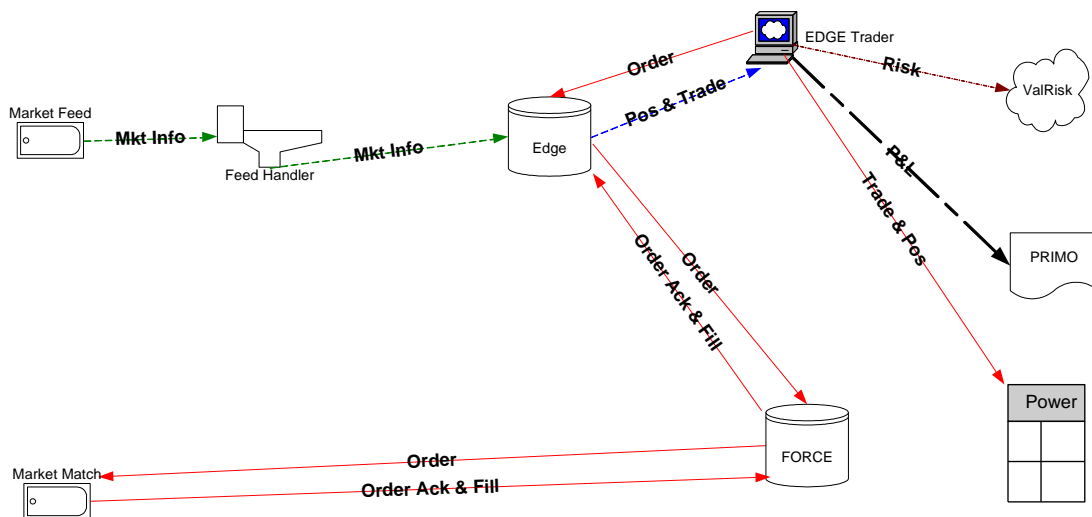
(BNP Paribas. “Enterprise Architecture First Phase – a tool for capacity management.”)

The example shows systems (nodes) and feeds (links) between them. Note that the three applications, *Edge*, *FORCE*, and *EDGE Trader* have their corresponding capacity dashboards (metrics) displayed by side. The dashboards summarize how the application performs within a period of time, measured by key aspects like Memory and CPU. One of the applications, *Edge*, is color-coded in red because based on the performance statistics it is diagnosed as currently over capacity.

Recall that the background section elaborated on the difference between a master entity of an application or a server and its physical instances. Correspondingly, requirements of the project also emphasized on the distinction between a summary graph and a detailed graph. The application that was to be developed was required to implement both of the two.



**Figure 3: Detailed View.** A detailed graph would display every individual instance. (BNP Paribas. “Enterprise Architecture First Phase – a tool for capacity management.”)



**Figure 4: Summary View.** A summary graph only shows the master entity. (BNP Paribas. “Enterprise Architecture First Phase – a tool for capacity management.”)

## 4. Business Uses

The requirements came from stakeholders each with their own need from the same application. In general business uses of the application can be grouped into three categories.

- Performance Management
- Personnel Management
- Technical Management

### 4.1. Performance Management

This is the most important business use of the three. If a server is over capacity, it gets reflected on the graph by the color. The manager sees the alert and wants to see what is going on with that server. He would be able to drill down to see the performance metrics (dashboards) built for that server. It is a log showing detailed information about how the server performs over a period of time. Any problematic record is highlighted so the manager could spot it at the first sight. Since the server is graphed within the context of a business flow, the problem is understood with reference to other entities the server is connected to. This use can be applied to systems (applications) and hardware (servers).

### 4.2. Personnel Management

One of the interested parties of the data is Human Resources. They are interested in organizational charting to improve personnel management. The graph shows a clear relationship between organization units and people under them. It is easy for HR people to see what position each employee falls in within the entire organization. Any personnel update can be reflected in the graph. The graph also shows what application or infrastructure component any employee currently has access to. If an employee moves from one department to another, what to deprive him of and what to grant him is easy to see from the graph. This helps to make access granting in a huge organization more regulated.

### 4.3. Technical Management

Similar to people in an org chart, processes can be shown in detail on the graph.

Managers can analyze the elements and the flow between them in a particular process. They can also see how one process interacts with other business flows. Together with performance management, managers are able to conduct efficiency analysis, optimize resources allocation, and expand or retract businesses.

A full list of interested parties and business uses can be seen in table *Interested Parties for Data* in Appendix B.



## 5. Development Phase I: Tools Assessment, Prototyping

### 5.1. Apex + Graphviz – in-house development

In this part of the tool analysis, Graphviz was tested as a graphing engine and Apex as a data management interface. In order for the two separate tools to act as an integral application, their interactivity with each other was also tested.

#### 5.1.1. Pre-analysis on the tool

##### Candidacy of Graphviz

- Embedded with optimized positioning algorithm
- Standard in representing structural information
- Rich in custom options

##### Candidacy of Apex

- Fast development
- Specialized in building applications from database
- Web-based
- Easy to deploy ( end users access the application through a URL)
- Easy to administrator ( scalable for thousands of users)

#### 5.1.2. Technology used

##### Programming Languages

- Python – automated the generation of input files in the DOT language.
- SQL – embedded in the Python script to make database queries. It was also widely used in Apex as the source for many standard components such as Interactive Report.
- PL/SQL – a major developing language in Apex together with SQL. It provided the source for Apex Processes and communicated between database references and application references.
- HTML – customized the displaying features both in Graphviz and Apex.

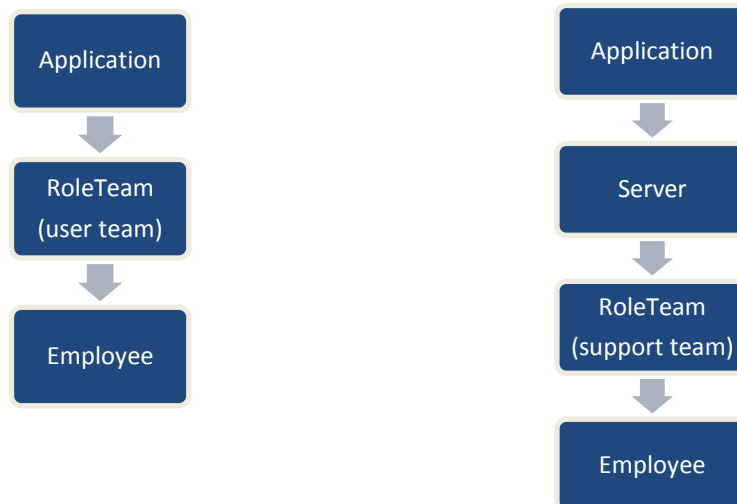
##### Software

- Python 2.7 – recommended bugfix release of Python.
- Graphviz-2.28 – latest stable release of Graphviz.

- Oracle Application Express 4.1 – latest release of Oracle Application Express
- Oracle SQL Developer – manages the data in database from a development level.

### 5.1.3. Database and Sample data

For the purpose of evaluating the tool, an Oracle database called “GECD\_EnterpriseArchitecture” was set up, along with nine tables filled in. There were four types of Enterprise Entities: Application, Server, RoleTeam, and Employee and five types of relationships: Application to Application (AA), Application to RoleTeam (AR), Application to Server (ASERVER), Server to RoleTeam (SR), and RoleTeam to Employee (RE). There were two major relationship chains, one for business user teams, and the other one for infrastructure teams. They can be illustrated by the following flows.



**Figure 5: Two Flows in Graphviz testing.**

Although the sample data used at this stage was largely simplified from the real data, it preserved the essential hierarchical structure of the business flows and sufficed for the testing purpose.

### 5.1.4. Development in Graphviz

#### 5.1.4.1. Database connection

cx\_Oracle, a Python extension module, allows access to Oracle and provides Python database API (Application Programming Interface). The following code used the module to establish the connection and fetch the data.

```

Import cx_Oracle
conn = cx_Oracle.Connection ("EA_APP/EA_APP@GECD_EnterpriseArchitecture")
Cursor = conn.cursor()

```

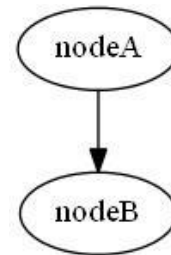
#### 5.1.4.2. DOT language and the graph

This section is to illustrate what basic format of file Graphviz takes as input and what the outputted graph looks like. In sample.txt file, write a piece of code as follows:

```

digraph myGraph {
nodeA;
nodeB;
nodeA ->nodeB;
}

```



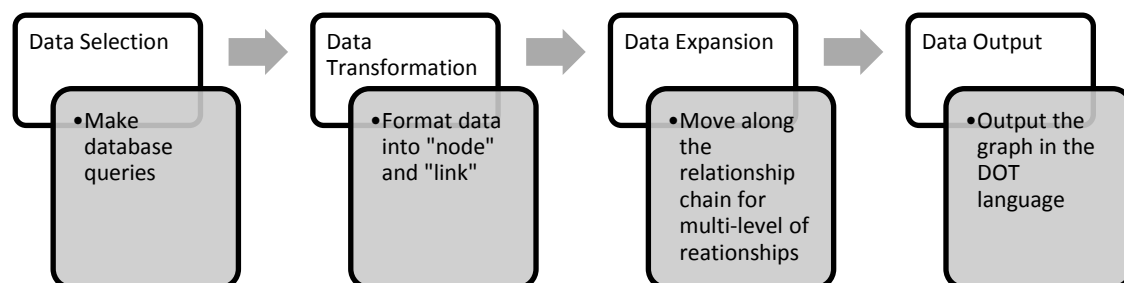
**Figure 6: Sample Graph I**

In the example, myGraph is declared as a digraph, which means links are directed with arrows pointing the direction. Inside ‘{}’ is where the nodes, links, and their attributes are defined. Here two nodes, nodeA and nodeB, and one link from nodeA to nodeB are created in the graph.

Instead of manually inputting the data and formatting it in the text file, python code automated this process and generated the graph systematically.

#### 5.1.4.3. Automation Procedure Architecture

The following flow shows the automatic drawing process that was coded in the Python script.



**Figure 7: Automation Procedure Architecture.**

#### 5.1.4.4. Implementation Design: move along different levels of relationships

Structural information of interest does not usually consist of one level of relationships. As to graph a multi-level relationship correctly, for example the

relationship of application-server-role/team, a way to pass the result of the first SELECT statement (for application - server) to the second statement (for server-role/team) was necessary.

There were two possible ways to pass the variables. The first one was to select all the data needed for graphing in one single chunk of code. This was equivalent to creating a master table with all the entities and linkage information assembled. The second option was to have each relationship in its individual SELECT statement implemented in separate functions in Python and pass the variables in function calls. In this way, the 'variable passing' was done dynamically at runtime.

The first way was technically simpler. However, it had the major drawback of inflexibility. Simply by varying the number of levels to visualize in the graph and the structure of the relationship chain could produce many different graphs. Writing SELECT statements for each of them was not a good practice of software engineering. There was a lot of repetition involved. The code was also hard to maintain in the long term and scale poorly. In contrast, the second way was much more flexible. It dealt with one level of relationship at a time and generates multi-level graphs by reusing each level and put them together. The variation was encapsulated at a higher level where it was easier to accept user input and dynamically created new combinations.

The second design option was implemented. Each relationship had its own function where only one level of data was selected. The function prepared for next level data selection by returning the list of destination node ids. When the next level function took in the list as a parameter, it used the list as part of WHERE clause and began another cycle of selection. Here is some sample code:

```

1
def selectionAR(appIds, appNames = None):
    idstring = formatIDs(appIds)
    namestring = formatNAMES(appNames)
    return"""
        select
        A.NAME,
        A.A_ID,
        (SELECT NAME FROM ROLETEAM R WHERE R.R_ID=AR.ROLE) TEAM,
        (SELECT R.R_ID FROM ROLETEAM R WHERE R.R_ID=AR.ROLE) NEXT
    from APPLICATION A
    LEFT JOIN AR
    ON A.A_ID= AR.APP
    where A.A_ID = ANY {0} OR UPPER(A.NAME) = ANY {1}""".format(
        idstring, namestring)

```

**Figure 8: Selection Function.** The function took application ids as input, formatted them and used them in the selection statement.

```

1
def drawAR(cursor, wfile, hilite = False):
    nextIds = [];
    for col1, col2, col3, col4 in cursor.fetchall():
        wfile.write(drawApplicationNode(col1, col2))
        wfile.write(drawRoleNode(col3, col4))
        wfile.write(drawLink(col1, col2, col3, col4))
        nextIds.append(col4)
    if hilite:
        wfile.write(highlightNode(col3, col4))
    return nextIds

```

**Figure 9: Process function.** The function recorded this level of information to the output text file and returned the nodes for next level.

#### 5.1.4.5. Functions

By the end of coding in python, the following aspects were explored and (or) implemented.

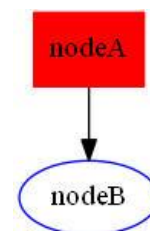
#### 1) Customized colors and shapes for different levels.

In terms of varying the appearances of nodes and edges, Graphviz has a variety of options to choose from including shapes, colors, and styles (Node Shapes). Here is a simple example to illustrate how it works.

```

digraphmyExample {
nodeA [shape = box,color = red,style = filled];
nodeB [shape = ellipse, color = blue, style = unfilled];
nodeA ->nodeB;
}

```



**Figure 10: Sample Graph II.**

The Dot language provides over 100 types of attributes to manipulate with (Node,

Edge and Graph Attributes). The full list can be found the documents in Graphviz.Org.

## 2) ‘Search’ functionality

Users could select the items they wanted to start the visualization with by providing the python program with command line arguments. The program implemented a complete set of Application Programming Interface (API) ready to use. No matter what type of application would be used in the future to accept user selection, as long as the type and an id list of nodes were supplied, the program could generate the desired graph automatically.

## 3) Expand in three directions: upward, downward and in the same level

Starting from any level with any node, the graph was generated systematically by expanding upward, downward, or remaining at the same level.

### 5.1.4.6. Picture Rendering

Graphviz supports a variety of graph formats. To generate a specific type of format, for example, the JPEG format, navigate to the directory where the dot executable sits and use the following command:

```
dot -Tjpg sample.txt -o samplePic.jpg
```

For other formats, substitute `-T{jpg}` with other `-T` options. For example, Svg files can be generated by using `-Tsvg`.

### 5.1.4.7. Sample code and pictures generated

See Appendix A.

## 5.1.5. Development in ApEx

Very different from the development in Python, development in ApEx involved little build-from-scratch work. Most of the time, developers clicked on buttons, specified attributes from dropdown boxes, and wrote small piece of code. ApEx did all the underneath “dirty” work and provided developers with fancy-looking in-built components with powerful capacities ready to use.

### 5.1.5.1. Development Preparation

It is very easy to develop in ApEx. All a developer needs to do is request a

workspace from Oracle and get the username and password set up. After that, login in from the login page on Apex.Oracle.com website and start developing.

#### 5.1.5.2. Most important features in Apex

The following is a walk-through of the most important features provided by Apex.

### Interactive Report

Perhaps the most important and useful feature in Apex is Interactive Report (IR). It displays a table of records stored in database with columns as the developer chooses. The content of the table comes from a single SQL SELECT statement. The Report is very easy to create. A couple of button clicks and attribute specifications produce a very standard looking report. Functionalities like search/filter, sort, and edit each record come automatically with nice looking interface.

The following SELECT statement made a report as below:

```
SELECT AA.ID,
       AA.SOURCE,
       (A1.NAME) FROMNAME,
       AA.DEST,
       (A2.NAME) TONAME
FROM AA,
     APPLICATION A1,
     APPLICATION A2
WHERE AA.SOURCE = A1.ID AND AA.DEST = A2.ID
```

Id	Source	Fromname	Dest	Toname
65	4	FORCE	2	ExchangeMatch
81	1	MktFeed	3	ValRisk
1	1	MktFeed	2	ExchangeMatch
2	2	ExchangeMatch	3	ValRisk
101	2	ExchangeMatch	1	MktFeed
41	3	ValRisk	2	ExchangeMatch

**Figure 11: Interactive Report.** The report displays all the records in table AA (Application to Application) with their names aside.

The major part of the report, the table-like assembly of records, closely resembled the table AA in the database with more user friendly interface.

The Interactive Report has three inherent functions.

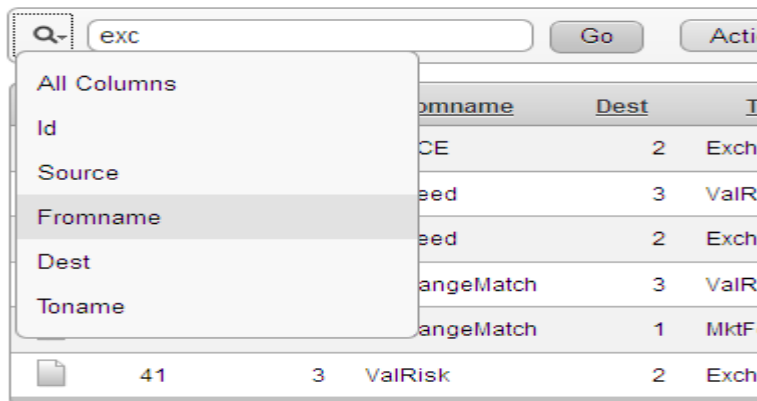
- Search and Filter.

A search on any column of the report can be performed in the search bar below.



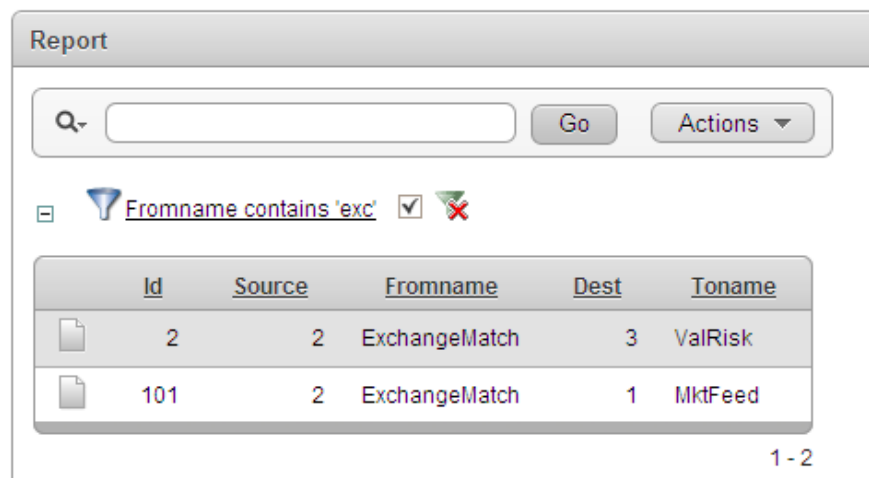
**Figure 12: Search Bar.**

Click on the magnifier to select the column name and type in the string to be contained in the search results. For example, if the user wishes to search for all the records whose *fromname* contains 'exc', choose *fromname*, type in 'exc', and click the button *Go*:



**Figure 13: Search.**

Now the report only displays the hits from search result.

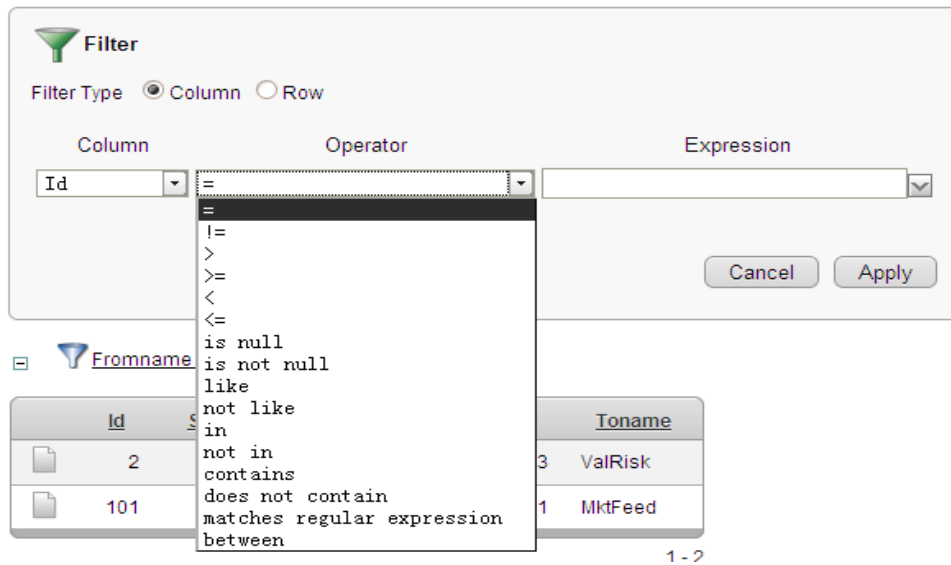


**Figure 14: Interactive Report after Search.** It only displays hits from the search.

If the user wishes to perform a more precise filter on the data, he /she can go to





Actions and select *Filter*. This opens up a new Filter window where the user can specify the filtering criteria.




**Figure 15: Filter.**

- Sort

Sorting on any column is easy. Click on the column name and select  or  for ascending or descending sorting.

- Edit on the records

Click on the icon  and the user is directed to another page with all the detailed information about that particular row of record assembled and ready to edit. This is closely related to another powerful component in Apex - Form.

## Form

Forms are usually used to edit one single record from a database table. They can be created in bundle with an Interactive Report for a table (*form page on table with report*), or by themselves. In either way, there is a way to tell the form page what record to load from the report. A Form typically looks like this:

The screenshot shows a dialog box titled "Form on AA". At the top right, there are two buttons: "Cancel" and "Create". Below the buttons, there are two input fields. The first field is labeled "\* Source" and the second is labeled "\* Dest". Both fields are currently empty.

**Figure 16: Form for Creating.** This is for creating a record.

The screenshot shows a dialog box titled "Form on AA". At the top, there are three buttons: "Cancel", "Delete", and "Apply Changes". Below the buttons, there are two input fields. The first field is labeled "\* Source" and contains the value "4". The second field is labeled "\* Dest" and contains the value "2".

**Figure 17: for Editing.** This is for editing a record.

If the form page is not created together with the report page, the developer has to establish the link himself. In the *Link Column* section of *Report Attributes*, choose the *Target as Page* in this Application and specify the page number of the form page. In the *Name and Value* fields, specify the Item name on the form page and the corresponding value to send from the report page. In this way, the form page knows which row of record is coming and what information should be loaded. Normally, the primary key of the table is the “messenger” and all the other fields of information are filled in automatically.

The analysis on the Interactive Report and Form showed Apex as a promising data management interface builder. IR displayed all the records of a table and Form allowed creating and editing any particular record for that table. The combination of these two fulfilled the need as to see and change the data. The powerful search/filter ability of IR also enabled managers to quickly locate a single record or a group of records without much effort. The selected records could later be used as the input for Graphviz to begin graphing with.

### 5.1.5.3. Advanced Interactive Report implementation

In terms of how managers want to use the Interactive Report to select some of the data to visualize, the standard IR lacks a way to indicate the selection.

Research showed that the right solution to this challenge was to use checkboxes as a separate column in the report. A PL/SQL package called `htmldb_item` was used for this (APEX Check Boxes in Reports Regions). The package creates form elements dynamically based on a SQL query. The following code demonstrates how to incorporate the `CHECKBOX` function into SQL statements.

```
SELECT HTMLDB_item.checkbox(1,id) CheckBox,  
       r.id,  
       r.name  
FROM ROLETEAM r
```

Note that in the first line, the `CHECKBOX` function took in two parameters. The first one was an index determining which `htmldb_application` global variable was used. For example, `1` indicated variable `F01` and `2` `F02`. The `htmldb_application` global variable was the variable that kept track of the column of the report and could be referenced elsewhere in the application. The second parameter was the value of the checkbox, in this case, the `id` for table `ROLETEAM`. Below is the screen shot of the `RoleTeam` report with checkbox added.



**Figure 18: IR with CheckBoxes.**

To illustrate how the checkboxes work, imagine the user checks records with `id` 1, 2, and 4. Since there are three records selected, `htmldb_application.F01.count` is 3.

The *htmldb\_application.F01 (1)*, *htmldb\_application.F01 (2)*, and *htmldb\_application.F01 (3)* are 1, 2, and 4 respectively. In this way, the selection was well indexed and could be referenced somewhere else in the application.

The Checkbox solution provided an intuitive way for the user to select the records and for the interface to accept selection. By using the *htmldb\_application* global variable, selected records could be referenced and prepared for the next step of action.

#### **5.1.5.4. More Research on Apex**

This section devotes to additional exploration in Apex at the stage of tool testing.

##### **1) More page Item types**

The default item types Text Field and Number Field were not so useful because users had to remember the entire string in order to edit. In such cases, Select List, Popup List of Value and Text Field with autocomplete were more helpful.

- Select List - a non-editable dropdown box
- Popup List of Value - a Select List with a pop up window instead of a dropdown
- Text Field with autocomplete - gives hints while users type in the string.

All three of them helped to make the application more user-friendly.

##### **2) Print out dynamically changing text**

When the developer tested on the usage of the checkboxes, there was a need to print out the values of *htmldb\_application.F01* variables. It turned out that Apex has a region type called 'dynamic content region' where the dynamically changing source can be displayed by function *htp.p*. The region was also used for debugging purposes.

#### **5.1.6. Two way interaction**

##### **5.1.6.1. From Graphviz to Apex**

The biggest obstacle that prevented Graphviz + Apex from being the right tool for the project was the static nature of the graph generated by Graphviz. It was believed by all the testers and developers that the graph provided no way of user interaction until the midpoint of the project.

The major breakthrough came from the finding that a URL link could be added to a node in the same place where customized attributes are defined. It was a very faint connection between the Graphviz and the outside world but it was also a very brilliant idea.

At the first phase of the exploration, a simple URL ([www.google.com](http://www.google.com)) was added to every node. The modified sample script looks like this:

```
digraphmyExample {
nodeA [shape = box,color = red,style = filled, url = "www.google.com"];
nodeB [shape = ellipse, color = blue, style = unfilled, url = "www.google.com"];
nodeA ->nodeB;
}
```

Research showed that the “hot region” for a node is defined as the entire area bounded by its shape. However, not all of the graph formats are clickable. As the Graphviz documentation from Princeton.Edu (Graphviz – QED) points out, URL links are only supported in ps2, cmap, imap and svg formats. Svg format was chosen later because it could be opened in a web browser.

The URL link was then customized based on the id of the node being clicked. After it was found that it was possible to embed the id in the link, the URL was changed to the path of a particular Apex page. The idea was to direct the user to the editing page for that particular node.

### **ApEx page URL**

In order to pass in the parameters to the right place of a URL, it is important to understand the constitution of an Apex page URL. Note that most of the Apex application URLs have a part starting from f?p. There are usually 9 parameters that can be specified after.

f?p = 1:2:3:4:5:6:7:8:9

- 1- The application number. To access the current application id, use &APP\_ID..
- 2- The page number.
- 3- The session. To access the current session, use &SESSION..
- 4- The request to pass to the destination page.

5- DEBUG. Decide whether the page is run in debug mode or not. either YES or NO

6- Two values, 'RP' for indicating resetting pagination and the other for pages where cache is reset.

9- Printer friendly. (URL Parameters)

Parameter 7 and 8 are very important. Parameter 7 is a comma separated list of page items ready to be assigned the values from the comma separated list of values specified in Parameter 8 (URL Parameters). As in the case here, the URL attached to any node could be

[http://nycs00057970:8080/apex/f?p=100:9:&SESSION:::NO::P9\\_NODEID:4](http://nycs00057970:8080/apex/f?p=100:9:&SESSION:::NO::P9_NODEID:4) (Oracle Application Express APEX: Passing Values between Screens)

Here the URL linked the node to page 9 of application 100. An Item on that page called P9\_NODEID was set to 4. The page 9 was a form page for editing a node. By passing the value 4 to the node id on that page, the information of that record could be pulled out and loaded into the fields of the page, ready for editing. In practice, the value 4 could be substituted by any variable holding the id. Setting the Parameter 7 and 8 of a page URL like this was extremely critical and useful for the communication between Apex and the outside world and within Apex itself.

Therefore, by manipulating page URL manually and incorporating it in the input dot script for Graphviz, the one-way interaction from the graph to the data management interface was established.

#### ***5.1.6.2. From Apex to Graphviz***

The ideal way of user interaction from Apex to Graphviz was to have a button trigger the drawing procedure. The procedure took the records selected as input and ran an OS command based on that. A button and a triggering process were not difficult to implement in Apex. However, how to invoke the OS command from Apex was a problem. There were basically two ways to do this. One was by external procedure call, and the other was by dbms\_scheduler job. In both cases, the command

would be executed on the same server as the Oracle database resided. After some research, external procedure calls turned out to be more complicated. Therefore, `dbms_scheduler` was chosen.

The possibility to achieve two-way interaction was critical in the assessment of the tool combination of Apex and Graphviz. After this founding, for the first time the tool combination was proven a feasible option.

#### 5.1.6. Evaluation

Graphviz and Apex have many attributes that made them strong candidates for the project.

##### **Merits on Graphviz**

Based on the pre-analysis, Graphviz was proven to have:

##### 1) Embedded positioning algorithms

By simply specifying in command argument major layout to use (Using GraphViz, a Brief Tutorial | Orient Lodge), the user gets a nicely displayed graph. Graphviz does all the intelligent work behind the scene. This largely reduces the amount of programming work that needs to be done.

##### 2) Great scalability

Graphviz is the industry standard in representing structural information. Its positioning algorithms optimize the arrangement of the nodes and minimize the overlapping of the links. This largely increases its scalability for a huge amount of data, which is the amount the developers would be faced with.

##### 3) Various attribute options

As shown in sample code and graph, Graphviz provides a variety of attributes including colors, fonts, line styles, and custom shapes that help distinguish between different types of nodes and links.

Based on the python program, Graphviz was found to assist in automating the process of feeding data and drawing. As opposed to many other graphing engines, it requires no more special file format than `.txt` as input. The syntax of its language,

DOT, is also simple and programmatic. Any programming language can generate the format automatically without much effort. Furthermore, the separation of the input, which contains the pure master data, and the rendering process, which outputs master data together with attribute data, makes the automation even simpler. Developers can plug their own data at the point of providing the input file without affecting the step of rendering. Recalling from the requirements, one of the basic components is to automatically pull out data and graph it. Graphviz satisfies the requirement easily.

### **Merits on ApEx**

ApEx was chosen to complement Graphviz both as a data management tool and a control panel. Apart from those listed in the candidacy section, the following points are stressed here:

#### 1) Standard database application builder

ApEx is designed for database applications. The source of many components in the application is a single SELECT statement. This specialty easily satisfies the need to build a data management interface above the underlying database.

#### 2) Easy deployment

No installation on the user end is required. As soon as a username and password are granted by the administrator, the user can login in and play with the tool. This shortens development cycle and allows easy enhancements even after the application is in use.

#### 3) Easy development

ApEx provides standard components (i.e. the Interactive Report) that come with powerful functionalities. All the developers need to do is specify attributes of the component by filling out forms and selecting from dropdown boxes. It is much faster than the standard programming way of building up bit by bit in a programming script.

#### 4) Powerful administrative management

As the requirements pointed out, the project was designed to incorporate the need from different groups of users. In terms of this, ApEx has powerful administrative and security management capacity. In the *Administration* section of the application,



different users can be granted different set of permissions. Many components of the application also have a *Security* section which prevents certain group of users from using it.

#### 5) Multi-language supported

ApEx supports not only one language. To interact with database, it uses SQL and PL/SQL. To render pages within the application, it supports html and javascript. The latter two languages are also widely used in tweaking the appearance of a standard component and defining processes or functions to link components.

#### **Merits on the combination**

In terms of the combination of the two tools, the bi-directional interaction of Graphviz and ApEx was established one way by hyperlinks and the other by stored procedures and `dbms_scheduler`. In this way, at least on a preliminary stage, Graphviz and ApEx could be integrated into one united piece of software.

#### **Drawbacks on the combination**

There were certainly some downsides in this option. The most obvious one was the limited user interaction that can be performed on the graph. Most of the desired actions like create a node, delete a link, and drag and drop could not be done on the graph. Although the hyperlink approach guaranteed at least some communication between the graph and its underlying data, the way it worked was still the least intuitive. The other drawback lay in the development in ApEx. On the one hand, doing all the dirty work for developers saved development time and was good for creating standard-looking components. On the other hand, however, it hid necessary details from developers. Since developers did not know the code that implements functionalities like search/filter in the IR, it became much more difficult for them to extend and customize the functionality. In most cases, developers did not even know where to change some of the attributes although there were actually options in the dropdown boxes. This could affect the learning curve of the developer and become a problem at the early stage of the development. The third problem of Graphviz and ApEx was their separation. Although they could be connected, the combination was

still not functioning as an all-in-one-spot application. Binding them strongly and embedding the graphs in Apex page was difficult but definitely the next step to go.

## **5.2. Visio – in-house development**

### **5.2.1. Hierarchical Design**

The second application investigated Microsoft Visio. Visio was chosen because it has the capability of heightened user interaction and increased graphical displays making it a more viable option. Visio has several interfaces that vary in terms of functionality and automated display. Each interface corresponds to a different popular template. For example, one can make an employee hierarchy with the ‘Hierarchy Template’. Initially a hierarchical design, which was set up to process flows and nodes in a user friendly design, was tested. However, the hierarchical design did not provide the user with the level of freedom that would be needed to complete future stages of the project.

### **5.2.2. Free Form Design**

Once the hierarchical design was deemed too limiting, a free form design was investigated. This free form design gave the user complete freedom. With this, a VBA macro was developed that allowed the user to automatically display a node and evenly placed and connected dependent nodes around it. However, flaws were found in this freeform design as well. The free form method would make it difficult to process large amounts of information, and the information that was processed would be confounded to a point that didn’t require unlimited user freedom.

### **5.2.3. Semi-Free Form Design**

Finally an option was developed that allowed the user to place nodes freely and had several organized ways of connecting nodes.

A tutorial can be viewed at:

<http://support.microsoft.com/kb/309603>

This tutorial demonstrates how to create a template for programmability with Visio. It creates a simple program that opens up a Visio document, pastes an object on the screen, saves the documents and quits. Visio VBA language was the main component of this option. Initially this presented a challenge because there is not much information available pertaining to this language, aside from the tutorial discussed earlier. However, the VBA Visio application allows the programmer to

record macros. Therefore, the user has the ability to see a script of their actions within the programming language which is a key component for developing this application further. This feature also allowed for the discovery of the key VBA Visio commands which were implemented throughout the rest of project.

#### **5.2.4. Methodology**

##### ***5.2.4.1. Algorithm For Application Development***

To develop the VBA Visio application, a table with employees and their direct superior was saved in an Excel file. The information from this file was then imported into Visio so that it was incorporated into a hierarchical display. To ensure that the employees were displayed in a tight format that avoids overlap an algorithm was developed. The original algorithm found the root of the hierarchy and gave that employee a level of 1. The algorithm then went through subordinates and gave each node a level which corresponds to its distance from the root. Next, the algorithm would evenly space the most populous level and apply the space to the entire hierarchy. It then connected all subordinates of the most populous levels, readjusting the superiors to avoid overlap. Finally the algorithm went through the data, level by level, placing superiors above the average of its subordinates until the boss was placed in the top middle. This algorithm provided a visually sound diagram of the hierarchical structure of the company.

##### ***5.2.4.2. Data Presentation***

The next stage of VBA Visio development was to take a more comprehensive set of tables from a sample database that we created, process them through Microsoft Queries into VBA Visio and present them with Visio. This new set of data contained nine tables saved in an Excel file: 4 tables listing nodes (Applications, Servers, Teams and Employees) and 5 tables listing links (Application to Application, Application to Server, Application to Team, Server to Team and Team to Employee). The highest level would be the Application level. Applications had servers and teams associated with them. Servers had teams associated with them and teams had employees associated with them.

The goal of this stage was to give the user the ability to search for an individual node of interest. The Visio application would automatically display all of its dependents and superiors, and for applications it would show other applications that the chosen application interacted with.

To achieve this goal the first step was to create a master table using Microsoft Queries that incorporated all the information from the nine tables discussed earlier. Each row of the master table displayed every employee's name and ID with the names and ID of their team members, server (if applicable) and application. The query would run through the table and the table would be saved as an Excel table which was then called upon by the VBA Visio application.

After the specific excel table, constructed based upon the initial query, was imported by VBA Visio, the information from the query needed to be streamlined directly into the VBA Visio code. A connection was established to allow VBA Visio to run query scripts and then each line of output from the query was saved into a master array in VBA Visio. A complimentary step involved replacing the home of the data tables from Excel to Oracle. A connection had to be established between Microsoft Queries and Oracle and the queries had to be tweaked to accept data from a different source.

#### ***5.2.4.3. Algorithm Optimization***

The next step was to update the pre-existing algorithm to handle the new dataset. Fortunately, the levels were predefined by node type making the update possible. Initially, if an employee worked for two teams he would only be displayed once and he would ideally be placed in a way that made sense for both sub structures. However making this ideal situation a reality was very difficult because it involved creating an overlap adjustment algorithm which shifted the position of overlapping nodes while trying to minimize the width of the entire graph. After experiencing many different complications with this overlap adjustment algorithm, it was decided that if a node was part of multiple superiors it would be displayed multiple times on the graph. This allowed the process to be considerably simplified and was more user-

friendly.

#### ***5.2.4.4. Increased User Ability***

In the original algorithm, the user only had the capability to select an application and the application and its dependents would be displayed. The updated algorithm involved going through the master array and assigning a 'y location' to each node associated with the selected application, based upon the node type (employee = 1, team = 2 etc.). Then, the algorithm would go through all employees associated with the application and assign them 'x locations' in sequential order. Since the master array was sorted in the query, this was a relatively simple process and there was no chance of crisscrossing nodes. Next, the teams, servers and applications were placed above the average of its dependents similar to the original algorithm. Finally, a rectangle was created for each node and placed in its respective place given its x and y location.

Along with effectively organizing nodes, the Visio application offers several types of linking options. Links can be completely manually created by giving an x and y coordinate to the start and end points. The links can snap into the bottom, top, left or right sides of two nodes or a new node and a link can be created as a relationship to an existing node. It was determined that the snap method was the most effective and provided the most freedom while not over complicated this part of the project. This feature was then incorporated into the algorithm so that went through each node starting with employees and snapped a link between its superior and the node.

Additional logic was added to allow the user to have the root be something other than application. This new logic also gave the user the ability to display the application to application level visually. A smaller query was installed into VBA Visio that made an array containing application to application information and each node was given an x location that corresponded to its sequenced ID number and centered around the selected application. Finally, each node was given a color that corresponded to its node type. This optimization completed the second phase of the project.

### 5.2.5. Bi-directional Interface

The next stage of the project was to allow the user to make changes to the graphical display. These changes must be designed to change the information stored in the database. It was thought that this would be promising for the Visio application because it has highly presentable interface. To make Visio a bi-directional application, three different possibilities were investigated.

The first option involved creating a control panel. This control panel was based on the display produced by the algorithm. The user is able to add, remove or edit a node or connection, or even re-center the graph around another node. When a control is selected it would reprint the graph, while simultaneously updating the database. The second option was a scanning method. The user would be able to make changes to the visual interface and press a button that scanned the graph. The program would compare all the objects to the pre-scanned graph and update the database with all changes. The final option was to record keystroke events taken by the user and decipher what the user changed and update the database.

The control panel option was dismissed, because it would be easier to create a control panel using ApEx/Graphviz. Therefore it was determined that this was not worth pursuing. The scanning method would only be viable if there was a small amount of data, otherwise it would be too slow to be an effective method. The key stroke option proved to be the most optimistic as it did only the necessary amount of work to complete the task. However, it is important to note that if the database was smaller, the scanning method would be easier to implement. More information about the possibilities of a small database application is mentioned in the conclusion of this section for future use.

Based upon the assessment above, the most effective way to record keystroke events was to use the already built in 'record macro function.' The record macro function allows the user to make changes to the interface while simultaneously saving VBA Visio code that would update to include the changes that you made. This means that if the user reverted back to the pre-change graph and ran the recorded macro it

would produce the same results as the manual changes did. The user could press a button that would save the old version, press record macro, make necessary changes, stop the recording and, press a button to save the new information. Ideally this could be shortened to pressing a button that saves and starts recording, make changes and press another button that stops recording and saves information. However, those capabilities were not reached. Instead, when recording was stopped, the user presses a save button that saves the recorded macro into a text file and then imports the text file into VBA Visio. VBA then searches the text file for triggers that tell the program that the user added, removed or edited nodes or connections. It would then take this command and update the database.

Work was done to explore the display ability of this application. When the user searches for the initial node structure, the nodes would be placed behind four translucent different colored screens corresponding to each node type. If the user wants to manipulate the data, they must click on the 'prepare to edit' button which saves the file as it is and brings the white nodes over the translucent barrier. This enables the user to move, delete and change the nodes and connections. When changes were done being made, the user can press 'commit' and the database would be updated and the nodes were again placed behind the barrier.

Finally, when a node was not in edit mode, the user is able to click an invisible button located above barrier and node itself. Clicking on this button would bring up all necessary information concerning the node as well as collapse, expand and bring to front options.

Although Microsoft Visio has many advantages like display ability and high functionality, it wasn't chosen as the graphical application for the Enterprise Architecture problem. This was because it is not a web based application, and can only deal much smaller amounts of data than needed for this project. Visio would be a great application for a more containable project involving a single team or process.



## **5.3. Oracle Data Modeler – in-house development**

### **5.3.1. SQL Developer Investigation**

SQL Developer was the first application used to achieve this project's goal. SQL Developer was chosen for this project because it allows the user to drag nodes (processes and applications) onto a screen and connect them using flows. After the user makes changes to the information of the screen it will automatically be saved as a .xml file. The user has the ability to to manipulate the .xml code to hide and show nodes, delete and create links, and change their source and destination through manipulation of the .xml code.

### **5.3.2. SQL Developer Conclusions**

Although this application was more interactive with the user, it was not capable of deleting and creating nodes through the .xml code. These are two major flaws in this program which makes the application less desirable. After investigation into the SQL Developer application, it was determine that the user could not control the graphic interface through the basic code and as a result was not pursued further.

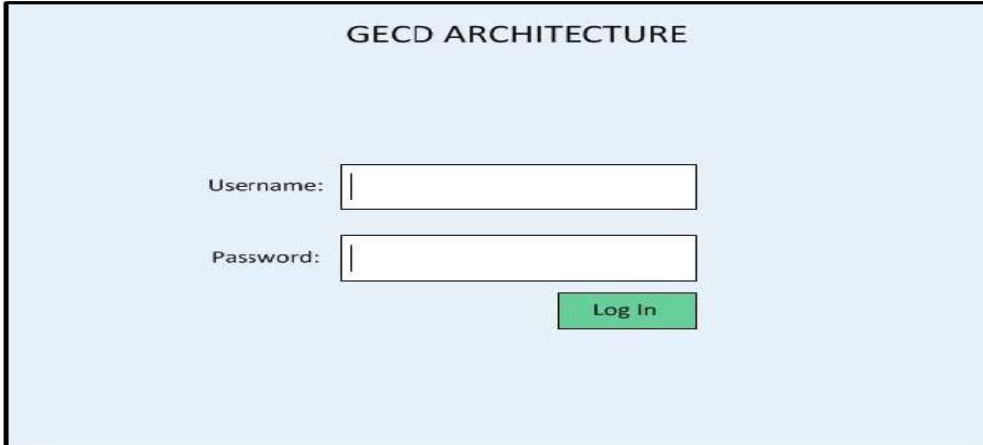
## 6. Development Phase II: Formal Development of Final Product

### 6.1. Detailed Design Plan

#### 6.1.1. User Interface Design

Based on the requirements, the following user interface was designed to be implemented in Apex.

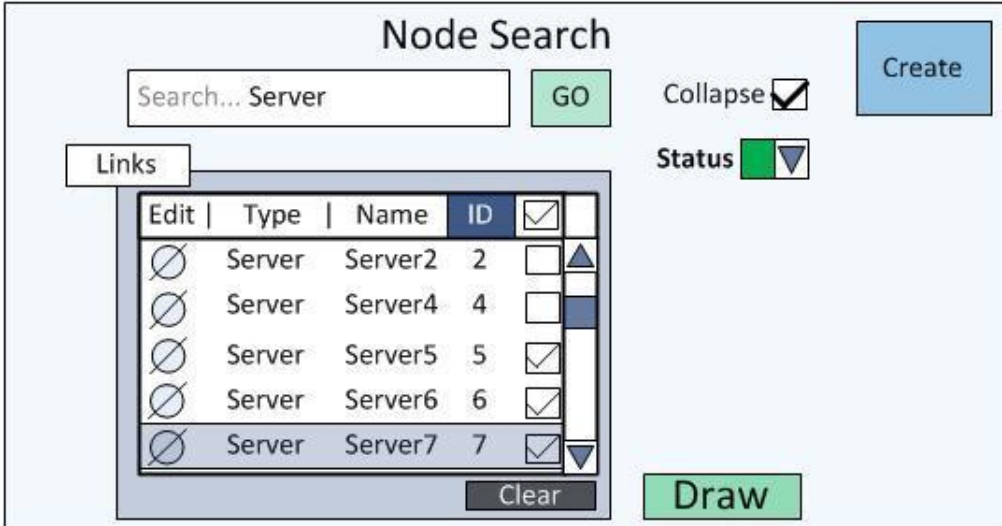
##### Page 1: Login page



The login page features a light blue background with the title "GECD ARCHITECTURE" centered at the top. Below the title, there are two input fields: "Username:" and "Password:". A green "Log In" button is positioned to the right of the password field.

Figure 19: Login Page Design.

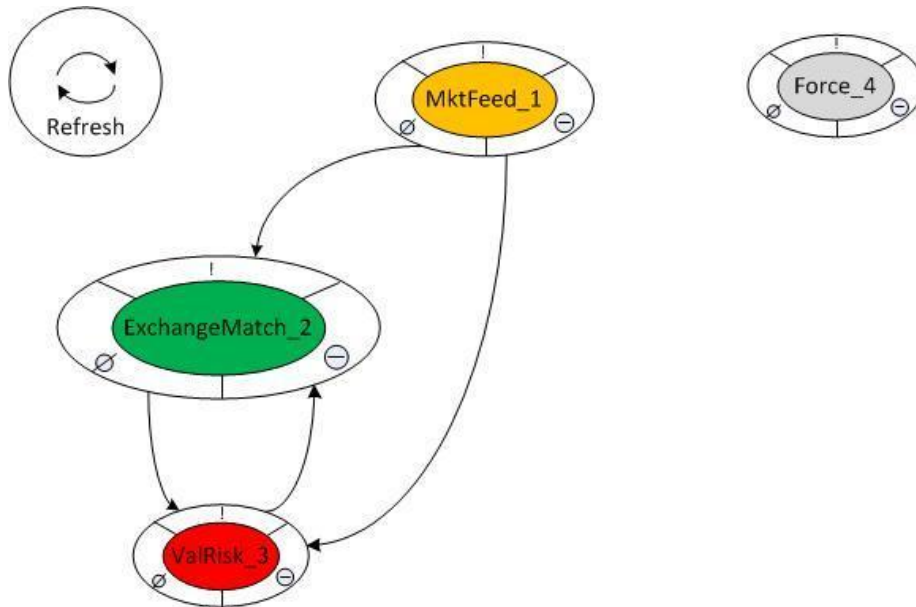
##### Page 2: Main page with the visualization on the bottom



The main page is titled "Node Search" and includes a search bar with the placeholder text "Search... Server" and a green "GO" button. To the right of the search bar are a "Collapse" checkbox (checked) and a "Status" dropdown menu (set to "Green"). A blue "Create" button is located in the top right corner. Below the search bar is a table with the following data:

Edit	Type	Name	ID	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	Server	Server2	2	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Server	Server4	4	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Server	Server5	5	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Server	Server6	6	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Server	Server7	7	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Below the table is a "Clear" button. To the right of the table is a green "Draw" button.



**Figure 20: Main Page Design.**

**Page 3: Node Editor**

**Node Editor**

ID 14 Cancel Delete Apply

Node Name

Node Type  ▼

Links

Edit	ID	Link Type	Name
⊗	4	Server	Server4
⊗	8	Server	Server8
⊗	2	Server	Server2
⊗	3	Team	Supp3

Create

**Figure 21: Node Editing Page Design.**

**Page 4: Capacity Management**

Capacity Management							
Server/App	ID	Name	Metric	Value	Units	Threshold	Timestamp
Application	2	ExchangeMatch	Tx/Sec	3,000	#	2,000	11/15/2011 11:38:29
Application	2	ExchangeMatch	Latency	8	ms	4	11/15/2011 11:38:32
Server	4	Server4	% Cap	95%	%	75%	11/15/2011 11:38:35

**Figure 22: Capacity Management Design.**

Page 1 was the login page. By providing username and password, users could get into the application. Page 2 was the main panel displayed after the user logged in. All the information of databases was listed here in the master interactive report. The page also implemented controllers to interact with the graph. Ideally, the graph was displayed right below the main table. By clicking on the Edit icon of any record, the user was directed to Page 3, a *form* page, where detailed information of that record was ready to edit. For each record of Application or Hardware, there was a corresponding Page 4 showing capacity thresholds and metrics.

### 6.1.2. Database Design

There are six types of enterprise entities in the visualization. Five of them are represented as “node”, Application, Hardware, Organization Unit (OU), People, and Process, and one as “edge”, Link. Link is a source-to-destination two dimensional record, which preserves the relationship of the other five types of entities. Process is a composited type of node entity. It can not only be displayed as a single node to illustrate how different Processes interact with each other, but also as a group containing all the sub entities to show what a particular Process consists of.

How to store these six types of entities in the database was very important. One idea was to have a master table containing all the records of node type entities and another table for Link. It was convenient to extract information for the master interactive report on Page 1 in this way. It was also easier to make references since all the information is stored at one spot. However, different entities might have totally

different columns. It was hard to accommodate all the needs from different entities in one table. Furthermore, data of entities were provided by a variety of departments in the company. For example, data for Hardware was fed by infrastructure teams while information of People came from an organization chart. Organizing different entities into separate tables made it much easier to manage data from various sources.

With the analysis above in mind, the database was designed as follows. Each entity of node type (OU, Hardware, Application, People, and Process) had its individual table (EA\_OU, EA\_HARDWARE, EA\_APPLICATION, EA\_PEOPLE, and EA\_PROCESS).

Link had its own table EA\_LINKS with records that establish a relationship from entity of reference 1 to entity of reference 2. Each Link record had a nullable field called Process\_ID, which indicated whether the relationship belongs to a particular process.

To feed all the data into the master report on Page 1, EA\_APP\_VIEW was made to compile all the records from the six tables and create a temporary “view” of the entire database. It was created solely by a SELECT statement that put all the data together. It was not a real table but could be used the same way as a regular table in the source of the interactive report.

Six tables and one “view” was a very important design decision made in the project. It realized both the generalization in the master report and the customization in each of the edit forms.

## **6.2. Technology Used**

### **6.2.1. Software**

Graphviz – Picture Generator

ApEx – Application builder

SQL Developer –Programming environment for stored procedures and database management

### **6.2.2. Programming Language**

Dot Language – primary language used in input script for Graphviz.

SQL –primary language used to feed data in ApEx components. It was also heavily used in SQL Developer for database management.

PL/SQL – major language used in conditional expressions and process sources in ApEx. It was also critical for functions and stored procedures in SQL Developer.

HTML – secondary language used in input script by Graphviz. It also played an import role in customizing page appearances in ApEx.

Javascript –secondary language used in functions for ApEx.

### 6.2.3. Technical Support

ApEx was locally installed in a server (nycs00057970:8080) of BNP Paribas. It was necessary for dbms\_scheduler to work because the scheduler needed to execute OS commands. It was also more convenient to reference local files.

Databases were also moved to the same server.

Oracle package dbms\_scheduler was used in a stored procedure to trigger the graphing process.

### 6.3. Architecture

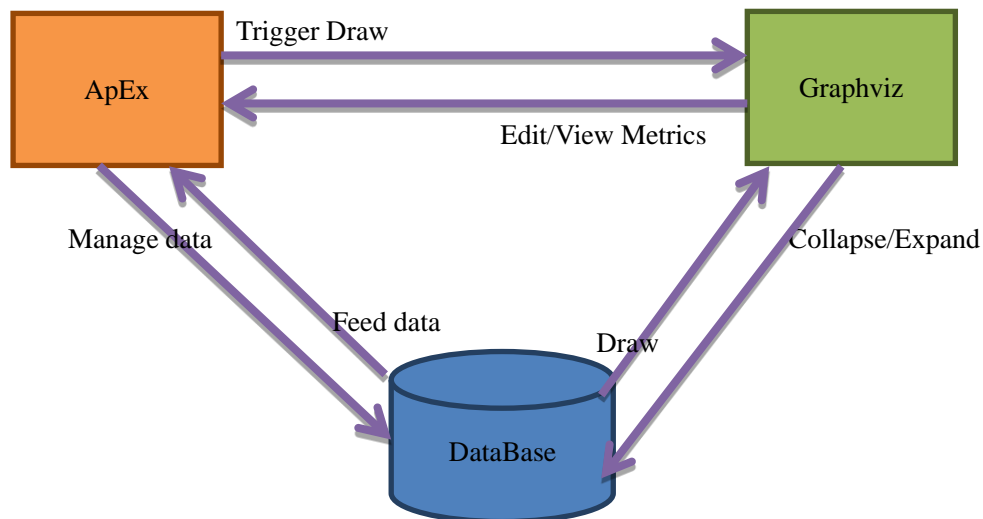


Figure 23: Architecture.

### 6.4. Data

In the formal development of the application, real data was plugged into the database.

## 6.5. Methodology

This is the section that explains the implementation of the most important features in the application.

### 6.5.1. The implementation of user preferences

User preference was a concept that users only saw the part of the data they selected displayed in a way they prefer. Data could be selected altogether in a pre-saved diagram or one by one through the checkboxes discussed in Phase I of the project. The way the data was displayed, either expanded or collapsed, could be specified in a separate checkbox.

Each graph had its own set of preferences and one user could have multiple graphs. Therefore user preference could not be stored together with the master data. Instead, two tables, `EA_USERPREFERENCES` and `EA_USERPREFERENCEDETAILS` were used to keep copies of the master data. Each copy captured the preferences set by a particular user for a specific graph.

The user preferences table kept track of the relationship between each graph and its owner. Each record in the table stored high-level information about a graph with a reference into the table `EA_USER`, where a specific user could be looked up. Detailed information of a graph was kept in the user preference details table. Records with the same reference id which pointed to the same graph made a full copy of the master data with `SELECT/DESELECT` and `EXPAND/COLLAPSE` columns set to user preference.

On the user interface side, the main report was dynamically populated by the data copy from the user preference details table based on what saved diagram the user selected to see. This set of user preferences could then be further tweaked by checkboxes. By checking the corresponding boxes, the `SELECT/DESELECT` or `EXPAND/COLLAPSE` column of a record was set and the user preference was saved back to the database. These two fields were critical for the “CRAWL” procedure to figure out how to generate the graph.

### 6.5.2. Improvement on graph interactivity

Hyperlinks embedded in sections of a node were more than web page URLs.

They could be used to launch a stored procedure located on the server specified by the URL. However, embedding stored procedures the same way as regular web page URLs had the side effect of opening extra windows.

Completely getting rid of the side effect was difficult given the way how URLs work. The compromised solution was to pop up the target window and immediately destroy it. This required several steps. Firstly pop up a second window with the target URL to launch the stored procedure. Secondly destroy the new window. Thirdly go back to the previous page in the main window since the main window has automatically changed to a blank page during the first and second steps. By the third step, it achieved the effect of “staying at the same page”.in the main window.

There were two possible places to implement the changes: the text file taken by Graphviz as input and the svg file outputted by Graphviz. Research showed that Graphviz has limited html options (Node, Edge and Graph Attributes). For example, it lacks the onclick attribute common in html script. This forbade the use of “on-click” event where the new solution could be possibly implemented. Graphviz also has no way to render JavaScript, which meant it was impossible to include multi-step arguments in one single URL.

The new solution was first implemented in JavaScript by directly altering the svg file Graphviz generated. New arguments were added among the xml code. Later, however, it was found that although Graphviz did not understand JavaScript, it treated the code as plain URLs without complaining. When Graphviz rendered the script, the code got passed on into the xml code in svg. This way it saved another round of processing of the graph before it got displayed.

At this point, the outgrowth of graph interactivity - the piled-up extra windows was replaced with a flash of pop-up. This largely improved user experience.

### **6.5.3. Embedment of graph in the application**

Originally, users had to switch between browser pages to interact with both the ApEx application and the graph. Embedment of graph in the application finally realized the notion to build an all-in-one-spot application with all technologies well



integrated.

The graph was first included behind the “src” (source) tag of an *html* region in the application. The graph was displayed without a problem - graph interactivity was fully supported within the page. However, the display did not synchronize well with what was actually stored in the database. Whenever an update was made to the graph, the svg file was regenerated correctly; however, the display was not always up-to-date. This posed a serious problem because users never knew if the current visualization was what they looked for. Many methods were tried including refreshing the page but none worked. It was believed that there was caching going on behind the screen. What was specified at the source was not always re-fetched each time the graph itself was changed.

The solution to this caching problem was to use *PL/SQL Dynamic Content Region* to contain the graph. This is a special region provided in Apex which automatically refreshes itself whenever its source is changed. The PL/SQL procedures to generate the graph and load the image were specified in the source. This way the graph displayed in the page was always the latest version.

The solution brought in striking functionalities to the application surprisingly. One was “live” graph interactivity. Originally by clicking on a node in the graph to indicate expanding or collapsing its children, the graph would not instantly change itself. Users had to go through an intermediate step to redraw the graph in order to see the updated version. The approach of *Dynamic Content Region* together with the hyperlink approach embedded in the graph, however, made the graph live. There was no extra step to refresh the graph. The graph vividly expanded or collapsed within seconds of user clicking.

## 7. Results

At the end of the project, a well-integrated visualization and data management application was built. This section is a full presentation of the final product.

Following are the most important application pages. These pages were implemented closely by the detailed design plan.

- 1) Login page. Users are prompted to this page when the application is launched.

**Figure 24: Login Page Implemented.**

- 2) Main page. The main body is the search region where users can specify user preferences.

Select/Deselect	Expand/Collapse	Id	Object	Name	Type	Links	Selected?	Collapsed?
<input type="checkbox"/>	<input type="checkbox"/>	1	APPLICATION	Edge	Trading	0	1	1
<input type="checkbox"/>	<input type="checkbox"/>	2	APPLICATION	Force 4	Trading	1	1	1
<input type="checkbox"/>	<input type="checkbox"/>	4	APPLICATION	ValRisk 1	Valrisk	0	1	-
<input type="checkbox"/>	<input type="checkbox"/>	121	APPLICATION	CBSX	Exchange	2	1	-
<input type="checkbox"/>	<input type="checkbox"/>	122	APPLICATION	CHX	Exchange	2	1	-
<input type="checkbox"/>	<input type="checkbox"/>	123	APPLICATION	BX	Exchange	2	1	-
<input type="checkbox"/>	<input type="checkbox"/>	124	APPLICATION	AMEX	Exchange	2	1	-
<input type="checkbox"/>	<input type="checkbox"/>	125	APPLICATION	NSDQ	Exchange	2	-	-
<input type="checkbox"/>	<input type="checkbox"/>	126	APPLICATION	NSX	Exchange	2	-	-
<input type="checkbox"/>	<input type="checkbox"/>	127	APPLICATION	PSX	Exchange	2	-	-
<input type="checkbox"/>	<input type="checkbox"/>	128	APPLICATION	CTS UTDF	Exchange	2	-	-
<input type="checkbox"/>	<input type="checkbox"/>	129	APPLICATION	NYSE	Exchange	2	-	-
<input type="checkbox"/>	<input type="checkbox"/>	130	APPLICATION	ARCA	Exchange	2	-	-
<input type="checkbox"/>	<input type="checkbox"/>	131	APPLICATION	BATS Y	Exchange	2	-	-
<input type="checkbox"/>	<input type="checkbox"/>	132	APPLICATION	BATS Z	Exchange	2	1	-

**Figure 25: Main Page Implemented.**

- 3) Visualization. There is a separate tab called *Diagram* that is used to display the graph.

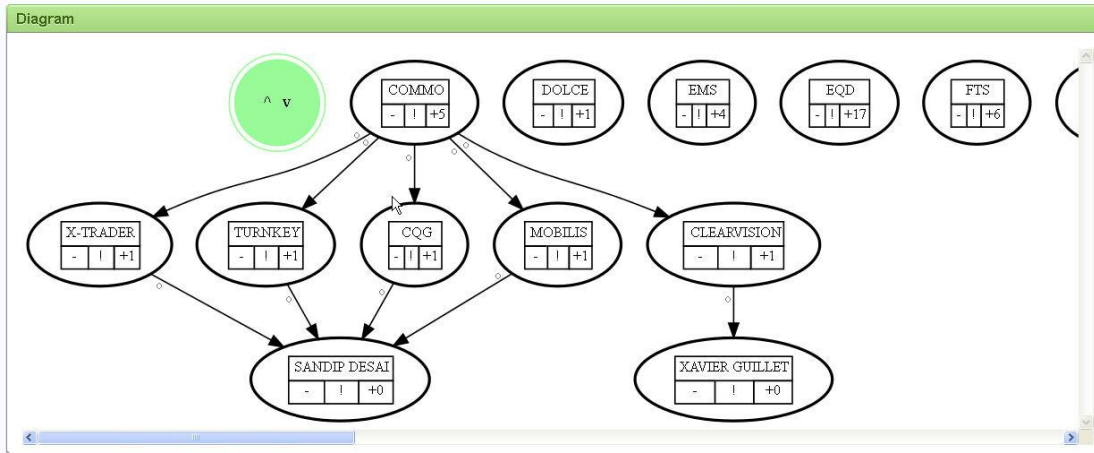


Figure 26: Visualization Page Implemented.

4) Performance Metrics page. It includes both diagrams and charts.

	<u>Id</u>	<u>Refid</u>	<u>Name</u>	<u>Type</u>	<u>Comments</u>	<u>Unit</u>	<u>Critical</u>	<u>Warning</u>
Select	5	2	My app latency	Latency	-	ms	20	7
Select	6	2	mem hog	Memory Usage	-	%	80	50

1 - 2

Report on Metrics				
ID	REFID	TS	VALUE	
25	5	21-NOV-11 05.44.26.978000 PM	11	
26	6	21-NOV-11 05.44.27.072000 PM	21	
27	5	21-NOV-11 05.44.27.165000 PM	2	

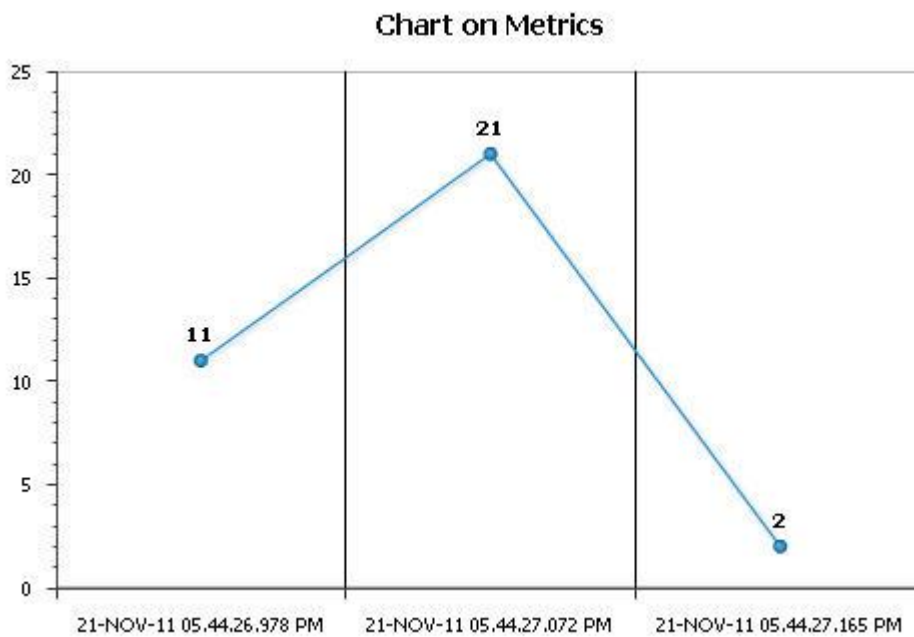


Figure 27: Capacity Page Implemented.

## 7.1. Major functions and capabilities implemented

The below is a description of major functions implemented on the pages just displayed.

### Database Visualization based on User Selection

	Select/Deselect	Expand/Collapse	Id	Object	Name
	<input checked="" type="checkbox"/>	<input type="checkbox"/>	1	APPLICATION	Edge
	<input checked="" type="checkbox"/>	<input type="checkbox"/>	2	APPLICATION	Force 4
	<input checked="" type="checkbox"/>	<input type="checkbox"/>	4	APPLICATION	ValRisk 1
	<input checked="" type="checkbox"/>	<input type="checkbox"/>	121	APPLICATION	CBSX
	<input type="checkbox"/>	<input checked="" type="checkbox"/>	122	APPLICATION	CHX
	<input type="checkbox"/>	<input checked="" type="checkbox"/>	123	APPLICATION	BX
	<input type="checkbox"/>	<input type="checkbox"/>	124	APPLICATION	AMEX

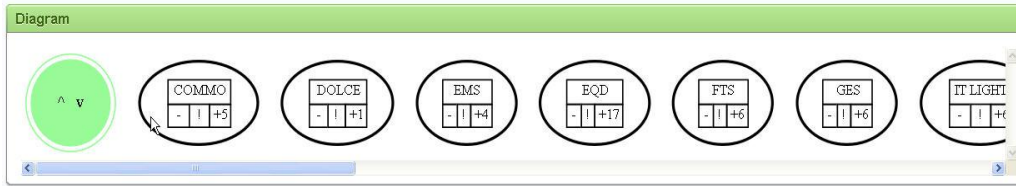
**Figure 28: User Selection Implemented.**

Within the context of a saved diagram, users can select/deselect through checkboxes the entities they want to include in their visualization. They can further indicate if the entity is expanded downward to include child entities or collapsed as a single node. The visualization of the selected entities is then displayed.

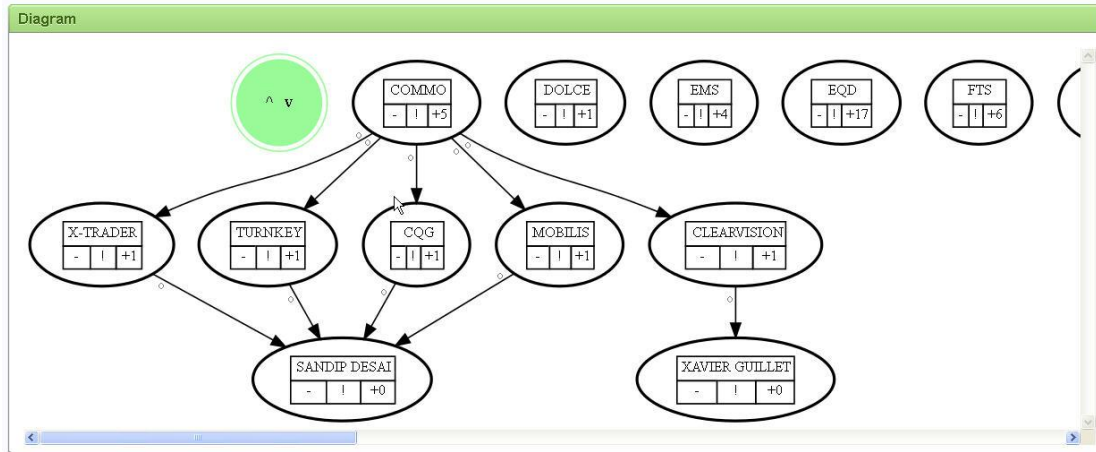
### Graph Interactivity

The entities on the graph are clickable in four sections, the name section, the “+” and “-“, and the metrics section. By clicking on the name section, users are able to edit the entity.

By clicking on the “+” and “-“sections of a node in the graph, users are able to expand and collapse the node in the next generation of the graph. By clicking on the metrics (!) section, users are able to view how an *Application* or *Hardware* entity performed over time against the threshold of an attribute (i.e. latency, memory etc.)



**Figure 29: Diagram collapsed.**



**Figure 30: Diagram expanded.**

## Capacity Metrics

Users can view the capacity metrics of an *Application* or *Hardware* entity either from the editing page of the entity or from the metrics section of the node in the graph. The metrics are a time-based series of performance statistics measured against certain attributes (i.e. latency, memory etc). They are presented both in a table and in a line chart. In the table, the records that pass the thresholds (critical or warning) are color-coded (red or orange) to highlight the problem.

## Summary View vs. Detailed View

Users can choose to see a summary view or a detailed view of their data. A summary view collapses all the physical instances of an entity into one single node. It hides unnecessary details from users in the graph. In contrast, a detailed view rolls down to show every instance of an entity. It is for a thorough examination of the entity if the summary view signals a problem.

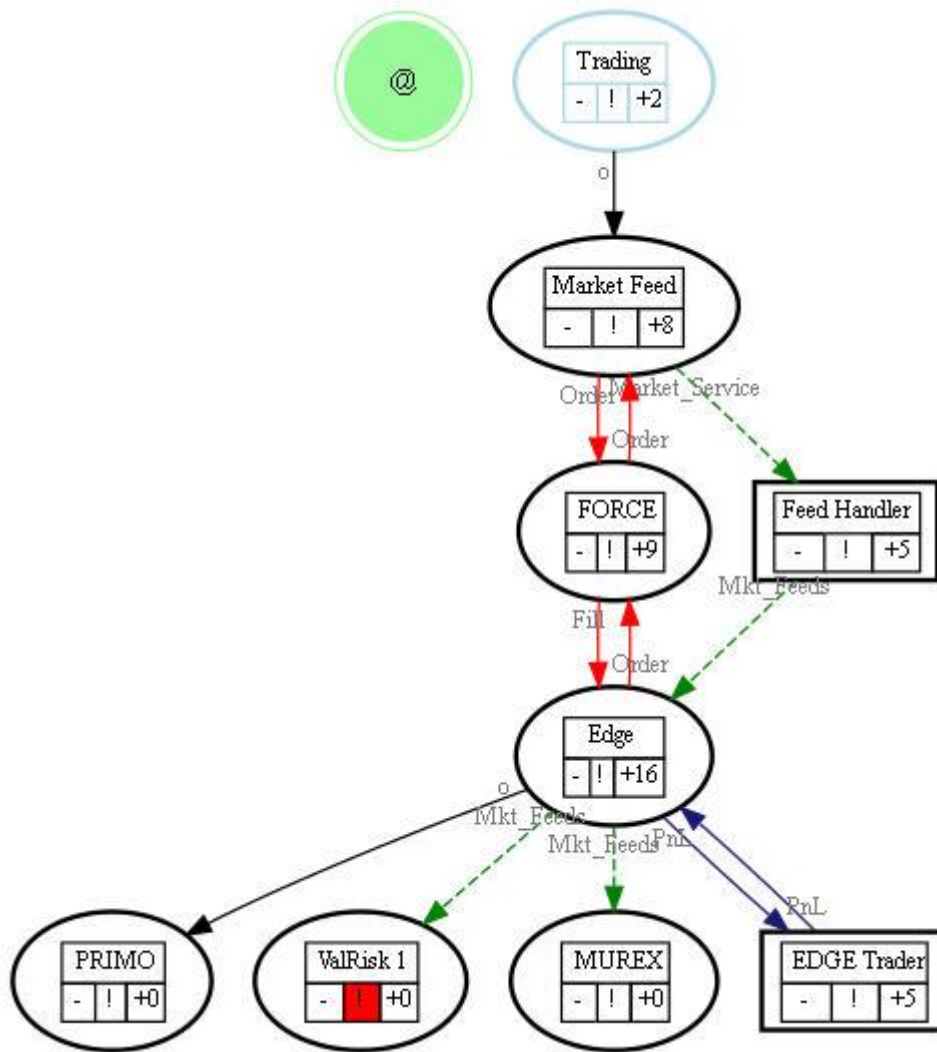
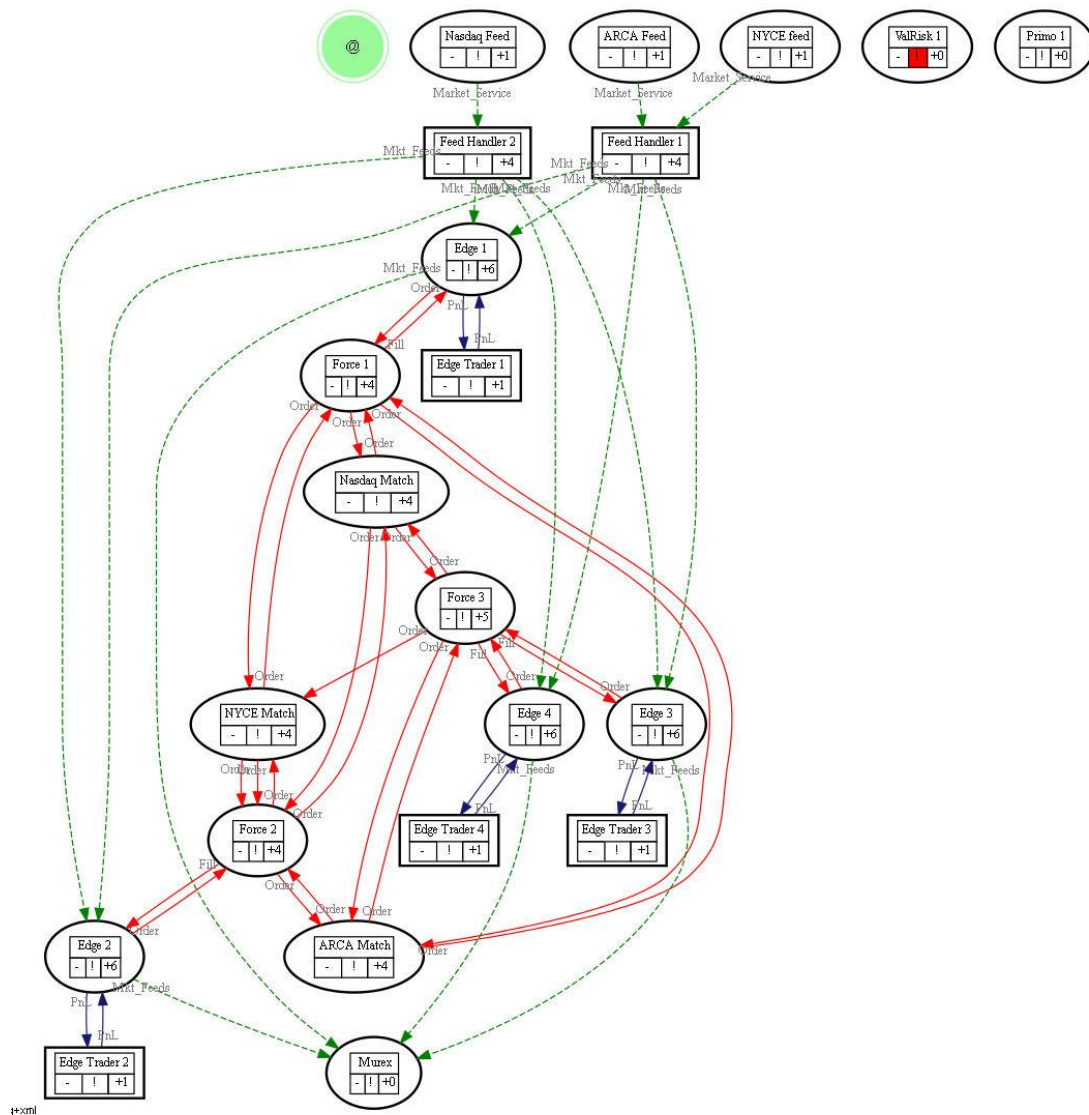


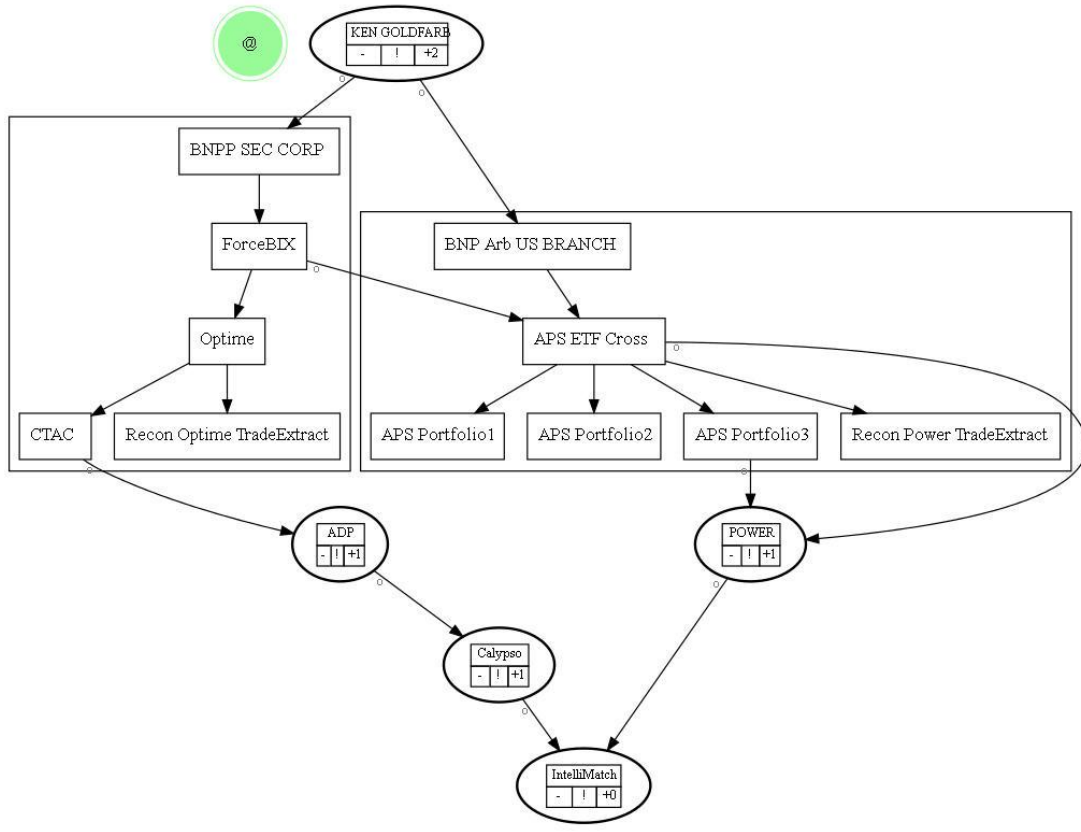
Figure 31: Diagram Rolled up.



**Figure 32: Diagram Rolled down.**

### Processes Sub-graph

Graphviz lends the user the ability to superimpose smaller graphs on top of the general one using the sub-graph functions. In our project, processes are shown in sub-graphs, and are shown with a black rectangular box. Entities outside the process can still connect to links within the sub-graph box. Sub-graphs are used to give the user information about both the operational hierarchy and process diagrams while maintaining graphical clarity.



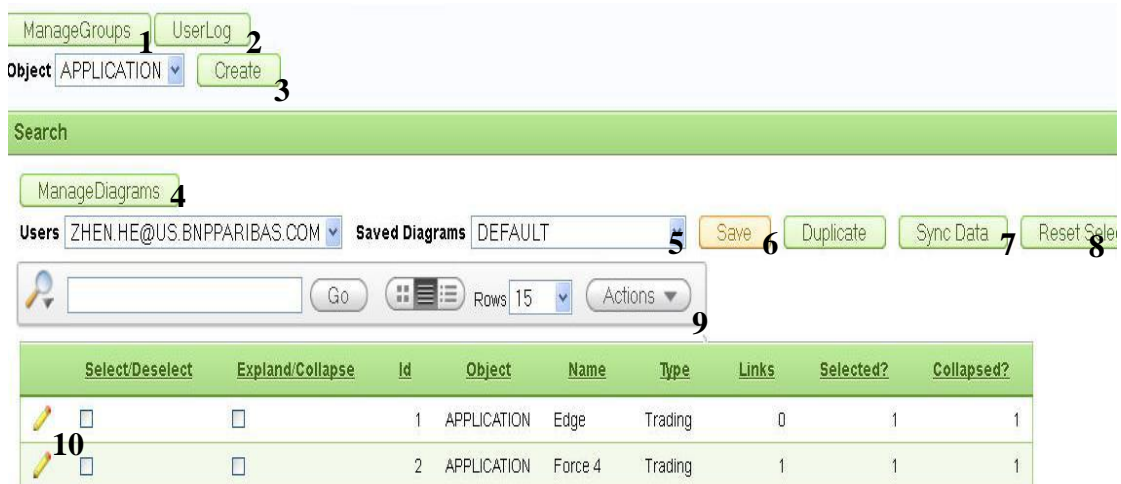
**Figure 33: Diagram with Composite Processes.**

**Database management**

Users are able to view the entire database in one master table, create/edit entities, and create/edit relationships between entities through a data management interface. They can also choose to synchronize their copy of data with master data if needed.

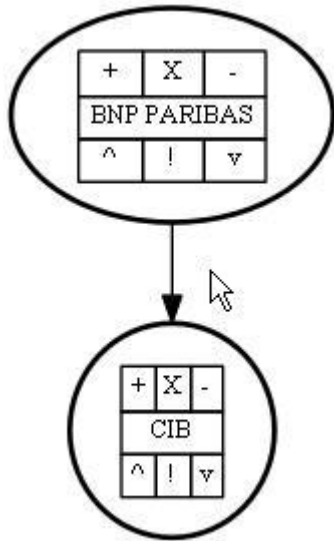


## 7.2. Summary of functions



**Figure 34: Summary of functions in ApEx interface.**

1. Manage the graphical attributes of an entity or a relationship in the graph.  
Users can assign colors, shapes, and styles to a node or link.
2. Keep a log of user actions made to the data. The actions include insert, delete, and update and can be sorted by date.
3. Create an entity of type Application, Hardware, OU, People, or Process.
4. Create a new diagram or edit an existing one. A diagram is a copy of the master database with a specific set of user preferences. It is equivalent to a graph in its database format.
5. Go back to any saved diagram under the current selected user.
6. Save any user preferences modification made to the current diagram.
7. Synchronize the current user copy with the master database. The mismatch between the two comes from create/edit/delete an entity or a relationship. The synchronization has nothing to do with user preferences.
8. Reset user preference.
9. Search for any particular record(s).
10. Edit an existing entity. Relationships that involve that entity can be edited from the editing page for that entity.



**Figure 35: Summary of functions on Super Node\***

+: Expand a node so that children of that node are shown in the graph.

-: Collapse a node so that children of that node are hid.

X: Hide the current node.

[Node Name]: Prompt the user to the editing page for the node.

^: Roll up the graph so it displays a summary view.

v: Roll down the graph so it displays a detailed view.

!: Prompt the user to the performance metrics page for the node. This section is color-coded according to the health of the entity.

\*Note that by the end of the project, Super Node as shown above was not fully supported in the application.

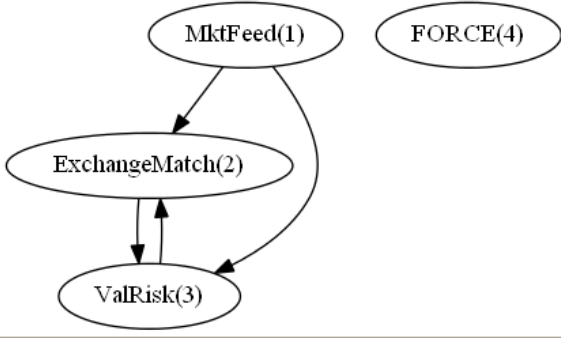
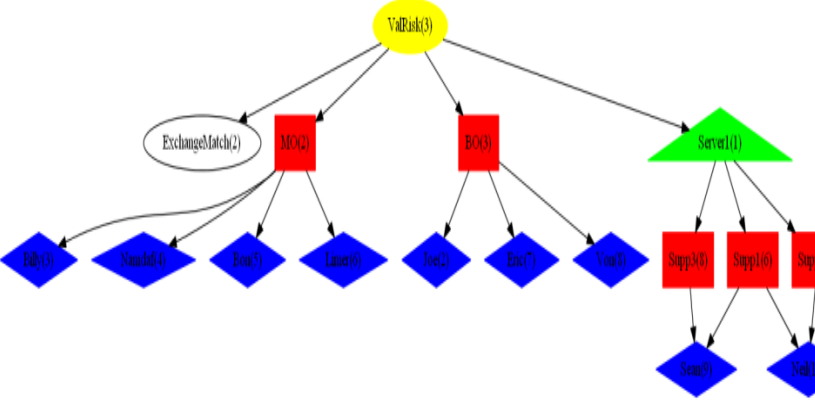
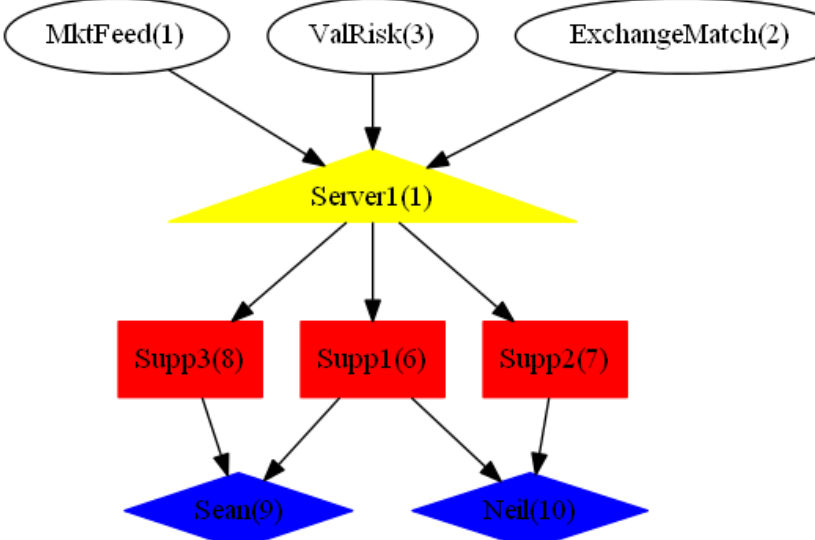
## 8. Conclusion

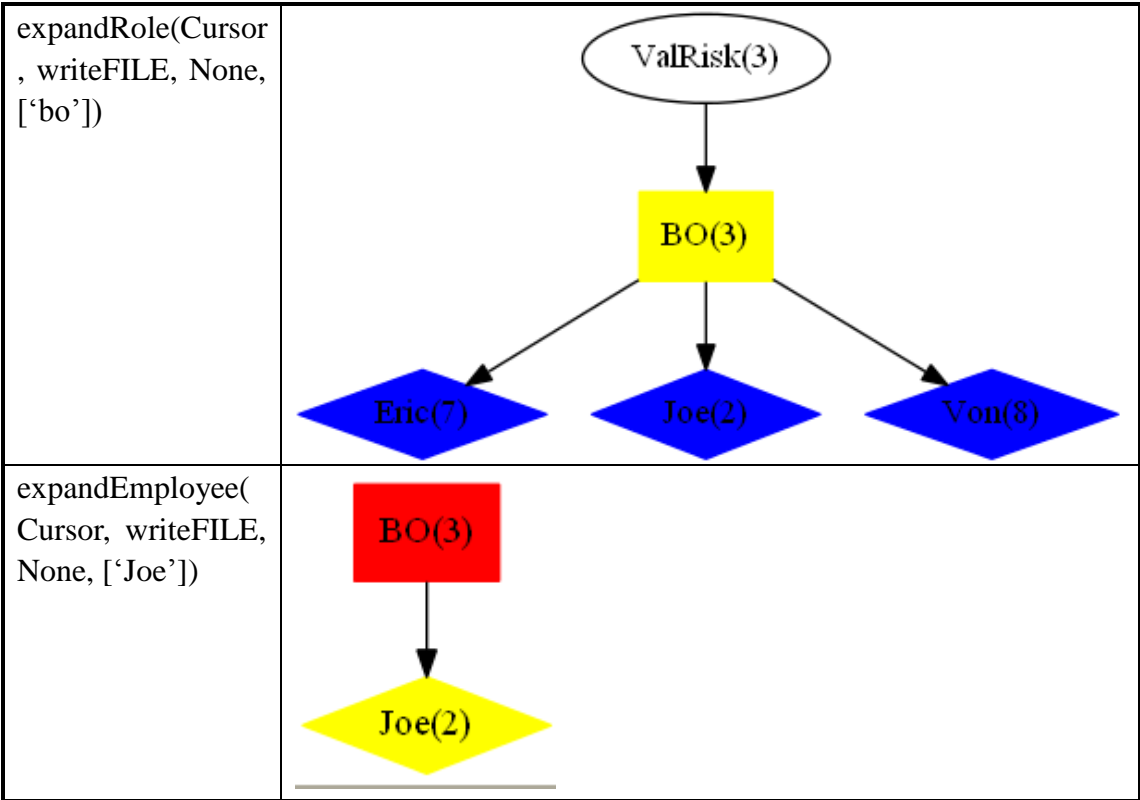
BNP Paribas, an international bank with headquarters in Paris, France, sponsored the Enterprise Architecture Project. The goal of the project was to create a graphical management application that allows the Global Equities and Commodity Derivatives group to better visualize a wide array of system flows. It was required for the application to have bi-directional communication with internal BNP databases and clearly show any possible problems with the data, such as an overloaded server.

The initial stages of the project involved finding the right set applications to complete these objectives. A variety of options were explored at this stage. Eventually it was determined that the combination of Graphviz and ApEx was the tool to use and all remaining resources were allocated to this strategy.

Throughout the project term, the application was transformed from a concept to a working prototype that will serve as a basis for further development. During the formal development portion of the project, many requests for additions to the project from the stakeholders of the project were translated into code and presented to the group. At the conclusion of the project, a team was put in place at BNP to continue development of the tool.

## Appendix A

Input (Python code)	Output (Graphviz graph)
<pre>default(Cursor, writeFILE)</pre>	 <pre> graph TD     MktFeed(1) --&gt; ExchangeMatch(2)     MktFeed(1) --&gt; ValRisk(3)     FORCE(4) --&gt; ValRisk(3)     ExchangeMatch(2) &lt;--&gt; ValRisk(3)     </pre>
<pre>expandApplication (Cursor,writeFILE, None, ['VALRISK'])</pre>	 <pre> graph TD     ValRisk(3) --&gt; ExchangeMatch(2)     ValRisk(3) --&gt; Mkt(3)     ValRisk(3) --&gt; B(3)     ValRisk(3) --&gt; Server1(1)     Mkt(3) --&gt; Billy(5)     Mkt(3) --&gt; Nando(6)     Mkt(3) --&gt; Bob(5)     B(3) --&gt; Jose(4)     B(3) --&gt; Eric(5)     B(3) --&gt; Yoon(6)     Server1(1) --&gt; Supp3(8)     Server1(1) --&gt; Supp1(6)     Server1(1) --&gt; Supp2(7)     Supp3(8) --&gt; Sean(9)     Supp1(6) --&gt; Sean(9)     Supp1(6) --&gt; Neil(10)     Supp2(7) --&gt; Neil(10)     </pre>
<pre>expandServer (Cursor, writeFILE, None, ['SerVER1'])</pre>	 <pre> graph TD     MktFeed(1) --&gt; Server1(1)     ValRisk(3) --&gt; Server1(1)     ExchangeMatch(2) --&gt; Server1(1)     Server1(1) --&gt; Supp3(8)     Server1(1) --&gt; Supp1(6)     Server1(1) --&gt; Supp2(7)     Supp3(8) --&gt; Sean(9)     Supp1(6) --&gt; Sean(9)     Supp1(6) --&gt; Neil(10)     Supp2(7) --&gt; Neil(10)     </pre>



## Appendix B

### Interested Parties for the Data (BNP Paribas. “Enterprise Architecture Overview.”)

	United States	Europe
<b>Organizational charting</b>	<p><b>PeopleSoft</b> : Ted Stephens/Michael Bastock HR Would like to have a better graphical source to maintain. This has to be up-to-date for end-of-year</p> <p><b>BEN</b>: Manoj Tejwani/Tim Mahler This system has simple org chart tools.</p>	<p><b>OREF</b>: Brice Degromard/Tarik Slassi/Reda Hadjouti</p> <p>Organizational charting system</p>
<b>System Graphing</b>	<p>John Crocker Core Integration Service Mngr <b>BAMM Plus</b>: IT manually inputs systems data.</p>	<p><b>ServiceNow</b> : Martin Saldarriaga/Reda Hadjouti Tool for managing the production activities (incidents, problems, changes, etc.) aligned with ITIL. ServiceNow contains a module to store the configuration items and their relationships. A configuration item can be an application, a server, a network card, etc</p>
<b>Hardware Mapping</b>	<p>John Crocker Core Integration Service Mngr <b>CA Asset Manager</b> : This takes an inventory at 1pm of all servers, how many cpu's, disk space, primary applications run <b>CA Network System Monitoring</b> – This measures cpu utilization, provides detailed performance reports</p>	
<b>System Entitlements</b>	<p><b>MyIT</b> Stuart Lincoln</p>	<p><b>STAF</b>: Brice Degromard/Tarik Slassi/Reda Hadjouti Staff Assignment Follow-Up</p>
<b>Analytics</b>	<p>Gerald Sebastien – Planning for Infrastructure Dashboard Scott Visconti/ Christophe Poulmarc'k -Planning for system dashboard</p>	<p>Patrice Piron – GECD Functional Architect</p>

## Appendix C

### ComputerSystem

SystemID	System	Description	Process	Sys	GroupTo	GroupLabel
1	NYCE feed	NY Futures Exchange Feed	null	9	1	Mkt Feed
2	ARCA Feed	Arca NYSE Exchange Feed	null	9	1	
3	Nasdaq Feed	Nasdaq Exchange Feed	null	9	1	
4	NYCE Match	NY Futures Exchange Market	null	9	4	Exchange Match
5	ARCA Match	Arca NYSE Exchange Market	null	9	4	
6	Nasdaq Match	Nasdaq Exchange Market	null	9	4	
7	Feed Handler 1	NYSE & ARCA Feed Handler	null	10	7	Feed Handler
8	Feed Handler 2	NASDAQ Feed Handler	null	10	7	
9	Edge 1	Dir Trdng 1		6	1	Edge
10	Edge 2	Dir Trdng 2	null	1	9	
11	Edge 3	Dir Trdng 3	null	1	9	
12	Edge 4	Dir Trdng 4	null	1	9	
13	Edge Trader 1	Dir Wrkstn 1	null	1	14	Tdr Wrkstn
14	Edge Trader 2	Dir Wrkstn 2	null	1	14	
15	Edge Trader 3	Dir Wrkstn 3	null	1	14	
16	Edge Trader 4	Dir Wrkstn 4	null	1	14	
17	Force 1	Force Dir 1	null	1	17	Force
18	Force 2	Force Dir 2	null	1	17	
19	Force 3	Force Dir 3 & 4	null	1	17	
20	Murex	Back Office	null	3		
21	ValRisk	Risk	null	5		
22	Primo	P&L	null	6		

### Infrastructure

InfraID	Name	Type
1	AppServ1	App Server
2	AppServ2	App Server
3	AppServ3	App Server
4	DB Serv1	DB Server
5	DB Serv2	DB Server

### ProcessID

Process ID	ProcessName
1	Equity Swaps
2	Forward Trading
3	Security Trading
4	Stock Borrow & Loan
5	OTC OPTIONS
6	Prop AQS

### Relation Classification

Relation ID	Relation Name	Line Link
1	Trade	
2	Position	
3	Referential	
4	Market	
5	Risk	
6	P&L	
7	Order	

### Map Infrastructure to System

App	Infra
17	1
17	4
18	2
18	4
19	3
19	5

(BNP Paribas. “Enterprise Architecture First Phase – a tool for capacity management.”)



## **Appendix D – Report of C-Term 2012**

### *Part I: Introduction to Structured Finance*

#### **Definition**

Structured Finance is a broad term. A summarization of some commonly used concepts is that structured finance is used to describe a sector of finance that was created to help transfer risk using complex legal and corporate entities. Typically the process of risk transferring involves activities that will not heavily affect the balance sheet. The term “structured finance” implies that the corporations or entities are using special purpose entities or special purpose vehicles (SPE or SPV) to exchange future cash flow from an existing asset or portfolio while financing the corporation or entity leveraged by the asset. This risk transfer as applied to securitization of various financial assets (e.g. mortgages, credit card receivables, auto loans, etc.) has helped to open up new sources of financing to consumers. However, the asset allocation from certain assets of corporation to cash flows in the balance sheet is risky, as the increasing cash flow comes from the condition that the asset-liability ratio is unchanged. Structured finance also includes the innovation of new financial instruments which allows for re-transfer funds to investors (as asset-backed securities)

#### **Classification**

Usually, structured finance is divided into two categories: Asset Financing and Capital Financing. Asset Financing is further divided into Current Asset and Fixed Asset.

Examples of Asset Financing:

Current Asset classes: cash financing (a loan-deposit); accounts receivable financing (factoring, payments); inventory financing (warehouse financing); order financing (credit packing, red clause letter of credit), etc.

Fixed Assets classes: mortgage, hire purchase, and finance lease of fixed assets.

The main types of Capital Financing are stock and equity financing; swaps; leveraged buyouts

Terminology used in this section:

Factoring: A business sells its accounts receivable to a third party at a discount rate in order to hasten the finance process.

Warehouse financing: A form of inventory financing. Loans are usually made to manufacturers and processors on the basis of goods or commodities held in trust as collateral for the loans.

Credit packing: A loan given to the beneficiary by the bank to enable the individual to purchase raw materials. The beneficiary is usually requested to deposit the DC with the bank as security.

Red clause letter of credit: A specific type of letter of credit which enables a buyer to extend an unsecured loan to a seller. Red Clause Letters of Credit permit documentary credit beneficiaries to receive funds for any merchandise outlined in the letter of credit.

Hire purchase: A persons usually agree to pay for goods in parts or a percentage at a time at an amount of interest.

Leverage buyout: A type of acquisition that acquires a controlling interest in a company's equity with a small amount of cash flow, but a significant percentage of the purchase price is financed through leverage (borrowing)

## **Specification**

Companies can finance themselves in a variety of ways based on different financing structures, but mainly by traditional finance or structured finance. Traditional finance is based on assets, while structured finance is based on credit: it fulfills the purpose of financing by constructing a rigorous transaction model. Companies can use integrated finance methods which may involve both traditional finance and structured finance. The best capital structure allows the limited assets to generate maximum value of present cash flow.

Traditional ways of financing are usually achieved by increasing corporate debt (debt financing) and increasing equity (equity financing) two ways. Debt financing and equity financing reflect the activities of the both left-hand-side and right-hand-side of the balance sheet. Structured finance, unlike the other two, mainly involves with relatively small activities on the balance sheet. Traditional finance involved mainly with fixed assets, such as a house or a newly started company, while structured finance mainly involves with financial assets, such as securities or derivatives.

In order to better illustrate the differences between traditional finance and structured finance, here is a scenario of a small bank: this bank grant loans to multiple individuals, and then the bank uses these loans to construct CDO products and sell these CDOs to investors. In the process, the bank realizes it has problems with its cash flow, so the executives of the bank decide to borrow money from another institution. The borrowing and loans here are ways of traditional finance, and CDOs are examples of structured finance. The observation towards changes in the balance sheet is provided:

The balance sheet of the bank at the beginning would be like this. They have large amounts of accounts receivable, meaning that the bank has released many mortgage loans to its clients:

Balance Sheet						
						22-Feb-12
Assets						
	Cash					\$5,000,000.00
	Accounts Receivable (AR)					\$20,000,000.00
	Property, Plant and Equipment (PP&E)					\$5,000,000.00
	Total Assets					\$30,000,000.00
Liabilities						
	Accounts Payable (AP)			\$5,000,000.00		
	Debt			\$0.00		
	Total Liabilities			\$5,000,000.00		
Shareholders' Equity						
	Common Stock and Additional Paid-In Capital (APIC)			\$10,000,000.00		
	Retained Earnings			\$15,000,000.00		
	Total Shareholders' Equity			\$25,000,000.00		
	Total liabilities and shareholders' Equity			\$30,000,000.00		

**Figure 1: The initial balance sheet**

All of a sudden, the bank has a new business opportunity, but the deal requires a 20 million dollar investment. Unfortunately, due to its poor liquidity, the bank doesn't have \$20m cash on hand. In order to secure this profitable deal, the bank turns to a mutual fund MFLA and borrows \$20m from the MFLA. Therefore the bank has more cash now, as well as accounts payable (for the easiness of the balance sheet, the interest is set to 0 here):

Balance Sheet						
						23-Feb-12
Assets						
	Cash					\$25,000,000.00
	Accounts Receivable (AR)					\$20,000,000.00
	Property, Plant and Equipment (PP&E)					\$5,000,000.00
	Total Assets					\$50,000,000.00
Liabilities						
	Accounts Payable (AP)			\$25,000,000.00		
	Debt			\$0.00		
	Total Liabilities			\$25,000,000.00		
Shareholders' Equity						
	Common Stock and Additional Paid-In Capital (APIC)			\$10,000,000.00		
	Retained Earnings			\$15,000,000.00		
	Total Shareholders' Equity			\$25,000,000.00		
	Total liabilities and shareholders' Equity			\$50,000,000.00		

**Figure 2: Balance sheet after the bank received money from the mutual fund**

Everything is going smoothly, and the bank is waiting the mortgage payment

from those individuals. However, executives at the bank heard bad news about their clients. If their clients are about to default, the bank will experience heavy losses and may even go bankrupt. The balance sheet for the bank after the default would be like the following (they might experience heavy losses in their retained earnings and common stock, but that's not the topic we are concerned with here):

<b>Balance Sheet</b>									23-Feb-12
<b>Assets</b>									
	Cash								\$25,000,000.00
	Accounts Receivable (AR)								\$0.00
	Property, Plant and Equipment (PP&E)								\$5,000,000.00
	<b>Total Assets</b>								<b>\$30,000,000.00</b>
<b>Liabilities</b>									
	Accounts Payable (AP)							\$25,000,000.00	
	Debt							\$0.00	
	<b>Total Liabilities</b>							<b>\$25,000,000.00</b>	
<b>Shareholders' Equity</b>									
	Common Stock and Additional Paid-In Capital (APIC)							\$10,000,000.00	
	Retained Earnings							(\$5,000,000.00)	
	<b>Total Shareholders' Equity</b>							<b>\$5,000,000.00</b>	
	<b>Total liabilities and shareholders' Equity</b>							<b>\$30,000,000.00</b>	

**Figure 3: If those clients default, the bank will have its account receivable sets to \$0**

A \$20m loss is unaffordable for a small bank. The executives of the bank won't allow the defaults to happen. But what can they do to reduce its loss?

Here is where the CDO becomes crucial. The executives of the bank are very smart, and they have thought about the probability of default in the past. In early days, they have created a CDO product and put it into the market. The CDO has following rules: if the clients didn't default, whoever has the CDO would enjoy the free cash; but if the clients default, then the holder of CDO would pay back the loss of the bank.

Investment bank IBLB has shown great interest in the very beginning. So the small bank had a deal with IBLB, with the price of CDO was finalized at \$1m. Now it is time for IBLB to pay the losses of the bank, which is \$20m. Clearly, without the help of CDO, the bank could be bankrupted because of the default of its clients.

The last balance sheet would be look like this:

Balance Sheet						
						23-Feb-12
Assets						
	Cash					\$24,000,000.00
	Accounts Receivable (AR)					\$20,000,000.00
	Property, Plant and Equipment (PP&E)					\$5,000,000.00
	Total Assets					\$49,000,000.00
Liabilities						
	Accounts Payable (AP)			\$25,000,000.00		
	Debt			\$0.00		
	Total Liabilities			\$25,000,000.00		
Shareholders' Equity						
	Common Stock and Additional Paid-In Capital (APIC)			\$10,000,000.00		
	Retained Earnings			\$14,000,000.00		
	Total Shareholders' Equity			\$24,000,000.00		
	Total liabilities and shareholders' Equity			\$49,000,000.00		

**Figure 4: Last balance sheet**

Compare figure 1 and 2, we can see that the balance sheet has changed a lot during a traditional finance process. However, compare figure 2 with figure 4, we could see that balance sheet only changed a little. Here is one of the most important advantages of structured finance: it allows the corporate to use their limited assets to generate great revenue.

Advantages of structured finance are list below:

1. Provide clients long term future cash flows
2. Improve the client's asset turnover ratio
3. Reduce client's asset liability ratio (assuming the structured finance product will not 100% default)
4. Credit enhancement, lower financing costs, diversifies investment products for investors.
5. Self-liquidating, one of the most important specifications of structured finance.

Unlike other finance methods, structured finance products typically do not require additional mortgages or warranties (or requires a little), allows corporate to use their limited mortgage assets into other finance activities.

## Process

Typically, a structured finance product will go through following process:

1. Divestiture of current assets, identify the suitable vehicle for issuing the bond/security, establish the asset pool.
2. Finish the writing for required documents for offering, complete every preparation work for issuing.
3. Obtain the approvals from the central bank and other regulatory authorities to issue bonds and/or securities.
4. Debt/equity/mortgage pay back;
5. Registration for claims and liabilities, disclosure of obligatory information according to the agreement.

Parts from the above, specific tasks might require different processes and operations. For example, partial credit guarantees usually involves with the lender utilizes their credit rating to help clients expanding their financing channels, while securitization includes the collection and the actual sale of certain financial assets, then issue securities which aims to generate cash flows used to repay these assets.

No matter what product is constructing, the credit supply process is always the key factor in the whole process. Any potential instability of credit supply process would jeopardize the whole product severely.

## Environmental Requirements

The United States has a good tradition of growing structured finance market and highest revenue among the world. If we looking back the U.S. structured finance market in terms of history, we can conclude that there are eight elements have played an important role to maintain the growth potential of both the market and revenue. These factors often complement each other, and development of one element could promote the development of other elements, therefore improve the overall quality of U.S. structured finance system.

## **Well-established regulatory system**

There should be a well-established legal framework to protect investors and their legitimate underlying assets for each securitization transaction. Most importantly, when the sponsor / seller in the situation bankruptcy, the law/regulation must protect investors' right to recover the asset and cash flows in the securities of the bankrupted entities. Therefore, speaking of securitization transactions, a special purpose entity which will not bankrupt and have good credits must be established. In addition, the regulation and laws should have clear items about the responsibilities and obligations among issuers, trustees, credit managers and people from service side.

## **Accurate analysis of cash flows**

From the perspective of financing, securitization is essentially a process to help the credit side to raise an amount of capital equals to present value of current assets in future cash flows after deducting the cost of securitization. In the beginning of the securitization, the issuer must conduct a careful cash flow analysis to determine whether the special purpose entity could fully repay the debt on time.

To conduct an analysis about pricing of future cash flow requires a lot of prerequisite assumptions. In addition to using assumed discount rate to calculate present value and pricing of future assets, a well-rounded cash flow analysis must have their own assumed indexes, such as the general economic conditions within the loan period, borrower's ability to repay loans, and the probability of default. Only by researching a large amount of historical data can the researcher be able to come up with reliable assumptions for analysis of cash flows.

## **Clear and reliable accounting activities**

During the process of asset securitization, there might be three relatively important accounting issues: firstly, the accounting activities towards the interest income of securities. As the securities issued by special purpose entities to which the loans have interest, it should be taxed on the special purpose entity accordingly. In order to avoid double taxation, as long as the special purpose entity meet the requirement of being a "grantor trust ", it will be deemed "grantor trust" and will be



granted federal tax exemption in the United States; secondly, to protect investors by clarifying the periodic cash flow between the loans (including interest and principal) of the accounting records.; thirdly, all cash flows generated by transactions should be subject to the strict inspection from professional accountants.

### **Accredited public rating organizations**

As the securities issued from lender are supported by its sponsored loans or commercial loans, stock investors will be more concerned about credit risks of loans and corresponding securities. For example, the housing mortgage-backed securities are credited by the Government National Mortgage Association (Finnie Mae), Federal National Mortgage Association (Fannie Mae) and Federal Home Loan Mortgage Corporation (Freddie Mac). If any housing mortgage-backed securities, commercial mortgage-backed securities and asset-backed securities didn't receive credit from any of these three institutions, then either the security will try a variety of ways to improve its credit rating, or the investors will be in big trouble.

In general, there are four meaningful ways to improve one's credit rating: first/secondary structure of cash flow (or over-collateralization), parent company guarantee, Guarantee Bond (Performance Bond), and letters of credit. Every security wants to get triple-A rating from big 3 credit rating agencies, which are Standard & Poor, Fitch and Moody's.

### **Comprehensive Investment Banking Service**

IPO's and sales of new securities underwriting are the responsibilities of investment bankers. Investment banks acts like a bridge between issuers and investors. However, during the securitization process, the investment banks are essential to the success of a securitization. Investment Bank is responsible for coordinating and helping issuers to deal with legal, accounting, tax and cash flow analysis issues. In addition, investment banks also play the role of dealers: securities pricing, purchasing all issued securities and sold them to individual and corporate investors. After the IPO, investment banks are served as the "market maker" in the secondary market: they actively trade securities in order to ensure the liquidity.

## **Healthy Treasury Bills and Notes market**

In order to ensure the healthy development of securities markets, a healthy and stable bonds and treasury bills market is necessary. Because the government bonds are almost risk-free (exception includes bond governments in turmoil such as Greek Government and Italian Government), so the trading of such bonds on the market will be the beacons for the risk-free return rates in different terms. In terms of risk-free, the curve that describes the relationship between terms and yield was called the 'U.S. Treasury Yield Curve'. This yield curve provides a benchmark for the pricing of fixed-income securities with different risk and different terms in the primary market and secondary market.

## **Active secondary market**

A successful primary market securities must be supported by an active secondary market. An active and healthy secondary market should provide useful information for the upcoming IPOs and pricing information. Once after the issue of new securities, investors need an active secondary market to trade securities and securities need these active markets for its own liquidity. By doing so, investors will be able to trade in a relatively stable market.

However, it should be noted is that investors are the key to a very active secondary market. If investors in a market simply hold securities rather than engage in any transaction, it will be very hard for those investment banks to keep the active of the market. To rely on investment banks to ensure the liquidity in the secondary market is very difficult.

## **Diversified investors**

An important factor that contributes the success in the current structured finance market is the rapid growth of investor groups. Investor group includes various types of investors, from short-term money market investors, to the portfolio managers of commercial banks, or the long-term pension fund managers.

In addition, with the increasing globalization of capital market development, foreign investors will be more and more important. Also, the success in the current structured finance market can also be attributed to the innovative development of

structured finance securities themselves. With products developed in the different kinds, different credits levels and different terms, structured finance securities are able to meet the demand from all investors. High-yield securities can attract "income-oriented" investors, securities with various period of maturity can attract the "term-oriented" investors, and securities with different levels of credit rating can attract those "credit-oriented" investors.

## **Valuation and Assessment**

This section will explain on a macro level of how to evaluate a structured finance product. There will be a section later to detailed explain the valuation of specific products.

Typically, the target being evaluated is an amount of assets of mortgage securities. As discussed above, the operation of a structured finance product usually doesn't involve the with the management activities of original asset holder, therefore a key element to value a structured finance product is to separate the credit/mortgage with original assets.

Besides, there are certain factors that may affect the pricing model of a structured finance product. First of all, the quality of the collateral adequacy matters. Usually the better of quality of collateral, the better of asset turnover ratio is, the better of previous payment history is, and the better the product is. In most cases, the quality of product contributes most in the pricing of structured finance. In addition, the structure of involved transaction's cash flow determines. Investors need their money before the deadline, so the better cash flow structure the transaction associates, the more likely the investor would like to pay for a higher price. Finally, regulation system is a crucial part of valuing a structured finance product, because of those the legal and tax considerations. The legitimacy of collateral representatives', the legitimacy of the cash flow, impacts of cash flow from tax perspectives, everything above will affect the pricing and assessment of a structured finance product. Besides, the political economy will affect the valuation of the product. Generally, in a political

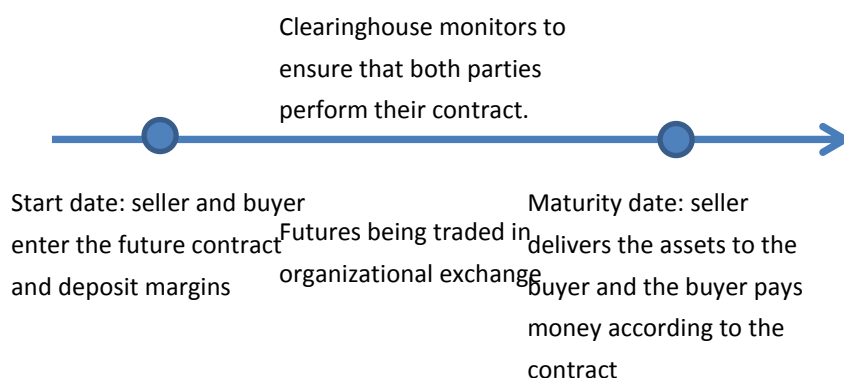
stable nation, the price of a structured finance product would be higher than those of political unstable countries.

## *Part II: Some existing popular structured finance products*

### **Futures Contracts**

Future contract is by definition a standardized contract between two parties to exchange a specified asset of some quantity and quality for a priced agreed today. The assets are going to be delivered at a specified future date. This contract itself theoretically costs nothing to enter. The buyer expects that the price to increase which means in the future they can receive the asset at cheaper price than the future market price. However the seller wants the price to decrease. If so, in the future the seller will be able to sell their asset above the future normal asset price. Because the future contracts are standardized, they are traded in exchanges. The future contracts are introduced by Japan in 1930s and became very popular later on. The underlying asset used to be traditional commodities. Now the assets expand to currencies, securities, financial instruments and intangible assets.

Here is a timeline to better demonstrate future contracts.



**Figure 5: Timeline for future contracts**

In the current finance world, the future contracts are carried out in the following way. Even though we mentioned before that theoretically the future contract costs nothing to enter, in real world, in order to minimize the counterparty risk, both

buyer and seller need to deposit some money, called margin, which is normally proportional to the asset that they are contracted on. Since future contract is marked to market daily. The daily profits and losses will be shown in the traders' account operated by a clearinghouse. At the settlement day, the futures will be settled by either commodities or cashes.

#### *Pricing Model*

$$F(t) = S(t)e^{r(T-t)}$$

Here T means maturity and r is risk-free return. F(t) is the expected future price of the asset and S(t) is the current price of the asset. This formula assumes continuous compounding and the future asset price equals to the current asset price with continuous compound interest.

#### *Risk Exposure*

In addition to the fluctuation of the market price of the assets, both parties in the future contract are subject to counterparty risk, which is the risk that the other party doesn't deliver the goods or doesn't pay the money according to the contract. And a clearinghouse is the one who guarantees both parties performance.

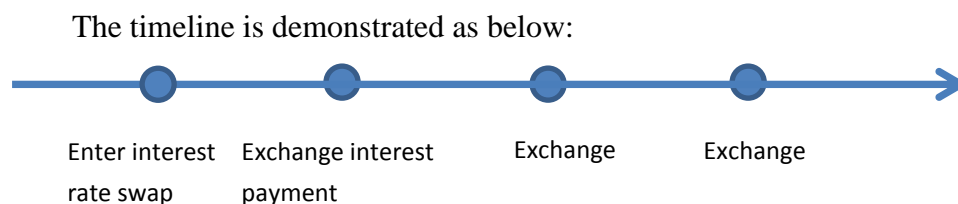
#### *Some Commonly Traded Future Contract*

Eurodollar CD future's underlying instrument is 3-month (90 days) Eurodollar CD. It is currently traded on Chicago Mercantile Exchange and London International Financial Futures Exchange. It has \$1 million face value with cash settlement contract. This means at the maturity date, the Eurodollar CD future is settled in cash for the value of a Eurodollar CD based on London Interbank Offered Rate - LIBOR. Many people use Eurodollar CD futures for hedging (Fabozzi, 24).

## **Interest Rate Swaps**

Interest rate swap is a contract for two parties who agree to exchange interest payments during certain future period. Normally one party agrees to pay a fixed interest payment periodically. They are called fix-rate payers and while the other party will pay at a floating rate according to some reference rate (the most commonly used

reference rate is London Interbank Offered Rate – LIBOR). They are called fixed-rate receivers. The interest rate swap is an over-the-counter instrument, and thus has a lot of varieties. It can be in different currencies and except for the previously mentioned most common kind of fixed-for-floating-rate swap. There is also floating-for-floating-rate swap. The interest rate swap is majorly used by companies who desire to change its financing structure from fixed rate to floating or from floating to fixed-rate in order to reduce funding costs. Interest rate swaps are a very popular instrument and it is now the largest component of global over-the-counter derivative market.



**Figure 6: Timeline for interest rate swaps**

When interpreting the interest rate swap, you can view it as a package of forward contracts with each payment as one forward contract between the fixed-rate receiver and fixed-rate buyer. Interest rate swap can also be viewed as a package of cash market instruments. For example, interest rates swap which exchanges LIBOR rate with 10% fixed interest rate paid annually for 5 years. An investor as a fixed rate receiver entering this interest rate swap equals to buy a 5-year fixed rate bond and financing this purchase by borrowing the notional amount of money for 5 years with LIBOR rate interest paid every year.

#### *Pricing Model*

The value of the fixed leg is:

$PV_{fixed} = C * \text{present value of the sum of the future payments (C is the swap rate)}$

$$PV_{fixed} = C \times \sum_{i=1}^M \left( P \times \frac{t_i}{T_i} \times df_i \right)$$

Here P is the notional amount,  $t_i$  is the number of days in period i,  $T_i$  is the basis according to the day count convention (It may possibly be 365 or be 360 for calculation convenience) and  $df_i$  is the discount.

For the present value of floating interest rate, since we don't know what the future interest rate will be. Thus we predict the future interest rate from forward rates which are derived from the yield curve. And the value of the floating leg will be:

$$PV_{float} = \sum_{j=1}^N (P \times f_j \times \frac{t_j}{T_j} \times df_j)$$

In this formula, N is the number of payments, F<sub>j</sub> is the forward rate, P is the notional amount, t<sub>j</sub> is the number of days in period j, T<sub>j</sub> is the basis according to day count convention and df<sub>j</sub> is the discount factor. The present value of both fixed and floating legs are essentially the sum of the present value of future payments. And when the PV<sub>fixed</sub> and PV<sub>float</sub> equal to each other, there will be no upfront payment from any party.

#### *Risk exposure*

There are two kinds of risks that traders may expose to. One is interest rate risk, which related to the fluctuation of the interest rate and the other is counterparty risk, which mainly concerns about the in-the-money party facing the risk of possible default of the other party. Since interest rate swap is traded over-the-counter without any clearinghouse in between. The counterparty risk of defaulting may be significant (Fabozzi, 26).

#### *Case study – how interest rate swap benefit both parties*

Next we are going to demonstrate a very interesting case to better explain the incentives of entities which are willing to involve in an interest rate swap.

There are two corporations. Corporation Good is a great company and it was rated A by more than 8 rating agencies. It issues a \$1000 million fixed-rate bonds for 5 years at 6%. While Corporation Bad is not very promising and it was rated C- by rating agencies last year. Thus Corporation Bad can only issue high-yield debt, borrows \$1000 million from a bank at 6-month LIBOR plus 2%. Here is some more background information. The interest rate that must be paid by the Corporation Good and Corporation Bad in both floating rate and fixed-rate markets are as below. For Corporation Good, it needs to pay 6-month LIBOR + 40 basis points in floating rate market and 6 percent in fixed rate market; while for Corporation Bad, it needs to pay

6-month LIBOR + 200 basis points in floating rate market and 10 percent in fixed rate market.

Now a very smart financial analyst of Corporation Good figures out a way to lower its funding cost by swapping into floating rate debt and another very brilliant financial analyst of Corporation Bad also sees an opportunity to lower its cost by entering this interest rate swap that is offered by Corporation Good. Their interest rate swap is agreed as following. For Corporation Good, it is going to pay floating rate of 6-month LIBOR and to receive fixed rate of 6.2%; for Corporation Bad, it is going to pay fixed rate of 6.45% and receive floating rate of 6-month LIBOR. The 0.25% that is paid by Corporation Bad but is not received by Corporation Good is for swap dealer. Let's hear the reasons from the two smart financial analysts.

For Corporation Good, it needs to pay fixed-rate bonds issued of 6% and the interest rate swap of 6-month LIBOR. It will then receive 6.2% from the interest rate swap. Thus in total it pays  $6\% + 6\text{-month LIBOR} - 6.2\% = 6\text{-month LIBOR} - 20\text{bp}$ . This number is lower compare to what Corporation Good should pay within floating rate market of 6-month LIBOR + 40bp as we mentioned in the background. Thus the interest rate swap is a benefit contract for Corporation Good.

For Corporation Bad, it needs to pay 6-month LIBOR + 200 bp to the bank and 6.45% to the swap dealer. It will receive 6-month LIBOR from Corporation Good. Thus in total it pays  $6\text{-month LIBOR} + 200\text{ bp} + 6.45\% - 6\text{-month LIBOR} = 8.45\%$ . This is also a number lower than what Corporation Bad should pay within fixed rate market of 10% as we mentioned before in the background. By entering the interest rate swap contract, although both company exposed to certain amount of counterparty risk, they can achieve lower financial costs than by directly borrowing from the market and this is the fascinating power of the interest rate swap (*Fabozzi, 33*).

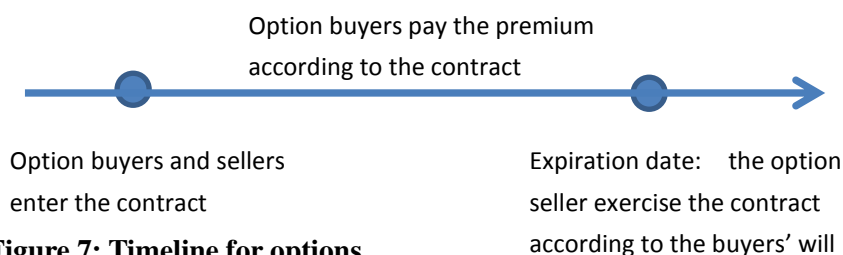
## Options

An option is a contract that grants an option, not an obligation for the option buyers to buy or sell things such as commodities at a specific rate to the option sellers,



also called as option writer, at a specific future time. The buyer will need to pay premium to the option writer in order to have this kind of option in the future. The specific rate that is agreed upon the option contract is called strike price and the specific future time when the seller needs to exercise the option is called the expiration date.

The timeline is demonstrated as below:



**Figure 7: Timeline for options**

Options can be classified by four different ways. First, if an option grants buying right to the option buyer, it is called a call. If an option grants selling right to the option buyer, it is called a put. Second, if an option can be exercised at any time up to the expiration date, it is called an American option. If it can only be exercised at the expiration date, it is called a European option. If the early option is possible but restricted, it is called a Bermuda option, which is a hybrid between American and European options. Third, there are also exchange-traded options and over-the-counter options. Exchange-traded options have standardized contract and there is a clearinghouse connecting the buyers and sellers. Over-the-counter options are tailor-made for big corporation or institution and usually investment banks and commercial banks will act as brokers in over-the-counter market. These options can be very complex and less liquid than the exchange-traded options. As the option product evolved, in order to compete with over-the-counter market, the flexible Treasury futures option was introduced in 1994 in Chicago Board of Trade (CBOT) which allow investors to customize option with certain limitations. Fourth, options can be classified according to its underlying. If the underlying is a fixed-income security, then the option is called options on physicals. If the underlying is interest rate futures, the option is named as futures options (Fabozzi, 36).

### *Pricing Model*

The value of the options is the intrinsic value, which is the economic value if the option is exercised immediately. For example, for a call option, if the current price of the underlying security is smaller than the strike price, the intrinsic value is negative and vice versa. A more mathematical way of valuing an option is by using the famous Black-Scholes option pricing model, which gives the valuation for European-style options. The most important concept to derive the Black-Scholes option is that we can reconstruct an option by a risk-free bond and a stock and we can get the same pay-off as the given option. Below are some important assumptions:

- The stock price follows a geometric Brownian motion process
- There is no transaction costs or taxes
- No dividends during the life of the option
- No risk-free arbitrage opportunities

Let's say the price of the option is a function of  $S_t$ , the price of the stock, and  $t$ , the time. We represent the option price as  $c(S, t)$ . The risk-free bond  $B$  with risk-free rate  $r$  will hold:  $dB = rBdt$  and the stock with geometric Brownian motion will hold:  $dS = \mu Sdt + \sigma Sdz$ . According to Ito's lemma<sup>1</sup>, the option price will hold:

$dc = \left( C_t + \mu SC_s + \frac{1}{2} \sigma^2 S^2 C_{ss} \right) dt + \sigma SC_s dz$ . First we need to reconstruct an option with  $x$  shares and  $y$  bond.

$$c = xS + yB$$

Different the above formula, we get:

$$dc = xdS + ydB = (x\mu S + yrB)dt + x\sigma Sdx$$

With the formula for  $dc$  above, we get two equations as results:

$$(x\mu S + yrB) = \left( C_t + \mu SC_s + \frac{1}{2} \sigma^2 S^2 C_{ss} \right)$$

$$x\sigma S = \sigma SC_s$$

We can easily get  $x = C_s$ . Plugging this into  $c = xS + yB$ , we get  $y = \frac{1}{B}(c - SC_s)$ . Plugging both  $x = C_s$  and  $y = \frac{1}{B}(c - SC_s)$  into  $(x\mu S + yrB) = \left( C_t + \mu SC_s + \frac{1}{2} \sigma^2 S^2 C_{ss} \right)$ . We finally obtain the Black-Scholes equation (Black,

637):

$$C_t + rSC_s + \frac{1}{2}\sigma^2 S^2 C_{ss} = rc$$

---

1. Ito's lemma for Brownian motion is

$(x, t) = \left( f_t + a(x, t)f_x + \frac{1}{2}b^2(x, t)f_{xx} \right) dt + b(x, t)f_x dz$ . In this equation  $a(x, t)$  and  $b(x, t)$  are deterministic function of  $x$  and  $t$ , and  $z$  represents a standard Brownian motion.

### *Risk exposure*

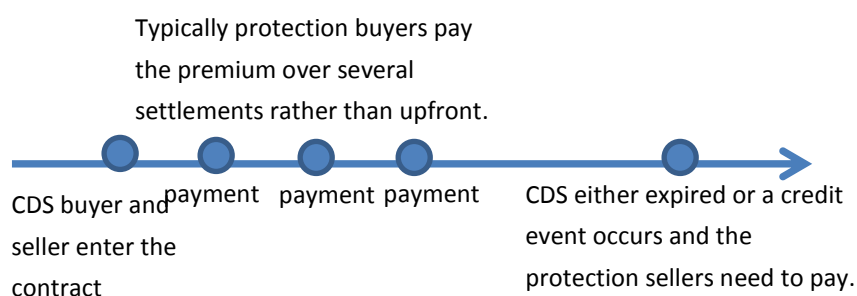
The risk exposure of option is a little different from the risk exposure for futures. It is asymmetric. The largest loss an option buyer will suffer is the premium and the gain that the buyer will get is going to be offset by the premium that he paid before. While for an option seller, he will gain at most the premium and the premium will also offset the downside risk. Concerning about counterparty risk, after the option buyers finish paying all premiums, he fulfilled his entire obligation to the option sellers. In contrast, option sellers are required to put down margin according to their position marked to market to ensure they will carry out the contract if the option buyers choose to exercise their right.

## **Credit Default Swaps**

In credit default swaps, there are two parties and a reference entity. One can be called the protection buyer who pays a fee to the protection seller so that when any credit event happens to the reference entity, the protection buyer will receive payments from the seller. If nothing happens by the end of the contract, then the protection seller will win the fee that protection buyer pays at the beginning and doesn't need to pay out anything. There are normally eight types of credit events, which include bankruptcy, credit event upon merger, cross acceleration, cross default, downgrade, failure to pay, repudiation or moratorium and restructuring. Here for the convenience of explanation, we will assume that the credit event happens to a company is bankruptcy, also called as defaults. In real world, Credit Default Swap is normally five years and the protection buys pays the fee separately rather than upfront.

Each swap premium payment can be calculated by multiplying notional amount, swap rate and the percentage of days within one payment period over 360. The protection buyer is possibly the one who holds bonds of the reference entity and exposes to the default risk of the reference entity. There is another condition for the protection buyer to buy credit default swaps, which is for speculative reasons and they think that the reference entity is very risky and going to default. For these buyers, they are like holding short position of the reference entity's bonds and the protection sellers are like buying reference entity's bonds. And credit default is like a tool that can create short position of bonds for individual, who is not very possible in real world without CDS and create a leveraged credit exposure for the protection seller since they are bearing similar risk as holding reference entities' bonds, while not paying the principal. When reference entity defaults, there are two kinds of settlement. One is cash settlement, which means the protection seller will pay the amount of money that is determined by the decline of the reference entity's bond price to compensate the loss for the protection buyer. The other method is physical settlement. The protection buyer will give the bad bonds of the reference entity to the seller and the seller is promised to pay the protection buyer the par value of the bonds. The Credit Default Swap is currently taking the largest part of the credit derivatives market.

Below is a timeline to demonstrate credit default swap:



**Figure 8: Timeline for CDSs**

Except from the customized credit default swap arrangements between two counterparties, Dow Jones also manages Credit default swap index, which is essentially a standardized basket of credit risk of many corporations as reference entities. The biggest difference between a normal credit default swap and a credit default swap index is that the premium payments stop when a credit event happens to

a normal credit default swap; while for CDS index, since it has a basket of reference entities, when one of the corporation defaults, the index buyers need to continue paying premium, but just with less money because the notional amount decreases as a result of the corporation defaulting (Fabozzi, 48).

### *Pricing Model*

Let's start exploring from a simple example. Suppose we have a CDS with swap rate of 300 basis points and face value of \$10 million. This means the protection buyer needs to make quarterly payments of  $\$10\text{million} \times 0.03 \times 0.25 = \$75,000$ . Then let's assume after 1 month, the reference entity suffers a credit event. We also know the recovery price as well, which is \$45 per \$100 of face value (recovery price can be interpreted as the remaining value of the reference entity after the credit event). After the credit event, the protection seller then needs to pay the protection buyer for the loss, which is  $\$10\text{million} \times (100\% - 45\%) = \$5.5\text{million}$ , and the protection buyer needs to pay the 1-month accrued premium, which is  $\$10\text{million} \times 0.03 \times \frac{1}{12} = \$18,750$ .

Next we are going to explain how CDS mark-to-market value works. Let's consider a protection buyer purchases 5-year protection on a corporation with swap rate of 60 basis points and tries to value his position after one year. On the date after one year, the 4-year CDS is quoted with 170 basis points in the market. Then the

Mark-to-Market Value

$$\begin{aligned}
 &= \text{current market value of 4-year Protection} - \text{expected present value of 4-year premium leg at 60 basis points} \\
 &= \text{expected present value of 4-year premium leg at 170 basis points} - \text{expected present value of 4-year premium leg at 60 basis points.} \\
 &= 170bp \times \text{Risky PV01} - 60bp \times \text{Risky PV01} \\
 &= 110bp \times \text{Risky PV01}
 \end{aligned}$$

The Risky PV01 (RPV01) is defined as the expected present value of 1bp paid on the premium leg until a credit event happens or the CDS expires. In order to calculate this RPV01, we will need a more complex model because we need to consider the possibility of a credit event happening over the CDS contract period which will essentially terminate the premium paying. Now for the protection buyer to realize this mark-to-market value gain, he can unwind it with the protection buyer for

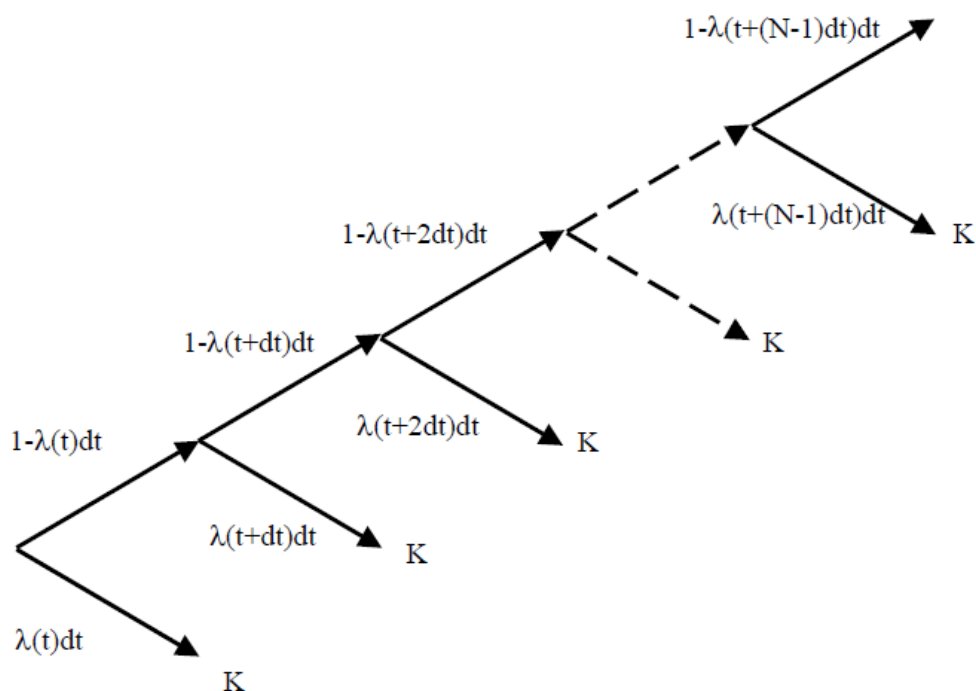
a cash rewind value, which will be equal to the mark-to-market position.

The most common approach to model the probability of credit events of the reference entities is the reduced-form approach. The probability of a credit event is modeled as Poisson counting process, which means the probability of a credit event happening within  $[t, t + dt]$  conditional on the surviving to time  $t$  and is proportional to a function  $\lambda(t)$ , which is called as hazard rate.

$$\Pr[\tau < t + dt | \tau \geq t] = \lambda(t)dt$$

We can interpret this model as the reference entity defaulting in a time  $dt$  with probability  $\lambda(t)dt$  or it surviving through the time  $dt$  with probability  $1 - \lambda(t)dt$ . We are also going to simply assume that the hazard rate process is deterministic, which also means it is independent of interest rates or recovery rates. Here is a picture that can clearly demonstrate this model.

**Figure 4. The equivalent of a binomial tree in the modeling of default in which the tree terminates and makes a payment  $K$  at default**



**Figure 9: The equivalent of a binomial tree in the modeling of default.**

According to this model, we can compute the continuous time survival probability to time  $T$  conditional on surviving to time  $t_v$  by having  $dt \rightarrow 0$ . And then

$$Q(t_v, T) = \exp\left(-\int_{t_v}^T \lambda(s)ds\right)$$

Next we are going to value the premium leg, which includes all the premiums made until the credit event happens. Let's assume there is in total N payments if the CDS makes to maturity and the swap rate is  $S(t_0, t_N)$ . We are also going to ignore premium accrued for now. Then the premium leg of existing contract is:

$$\text{Premium Leg PV}(t_v, t_N) = S(t_0, t_N) \sum_{n=1}^N \Delta(t_{n-1}, t_n, B) Z(t_v, t_n) Q(t_v, t_n)$$

$\Delta(t_{n-1}, t_n, B)$  is the day count fraction,  $Z(t_v, t_n)$  is the Libor discount factor and  $Q(t_v, t_n)$  is the arbitrage-free survival probability of the reference entity conditional on surviving to  $t_v$ . Next let's consider the premium accrued, which we will need to consider the possibility of defaulting between the two payments. Then formula will look like:

$$\text{Premium Leg PV}(t_v, t_N) = S(t_0, t_N) \sum_{n=1}^N \int_{t_{n-1}}^{t_n} \Delta(t_{n-1}, s, B) Z(t_v, s) Q(t_v, s) \lambda(s) ds$$

The above formula can be approximated as the following equation by taking the average accrued premium as half of the full premium which is set to be paid at the end of the payments period.

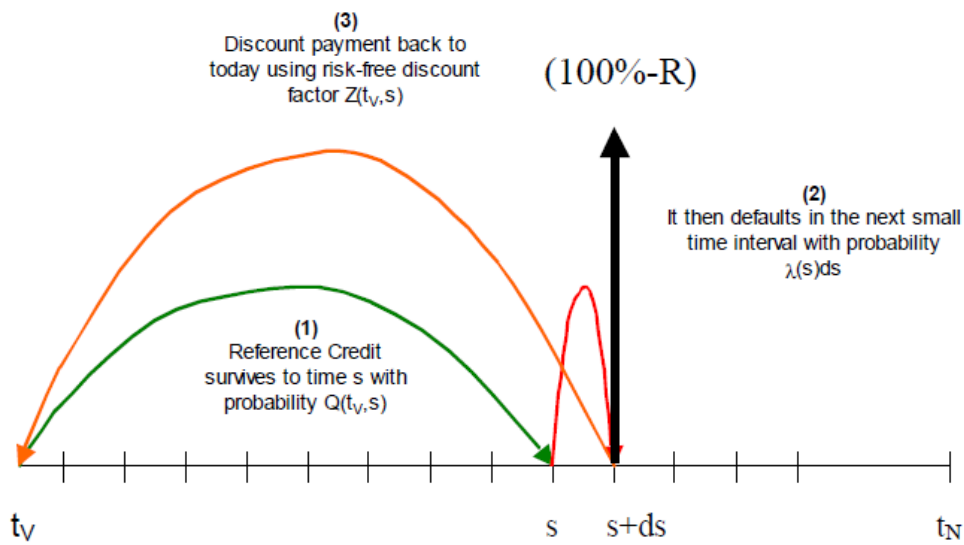
$$\begin{aligned} \text{Premium Leg PV}(t_v, t_N) &= S(t_0, t_N) * RPV01 \\ &= \frac{S(t_0, t_N)}{2} \sum_{n=1}^N \Delta(t_{n-1}, t_n, B) Z(t_v, t_n) (Q(t_v, t_{n-1}) - Q(t_v, t_n)) \end{aligned}$$

Thus we get equation for RPV01 for accrued premiums as

$$RPV01 = \frac{1}{2} \sum_{n=1}^N \Delta(t_{n-1}, t_n, B) Z(t_v, t_n) (Q(t_v, t_{n-1}) - Q(t_v, t_n))$$

Next we are going to value the protection leg, which is contingent payment of (100%-recovery rate) of the face value of the CDS depending on the credit event. The following timeline can clearly demonstrate the calculation logic of the protection leg.

**Figure 6. Steps in the calculation of the expected present value of a recovery rate which is paid at the time of a credit event**



**Figure 10**

The formula can be written as the following:

$$Protection\ Leg\ PV = (1 - R) \int_{t_V}^{t_N} Z(t_V, s) Q(t_V, s) \lambda(s) ds$$

Here R represents the expected recovery rate if a credit event happens. Z is the Libor discount factor and  $Q(t_V, s)$  is the arbitrage-free survival probability of the reference entity living to time s.

With the present value of both protection leg and premium leg available, we can then equate them to get the breakeven swap rate, plug in the RPV01 value to get the mark-to-money value and find out the current market spread to maturity (O’Kane, 1).

**Risk exposure**

The protection buyer is facing the risk of not receiving the payments promised by the seller. Thus a common way to prevent such a risk is asking protection the seller to post collateral for its responsibility to pay the protection buyer whenever a credit event happens.



## Cash Flow Collateralized Debt Obligations

A collateralized debt obligation (CDO) typically has a CDO portfolio manager. He is in charge of first raising money by issuing its own bonds and then invests the money he raises into either bonds, loans, or other assets. The assets the manager invests are the underlying assets of the portfolio. Then the payments on the asset portfolio can be used to repay the bonds that the CDO raises its money from. The manager will also collect fees for actively managing the portfolio. This is only the basic idea of CDO. The most important feature of CDO “credit tranching”, which means a CDO can issue different classes of bonds, for example, senior debt, mezzanine debt, subordinate debt and equity. Each class is exposed to a different level of risk. The most senior debt is the least risky debt because the more senior class will have higher priority to receive the asset repayments than the less senior class. Thus naturally, the less senior class demands higher interest rate because it is exposed to greater risk. CDO is invented in 1987 and then became a fast growing sector. There are two kinds of bankers who can take advantage of CDOs. First kind is called arbitrage CDO, which bankers can make money from the spread between the yield that CDO earns on the underlying asset and the payments that CDO pays out for CDO investors. The other kind is called balance sheet CDO, which can help a bank move its assets from the balance sheet to a CDO portfolio so that there will be less asset on the bank’s balance sheet.

### *CDO life-cycle*

First, ramp – up phase is when the CDO manager raises money from issuing CDO bonds and then uses the money to set up the initial portfolio. There are regulations from CDO governing documents to constrain the portfolio’s average maturity date and other parameters. Second is revolving period when the CDO manager receives the payments from underlying assets and repays CDO bonds. When an underlying asset reaches its maturity date, the CDO manager doesn’t have to amortize some of the CDO bonds, and instead he can reinvest the money into other assets. Third phase is the amortization phase when the manager amortizes all the

CDO's asset and finishes repaying all the bonds that the CDO issues.

### *Different types of CDOs*

CDOs can be classified by the underlying assets. Below is a table for this kind of classification:

<b>Underlying Portfolio</b>	<b>Different types of CDO</b>
<b>Bonds</b>	Collateralized bond obligation (CBO)
<b>Loans</b>	Collateralized loan obligation (CLO)
<b>Asset-backed Securities or Mortgage-backed securities</b>	Structured finance CDO
<b>Mix of bonds, loans, asset-backed Securities</b>	Multisector CDO
<b>Other CDOs</b>	CDO squared
<b>CDS</b>	Synthetic CDO

**Table 1: Classification of CDOs**

CDO can also be classified by its repayment method. If the CDO repays its bonds with cash and a set interest rate, this is called cash flow CDO. If the CDO repays its bonds depending on the market value of the underlying assets, this is called market value CDO.

### *Synthetic CDOs*

Synthetic CDO's underlying portfolio consists of credit default swaps. Synthetic CDO works slightly different from the normal CDOs. The CDO portfolio acts as a protection seller of a credit default swap. As we mentioned before, the CDO portfolio can then receive payments from protection buyers. The CDO may also invest money on low risk securities and then the payments from the underlying CDS and interest from low risk securities can pay the bonds that the CDO issues. The reason that synthetic CDO invests money on low risk securities is because when a credit event happens in the underlying CDS, CDO manger can use the money in the low risk securities to pay the protection buyer according to the CDS contract. The effect of the credit event in synthetic CDOs is similar to the effect of when the asset defaults within regular CDOs because both events will potentially decrease the ability

for the CDO to repay its bonds and decrease the CDO's rating.

In the synthetic CDO, there is an additional class called unfunded class. The investors who buy this class will act like a protection seller themselves. They don't need to pay anything upfront and they will receive payment from the CDO portfolio as a protection seller. However, when a credit event happens, the unfunded class investors will need to pay money to the protection buyer through the synthetic CDO (Fabozzi, 119).

### *Part III: 2008 Financial Crisis*

The year 2008 marked down an unprecedented global financial crisis in human history. From mid-2007 to 2008, the world experienced a series of collapse of financial institutions, bailout of banks by governments, and downturns in stock market worldwide.

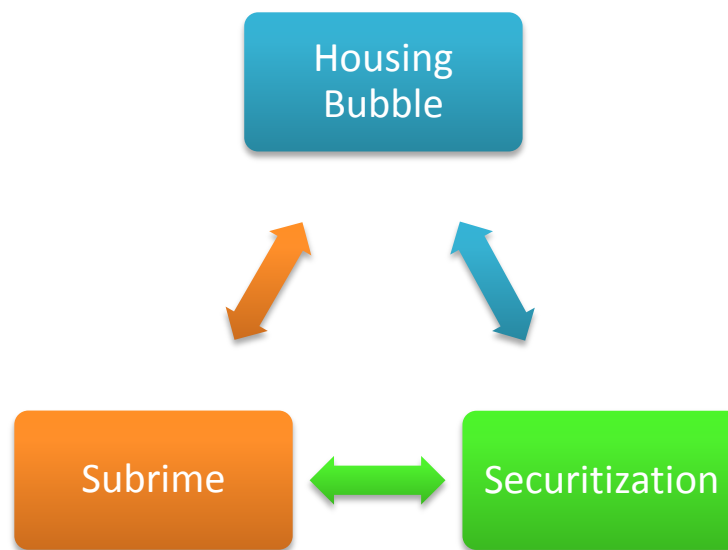
Three out of the five largest investment banks failed – Lehman Brothers filed for bankruptcy protection, Merrill Lynch was purchased by Bank of America, and Bear Stearns was absorbed by JP Morgan. In addition to this, Fannie Mae and Freddie Mac were completely taken over by the federal government. American International Group (AIG) survived only after receiving an \$85 billion capital injection from the government. Starting from Monday October 6<sup>th</sup>, the stock market declined for a week straight in which the Dow Jones Industrial plunged 1874 points.

Although the exact cause of this crisis was still under debate, this crisis was largely coined with three words: Housing Bubble, Securitization, and Subprime Mortgages. This section was devoted to explore the cause of the 2008 crisis around these three concepts, examine the major game players, and understand how they interacted with each other.

## **Overview**

The crisis was largely the net product of the three interactive factors and it is hard to isolate any one of them to account for the crisis. The three factors affected

and reinforced each other in an endless loop only to worsen the situation. Their interaction can be illustrated as follows.



**Figure 11: Three concepts in 2008 crisis**

As the start, housing bubble planted the seed of evil. Initially fueled by a healthy increase in natural demand after World War II, rising house prices was soon artificially boosted by well-intended government policies. In order to increase homeownership, the fed government broke the law of natural demand and supply and created a favorable environment for home buyers fueled by “cheap money”. Urged by the government, mortgages lenders began to lower their standards to include more people with weaker credit. Subprime mortgages were created to cater to the growing appetite of the real estate industry. House prices continued to skyrocket. Soon after investment banks on Wall Street saw profits in subprime, they joined mortgages lenders. In order to be less exposed to the risk of subprime, banks securitized subprime mortgages and created complicated second level securities backed by these mortgages. They then shipped thousands of them to investors all over the world. With rapid growth in economy, developing countries were more than happy to purchase these financial products with their excess capital. Capital flew back into U.S. As securitization proved successful in diversifying and managing risk, more funds went to subprime loans. In turn, more subprime loans increased demand for housing, leading to even higher house prices.

The chain effect of housing bubble, subprime mortgages, and securitization process moved in a cycle just like electricity moved in a short circuit. They ran freely without disturbance from outside and overheated each other until the “wire” could no longer bear the heat. The crisis then broke out.

## Key Concepts

### Housing Bubble

One thing noticeable about the outbreak of the crisis was the coinciding collapse of U.S real estates in 2007. Before then, house prices had been skyrocketing at an unusual pace for a decade. Below are some quick facts.

- House prices rose steadily from the 1990s to 2006.
- The appreciation in house prices exceeded 10% per year from early 2004 to early 2006
- The home-ownership rate rose to a record level of 68.6% of households by 2007
- In Boston, the median home, which had sold for a reasonable 2.2 times median income in the mid-90s, rose to 4.6 times income in 2000s.

Notably, the last piece of information revealed a very interesting fact about the U.S house market – its growth went far ahead of the growth of household income. As Fannie Mae’s managers pointed out (Lowenstein), if we overlay a graph of house prices with a graph of incomes, the two lines tracked each other from 1976 to 1999. During that period, home prices grew in response to income growth. As a matter of fact, every blip in income growth was reflected in a corresponding change in home prices. As time moved into 20<sup>th</sup> century, however, the two lines deviated. Household income experienced a growing rate of only 2 percent while home prices rocketed alone. The growth of home prices by itself was very unnatural.

Generally, a mere rise in the price of an asset does not necessarily give rise to an inflation bubble (Lowenstein). As long as the increased price is aligned with changes in the assets’ demand or supply, the rise in value is healthy to the economy. A

bubble market, on the other hand, is one that lost its connection to the natural demand and supply. Since there was no sign of increased buying power in the 2000s – household income did not gain much, rapid appreciation of houses turned the housing market into a bubble market. The bubble market then fostered the rapid growth of subprime mortgages, with the latter transformed into securities which were traded all over the world and widely spread out the seed of the crisis.

If it was not the income to back the home price up, it must have been something else that could account for the expanding house market. As analysis later in this report showed, the U.S. government, along with Wall Street banks and investors overseas, was responsible for the start and growth of the housing bubble.

### **Subprime Mortgages**

Subprime mortgages, to the opposite of prime mortgages, are a type of mortgages that are made to borrowers with shaky credit ratings. The word “Subprime” refers to the low credit rating of the borrower. Subprime mortgages borrowers usually have a credit rating below 600 on a scale of 300 to 900 while a typical consumer can have as high as 700.

Since the borrowers of subprime mortgages are riskier – the chance that they make late payments or even go bankruptcy is much higher, interest rates charged on subprime mortgages are usually higher than prime mortgages. Still, they can vary wildly based on factors such as the actual credit score of the borrower, size of down payments and number of late payments.

Subprime Mortgages played a very important role in the 2008 financial crisis. During the pre-crisis period, especially into the 2000s, subprime mortgages issued to home buyers enjoyed a sharp rise in response to the increase in the U.S. house prices. The percentage of low-quality subprime mortgages rose from 8% to approximately 20% within 2004-2006 timeframe. Among them over 90% were adjustable-rate mortgages which are mortgages with an initial low interest rate that would grow much higher in a process called mortgage reset.

Subprime mortgages would normally have added much instability to the

banking system and the U.S. economy. However, when house prices continued to rise, no problem surfaced in the early 2000s. Nobody questioned about subprime mortgages even if the borrower paid down little or even paid up. It was largely believed that, if a borrower was short of money paying his first mortgage, he could always take on a second one because higher value of his home gave him more collateral. After all, he could sell his house at a higher price to pay off the first mortgage.

As subprime mortgages defaulted moderately and subprime lending appeared profitable, Wall Street banks joined mortgage lenders in the game of subprime. Instead of being exposed directly to subprime mortgages, investments created securities backed by these mortgages through the process of securitization. They also introduced Credit Default Obligation (CDO) as the second layer of securitization. Through these products, subprime mortgages were packaged and shipped to different parts of the world and traded as completely different products to various investors.

Subprime mortgages began to fall back to their real value soon after house prices declined steeply after 2006. Refinancing became much more difficult to home buyers without the backup of real estate boom. They soon found themselves paying even higher interest than what they already could not afford. Securities backed by subprime mortgages slumped in their value, deeply wounding investment banks. Global investors no longer purchased mortgage-related products as a result of loss of credit confidence. Subsequent to the deflation of housing bubble, subprime mortgages generated waves of default and failure throughout the global financial system.

## **Securitization**

Unlike the previous two concepts which are widely accepted as ill-advised, securitization gained its bad reputation only recently, just because of the subprime mortgage crisis. Before then, it was thought as an innovative financial practice to manage risk.

### **Definition**

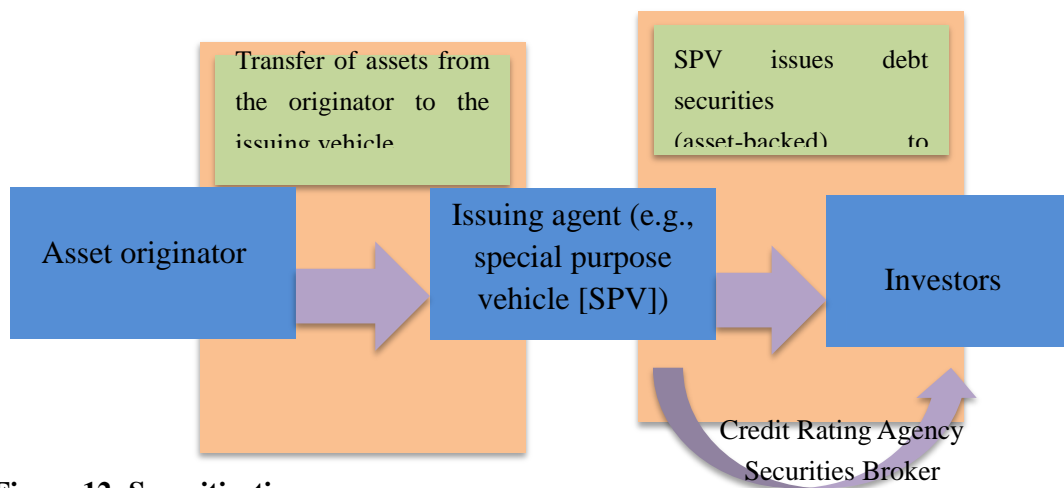
Securitization is the process of pooling various types of assets and

repackaging them into interest-bearing securities. The interest and principal payments from the assets (i.e. houses) are passed through from the originator (i.e. home mortgage issuer) to purchasers of the securities (i.e. investors).

### Historical Review

Securitization was started in the 1970s. Back then, U.S. government-backed agencies were the only players in the business. Fannie Mae and Freddie Mac, two giants in the industry, securitized only prime mortgages. Starting from the 1980s, other income-producing assets began to be securitized. The market of securities backed by risky subprime mortgages also grew drastically. Entering into the 2000s, increasing numbers of financial institutions joined the game of securitization as they saw lucrative profits out of the business of subprime mortgages. Investment banks, insurance companies, pension funds, and hedge funds were eager to bite away their share. The scale and prevalence of the crisis explained how broadly and wildly securitization was practiced in the decade preceding the outbreak of the crisis.

### Process Illustration



**Figure 12: Securitization process**

The process basically involves two steps. It starts with a company which wants to remove the loans or other assets they hold onto out of its balance sheet. The company, also called the originator in this process, pools these unwanted assets into so-called reference portfolio. It then sells the asset pool to an issuer (arranger), such as a special purpose vehicle (SPV). SPV, a bankruptcy-remote trust set up by a financial institution, has its specialty in acquiring these assets and writing them down the



balance sheet for the originator. At the second step, the issuer finances the acquisition of the pooled assets by issuing tradable, interest-bearing securities that are sold to capital market investors (Jobst 2008). The investors receive fixed or variable rate payments from a trustee account funded by the cash flows generated by the reference portfolio (Jobst 2008). Thus, credit risk is transferred from issuers to investors.

In the real world, rating agencies and securities brokers also facilitate the process. They help to bridge the information asymmetry between SPV and investors. Rating agencies assign investment grade to the securities traded in the market. The grades provide guideline to investors. Securities brokers provide counseling to investors.

### **Tranches**

Tranching is an innovative way issuers created to market the reference portfolio. Instead of selling the entire portfolio as a big chunk, they slice it into tranches, each of which has a different level of risk and marketed separately. Investment return varies dramatically among different tranches. The safer the tranche is, the higher its priority to get its share of income generated by the underlying assets. On the other hand, the riskier the tranche is, the higher the rate of return.

Based on the seniority of the risk, tranches are classified as junior, mezzanine, and senior tranches. The most risky junior tranche, usually the smallest of the three, bears most of the credit exposure and receives the highest return. The senior tranche, to the opposite, are the least expected to default. However, the expectation is very sensitive to changes in the quality of the underlying asset. When the borrowers of subprime mortgages began to have problems making payments, junior tranches were first to be affected but loss of confidence among senior tranche holders fired panic among investors. That caused a fire sale of such securities.

### **Rationale for Securitization**

Securitization represents a new way for companies to raise money, the way that actually increases capital availability. Suppose a leasing company wants to raise cash. Under traditional procedures, the company has to do so by issuing bonds or

loans. Its ability to do so, poorly or well, and the cost incurred, is largely determined by the overall financial health of the company and its credit rating. If the company is not functioning so well recently or if it does not meet some standard criteria, it will not be able to issue quality bonds or loans that are attractive to investors at a low cost. If only it could sell some of its leases directly to potential buyers, that would have solved its problem of issuing bonds or loans. However, there is no secondary market where individual leases can be traded. But by securitization, the company can raise the cash it wants by pooling the leases and selling the package to an issuer, with the latter convert the pool into a tradable security.

Moreover, securitization lowers the cost of capital. Since the assets are now detached from the originator's balance sheet, the credit grade of the security issued is no longer tied to the overall credit rating of the originator. This means issuers can finance the pool of assets they purchase more cheaply than normal and in turn costing originator less to sell the pool. For example, by securitization, a company with an overall "B" rating with a triple-A rated asset pool is able to raise funds at the rate for triple-A instead of "B". Also unlike conventional debt, securitization does not inflate a company's liabilities because the assets are now off-balance-sheet.

Investors also benefit from securitization because it creates a broad array of attractive investment options. From the same asset pool, people who want steady and stable return with the least risk exposure can take away senior tranches while speculators can gamble with their risky junior tranches at extremely high rate of return. The flexibility of securitization transactions also help pension funds and other collective investment schemes which require a diverse range of highly rated long-term fixed-income investments beyond what the public debt issuance by governments can provide (Jobst 2008).

### **Securitization and 2008 Financial Crisis**

Despite its great attributes in raising capital, lowering cost, and diversifying investment options, securitization played its negative to the extreme in the crisis.

The process of securitization encouraged predatory lending in the relationship

between the mortgagor and originator and the relationship between the originator and the issuer. Predatory lending describes fraudulent and deceptive practices by the loan lender that aim to mislead and take advantage of the borrower. In particular, subprime mortgagors could be very financially unsophisticated. They were either unaware of the variety of mortgage products available to them or unable to make a choice between the available options. That opened the door for predatory lending. Especially when the originator saw the huge profits generated from subprime mortgage securitization, it had stronger tendency to trick the borrower into buying unsuitable mortgages. Predatory lending also occurred between the originator and the issuer. When the pool of mortgage loans was bought by the issuer, it had the responsibility to check the originator's credit status and the quality of the underlying mortgage pool. The detail of the deal between them was then finalized by the result of the examination. However, with the information asymmetry existing between the originator and the issuer, the originator had the tendency to misrepresent the quality of the mortgage borrower in order to write more mortgages off their balance sheet. This led to mortgage fraud. Predatory lending and mortgage fraud clearly knocked financial soundness off the chain of securitization and brought unpredictable instability to the whole process. Theoretically the issuer could put due diligence on the originator to help prevent the spread of mortgage fraud but in reality, especially in those days when securitization was fervently practiced by Wall Street investment banks, the close check was usually skipped.

Securitization also created moral hazards. A moral hazard refers to a situation where one party holds responsibility for another but has the incentive to put his or her own interests first (Dowd 2009). Most moral hazards involve excessive risk-taking: why not take the gambling if "heads I win, tails you lose" is the rule? After all, I do not have to assume any loss since you bear the risk for me. Moral hazards, if not controlled properly, often lead large-scale risk-overtaking just as what happened in the crisis.

Traditionally when a mortgage lender granted a loan to the borrower, he or she

held onto it until its maturity. If the mortgage holder defaulted, the lender assumed all the loss. Therefore, it was natural for the lender to scrutinize the borrower before granting a mortgage. In this way, not many subprime borrowers would have been successful getting a mortgage. However, under the new securitization process, the originator did not have the incentive to make serious check on the borrower because they did not expect to hold the mortgage for very long. They were only concerned about the payment it got for originating the mortgage. Now even the doziest mortgage broker could originate subprime mortgages for the least creditworthy borrowers (Dowd 2009). Unfortunately, this giant Ponzi scheme could only last as long as the housing bubble continued to inflate and new entrants continued to come into the market. Once interest rates started to rise and house prices began to fall, the supply of suckers inevitably dried up and the whole edifice began to fall in on itself (Dowd 2009). In the 2008 financial crisis, investment banks and investors all over the world helped to “relay” the crappy subprime mortgages in a string of securitization. When the head of the string burned soon after the collapse of housing market, the entire string got fire in a flash.

Furthermore, securitization introduced complex financial products which attracted misuse and abuse. Securitization was initially used to finance simple, self-liquidating assets such as mortgage. However, potentially all types of assets with a steady cash flow could be structured into a reference portfolio which after pooling could be converted to securities. In addition to mortgages, corporate and sovereign loans, consumer credit, project finance, lease/trade receivables, and individualized lending agreements could all be used to back up securities. The securities created this way generally are called asset-backed securities (ABS) though those securities backed by mortgage loans are called mortgage-backed securities (MBS) more precisely. Moreover, a variant called collateralized debt obligation (CDO) was created more recently to include an even more diverse range of assets.

Given the complexity and variety of instruments, it was very difficult for the investors to figure out what was the right choice for them. It was even difficult for

rating agencies or securities brokers to understand the instruments they were grading or offering counseling on. Commented by Allen Greenspan, former chairman of Federal Reserve, “I have not shallow background in mathematics and I have access to hundreds of top math PhDs. But even I could not completely understand those CDOs. Could anybody in the world possibly understand them? I doubt.” (House of Cards 2009)

Since nobody could have a thorough insight, buyers could largely be manipulated by sellers. That was exactly how Narvik, Norway, a town far above the Arctic Circle, was fooled to buying CDOs in 2006 (House of Cards 2009). The town of 17,000 people, suffering from a shrinking population and a growing budget deficit, needed money. So when sales people showed up, selling what they claimed “safe with high yield” products from Citigroup, the mayor and her counseling were overjoyed. From everything they examined they found no sign of problems except the content of the product remained mysterious. The product was rated triple-A – that was all they need to know to invest. Narvik thought their budget problem was over. However, when the subprime mortgage broke out, the products they held suddenly turned into nothing. Now the town is crashing and its residents have to pay the price of this poor investment for over a decade.

Securitization can be a valuable tool. It was the oversight in regulation and greedy in human nature that spoiled the good will of the tool. The lesson is: don’t just throw out the water at the baby; monitor the temperature and change the water accordingly. Water can sustain you as well as kill you!

## **Major Game Players**

### **The U.S. Government – Over interfered the housing market**

The initial rise of the housing bubble was thought to be a product of government interference with the natural demand and supply.

During 1930’s Great Depression, the United States had experienced the

greatest mortgage crisis ever in its history. About half of mortgage was in default and the amount of mortgage lending had fallen by about 80 percent. In response to this, in 1938, Fannie Mae was created as a government agency to help with home mortgage lending market. In addition, Congress chartered Freddie Mac in 1970. These two companies functioned as private corporations but were sponsored by the government. Their intimate relationship with the government resulted in the companies' dominance in the industry on the one hand. On the other hand, however, it restrained the companies' freedom from political influence.

Initially Fannie and Freddie only held onto prime mortgages. They set strict industry standards that only those with strongest credit would be issued home mortgages. Starting from the 1990s, however, they were pushed by Congress to accept documented loans available to borrowers with spotty credit history. The goal claimed by the government, was to increase home ownership. By lowering standards, it was hoped that more people could afford loans issued by Fannie and Freddie.

Originally backed by the Democrats and the Clinton Administration, the policy was further pushed by the Bush Administration. As time moved on into year 2001, the overall mortgage environment became more and more favorable for home buyers.

The terrorist attack on September 11<sup>th</sup> 2001 also unintentionally helped with the on-going housing boom. As the result of the attack, the country was immersed in a mood of terror and depression. The economy halted as people were afraid to go out shopping. Allen Greenspan, the Chairman of Federal Reserve at that time, feared that a financial crisis of decades would come. The only way to prevent the crisis, as he believed, was to encourage people spending. Starting from 2001, Greenspan made a series of cuts for short-term interest rates all the way to 2003 until the rates finally dropped to only 1 percent – history lowest of the generation. The inflation rate was at a time equal to or even greater than the interest rate. Banks were effectively borrowing money for nothing. People were spending far more than they could afford.

## Investment Banks

Among all that experienced huge loss in the crisis, investment banks on Wall Street have won the least sympathy. Quite to the contrary, they were blamed intensely for their irresponsible conducts and insatiable greedy. The oversight in the risk they were taking in securitization and subprime mortgages not only cost themselves high price but also blew the storm of credit crisis over the globe.

Investment banks helped to supply capital from oversea investors to the U.S. housing market and sustained the housing bubble for quite a long time. Breaking the exclusive right of Fannie & Freddie to securitize, investment banks offered an alternative for mortgage lenders to write the loans off their balance sheet in the 2000s. They soon became a steady source for these lenders. By issuing securities converted from subprime mortgages and other kinds of other debt overseas and recycling capital back to the domestic housing market, investment banks expanded the appetite of the housing industry and subprime mortgage business dramatically.

To be less exposed to direct risk transfer from subprime market, investment banks sought ways to further enhance the practice of securitization. They soon invented CDOs – a vehicle to offload unwanted risk and make a fortune in the process. The table below lists the top CDO underwriters and how many deals they had from 2002 to 2007.

<b>Underwriter</b>	<b>2002</b>	<b>2003</b>	<b>2004</b>	<b>2005</b>	<b>2006</b>	<b>2007</b>	<b>TOTAL</b>
<b>Merrill Lynch</b>	0	3	20	22	33	18	107
<b>Citigroup</b>	3	7	13	14	27	14	80
<b>Credit Suisse</b>	10	7	8	9	14	6	64
<b>Goldman Sachs</b>	3	2	6	17	24	7	62
<b>Bear Stearns</b>	5	2	5	13	11	15	60

<b>Wachovia</b>	5	6	9	16	11	5	52
<b>Deutsche</b>	6	3	7	10	16	5	50
<b>Bank</b>							
<b>UBS</b>	5	2	5	10	16	6	35
<b>Lehman</b>	3	4	3	6	5	6	35
<b>Brothers</b>							
<b>Bank of America</b>	2	2	4	9	10	2	32`
<b>TOTAL</b>	47	44	101	153	217	135	697
<b>DEALS</b>							

**Table 2: CDO underwriters**

Sometimes it was hard to sell the mezzanine CDO tranches because unlike the most senior tranches, they usually did not get an investment grade. To solve this problem, investment banks repackaged them into new CDOs. In this way, the mezzanine CDO tranches were turned into part of new AAA bonds. This development was the notorious “CDO squared” or sometimes “CDO cubed”. On the one hand, it held investors more distant from the underlying mortgages they were actually investing in. Investors were literally investing in something riskier than they thought. On the other hand, it deteriorated the CDO quality and made investors more exposed to the risk without them knowing it.

The banks went further and further down the road of securitization. They often conducted many iterations of securitization on their products. It was shown that Merrill Lynch created “CDO<sup>2</sup>” with as much as 15% of the assets from their prior CDO transactions. It also bought 59% of its CDO tranches that were resold into CDO<sup>2</sup>. The table below summarizes the amount of repackaging done by the banks. In particular, Merrill Lynch topped with an average 4.79 iterations on their CDO assets. As the amount of repackaging increased, the complexity involved in these products multiplied. It became more and more difficult to perform analysis on their underlying collaterals.



<b>Bank</b>	<b>Largest CDO Buyer of Bank's CDOs</b>	<b>Largest CDO Supplier to Bank's CDOs</b>	<b>LEVEL</b>
<b>BoA</b>	Citigroup	Bank of America	3.00
<b>Barclays</b>	Merrill Lynch	Barclays Capital	2.79
<b>Bear Stearns</b>	Citigroup	Bear Stearns	3.94
<b>Citigroup</b>	Citigroup	Citigroup	4.17
<b>Credit Suisse</b>	Merrill Lynch	Credit Suisse	2.07
<b>Deutsche Bank</b>	Merrill Lynch	Deutsche Bank	1.62
<b>Goldman Sachs</b>	Goldman Sachs	Goldman Sachs	2.32
<b>JP Morgan</b>	Merrill Lynch	JP Morgan	2.79
<b>Lehman</b>	Merrill Lynch	Lehman Brothers	2.99
<b>Merrill Lynch</b>	Merrill Lynch	Merrill Lynch	4.79

**Table 3: CDO repackaging level (Barnett-Hart 2009)**

Investment banks soon felt tired of relying on mortgage lenders and other loan originators to provide them with the collateral required for CDOs. Instead, they wanted to be their own originator. They began to repackage their own collateral into CDOs. Bear Stearns underwrote CDOs with as much as 30% of the collateral issued by their in-house RMBS (*Residential Mortgage-Backed Securities*) business. Merrill Lynch bought 32% of all its in-house RMBS used in CDOs. Playing the role of both an originator and an issuer enabled the banks create and trade ABS or MBS more freely. Now that nobody could restrain them generating huge profit out of the business, nobody could also stop them over-taking risks.

While the Wall Street was wild packaging securities they created and enjoyed “riskless” profits from CDOs, they were winding up tremendous amount of risk due to so-called “super senior” tranches. Super senior tranches were created by chopping the uppermost tier of the AAA portion of a CDO. The tier held “super” low credit risk and all the lower tiers could be sold for higher yield than original. Many banks kept these super senior tranches to themselves because: 1. Very least capital was required to keep AAA securities. 2. It was difficult to sell these super senior tranches because

of low yield. A JP Morgan report revealed that banks held around \$216 billion worth of super senior tranches of ABS CDOs in 2006 and 2007 (Barnett-Hart 2009).

The banks did not worry much about their increasing exposure to SS tranches because they assumed the risk of default was zero. In order for super senior tranches to default, the economy had to turn down completely from bottom to surface, which was very unlikely to happen. Under such assumption, banks treated their SS CDOs as fully hedged even if they were only partially hedged – usually by credit-default swaps. However, this method of hedging of hedging left the banks with counter-party risk from other financial institutions (Barnett-Hart 2009). It turned out that later it was these positions that caused the majority of bank's write-downs. As Merrill put it,

“The bottom line is that we got it wrong by being over-exposed to subprime. As the market for these (subprime) securities began to deteriorate, we began substantially reducing our warehouse risk by constructing CDOs and retaining the highest parts of the capital structure, which we expected then to be more resistant to market disruptions in terms of both liquidity and price...our hedging of the higher-rated tranches was not sufficiently aggressive nor was it fast enough.” (Barnett-Hart 2009)

In November 2008, Merrill Lynch, Citigroup, and Lehman Brothers, took write-downs of \$51.2, \$46.8, and \$15.3 billion, as the banks with the highest combined amounts of CDO and subprime assets. (Asset-Backed Alert, Nov. 18, 2008) The massive write-downs destroyed many of banks including Merrill and Lehman, pushing others to the brink of disaster. Furthermore, CDO losses have spread far beyond the investment banks on Wall Street, affecting very pool of investment money from pension funds to Norwegian villages. (House of Cards 2009) It remained still unsolved how much impact exactly investment banks had brought to the global economy.

## Rating Agency

The violent crash of the asset-backed (especially the subprime mortgage

backed) structured finance market was believed as one of the major catalysts for the 2008 financial crisis. Credit Rating Agencies have drawn much criticism for their role in fueling this unsustainable and problematic market.

Rating Agencies play a crucial role in financial markets. They have the responsibility to assign an investment grade to various debt-related financial instruments, i.e. bonds. Investors rely on these ratings to make their investment decisions. Due to the lower transparency and higher complexity in the structured finance market, investors have an even heavier reliance on rating agencies than any other market. Inflation of credit ratings easily boosted the market to grow dramatically in a short period of time and subsequent downgrades in ratings accelerated the collapse of the market. As many highly rated securities defaulted in the crisis and rating agencies had no choice but to downgrade them, it was clear that these agencies did not correctly place their rating at first place.

There were many reasons why these weathercocks made such huge miscalculation this time. The data used to develop ratings was different from what was available before. Traditionally, when rating agencies rated corporate debt, they based their ratings on publicly available, audited financial statements. In contrast, structured debt ratings were based on nonpublic, nonstandard, unaudited information supplied by the originator or issuer (Katz 2009). There could be potentially a lot of misrepresentation in the information, especially when the entire industry went insane in subprime mortgage business. Moreover, since rating agencies had no obligation to perform due diligence to check the accuracy of the information and the new mortgage-backed securities and CDOs were so complex to examine, rating agencies tended to and sometimes had to rely on representation and warranties provided by the originator or issuer. This largely undermined rating agencies' independence.

Rating agencies also fell behind updating their rating method to better fit the structured financial products. More than often, they lacked extensive historical data to make the correct distribution assumption of the innovative products. In order to rate anyways, they used older models that were created for traditional products. The

models turned out to be inadequate and inappropriate. For example, they failed to account for default correlation rise in the pool of assets in response to declines in housing prices. Rating agencies were also reluctant to invest in newer databases and rating models because they were costly and hurt profits.

At the root, there is always this conflicting interest in the nature of credit rating. Rating agencies run their business by charging securities issuers. The more securities they can get to rate, the more they get paid. If the issuers are consistently unsatisfied with the grade they get for their instruments, they will go to other agencies to get a better chance. This “issuer pays” business model encourages rating agencies to relax their own criteria in order to maintain or attract more market share. Moreover, a few large investment banks which controlled much of the deal flow made it even worse. They often “shopped around” for the highest ratings on their lucrative issuance deals by playing one rating agency against. Continuing pressing rating agencies, these banks often landed the privilege to consult agencies informally on structures they could create to achieve high ratings. The practice inevitably caused an inflation of credit rating and adversely hurt the industry standard.

Misrepresented information provided by issuers, complex structured finance products to rate, outdated rating models in use, and pressures to lower standard from peers and clients all led credit rating agencies to creating a rating bubble in the period of pre- crisis. According to a report issued by the BIS and Basel Committee’s joint forum (Report on securitization incentives), between 1990 and 2006, assets with the highest credit ratings rose from a little over 20 per cent of total rated fixed-income issues to almost 55 percent. It means more than half of the world’s debt securities were considered risk-free. The report also says during the same period, Asset-Backed Securities (ABS) accounted for 64 percent of the total growth in the amount of AAA-rated fixed income while public debt, corporate debt, and other debt contributed only 27, 2, and 8 percent to the total, respectively.

The bubble did not sustain for very long. It soon crashed as house prices went down and mortgages defaulted. As of 2007, triple-A rated CDOs were downgraded an

average of 16 rating points (1 means AAA, 22 means D) (Barnett-Hart 2009). It was shocking that as many as 70% of CDOs defaulted among all rated by some agencies. Numerous investors bore loss. Structured finance market became as volatile as ever. With its careless use of power to blow a credit rating bubble up and smash it in a flash, credit rating agencies helped to spill out the evil.

## **Global Investors**

It is never the heat itself that causes a fire. It is also the fuel. Global investors played the role of fuel in the crisis. Without their help, the crisis would never become so widely spread out and so profoundly influential. It might not have even happened.

Low interest rates not only encouraged spending instead of saving among consumers. It also turned investors to higher-yielding securities. A variety of investors ranging from professional investors and corporate CEOs to hospital funds and state pension funds began to look at loans to private equity deals. They were assured that these loans were safe as well as the mortgage pools. After all, when interest rates were low, the only way to earn more was to assume more risk.

Low interest was not limited to the United States. It was a worldwide phenomenon at the time. China, Japan, Germany, and oil-exporting nations were all experiencing the same issue. Unlike the U.S., they were spending less than their income and thus had extra money to lend. Thus they became the investors who purchased higher-yielding securities backed by private equity loans from the United States.

It was these investors' dollars that fueled the credit binge. In 2000s, when borrowing reached its peak, the United States alone sopped up 70 percent of the surplus capital flowing from the developing countries. It was largely argued that America was borrowing to spend only because other countries were lending. Although it sounded like America was blaming others for their own mistake, it was undeniable that global investors played their role in the development of the crisis.

## References

Angel, James J., Lawrence E. Harris, and Chester S. Spatt. "Equity Trading in the 21st Century." 23 Feb. 2010. Print. 25 Sept. 2011.

"Apex – Change Application-Level Default Templates « My Favorite Software Tools." *My Favorite Software Tools*. 31 Jan. 2008. Web. 29 Nov. 2011. <<http://stewstools.wordpress.com/2008/01/31/apex-change-application-level-default-templates/>>.

"APEX Check Boxes in Reports Regions." *Oracle Consulting, Oracle Support and Oracle Training ByBC Oracle Consulting*. Web. 03 Nov. 2011. <[http://www.dba-oracle.com/html/db/t\\_html/db\\_check\\_boxes\\_reports\\_regions.htm](http://www.dba-oracle.com/html/db/t_html/db_check_boxes_reports_regions.htm)>.

"APEX CSS Repository: How to Include Background Images Easily Using Data URIs." *Random Insights into Oracle*. 11 Sept. 2011. Web. 1 Dec. 2011. <<http://oracleinsights.blogspot.com/2011/09/apex-css-repository-how-to-include.html>>.

Ashcraft, B.Adam, and Til Schuermann. "Understanding the Securitization of Subprime Mortgage Credit." *Federal Reserve Bank of New York Staff Reports* no. 318. Print. Mar. 2008. Feb. 2012

Bank of China. *Underwriting of Structured Product Financing*. Retrieved February-17<sup>th</sup>, 2012  
[http://www.boc.cn/cbservice/cb2/cb24/200807/t20080710\\_817.html](http://www.boc.cn/cbservice/cb2/cb24/200807/t20080710_817.html)

Barnett-Hart, K. Anna. "The Story of the CDO Market Meltdown: An Empirical Analysis." Harvard College Cambridge, Massachusetts. Print. 19 Mar. 2009. Feb. 2012

Bigio, Saki, and Jennifer La'O. "The 2008 Financial Crisis: Institutional Facts, Data and Economic Research." Print. 29 Aug. 2011. Feb. 2012.

Black, Fischer and Myron Scholes. "The pricing of Options and Corporate Liabilities." *Journal of Political Economy*. Vol 81, No. 3 ( May – June., 1973) 637-654, Print.

BNP Paribas. "Enterprise Architecture Overview." Print. 24 Jan. 2011. Nov.2011.

BNP Paribas. "Enterprise Architecture First Phase – a tool for capacity management." Print. Nov. 2011.

Capone, P. Elisa. "Collateralized Debt Obligations(CDOs): An Introduction." *RGE monitor*. Print. 7 Mar.2007. Feb. 2012.

- "Cross-Platform Dynamic Graphics and GIS Solutions for Real-Time Data Display."  
*GLG Toolkit: High Performance Dynamic Graphics for C/C ,AJAX, Java and .NET.* Web. 03 Nov. 2011. <<http://www.genlogic.com/products.html>>.
- Dowd, Kevin. "Moral Hazard and The Financial Crisis." *Cato Journal*, Vol.29, No. 1. Cato Institute. Print. Winter. 2009. Feb. 2012
- Fabozzi, Frank J., Henry A.Davis and Moorad Choudhry. *Introduction to Structured Finance*. The United States of Amrica: John Wiley & Sons, Inc, 2006. Print.
- Fagan, Mark, and Frankel Tamar. "MBS, ABS, SPV, CDS, ARM, BBB+: Understanding the Alphabet Soup of Securitization." Copyright Fagan and Frankel. Print. Oct.2008. Feb.2012.
- Fender, Ingo, and John Kiff. "CDO rating methodology: Some thoughts on model risk and its implications." *Bank for International Settlements Working Papers No. 163*. Monetary and Economic Department. Print. Nov. 2004. Feb. 2012
- Gansner, Emden R., Eleftherios Koutsofios, and Stephen North. "Drawing Graphs with Dot." Web. 7 Nov. 2011. <<http://www.graphviz.org/pdf/dotguide.pdf>>.
- "GLG Toolkit: Graph Layout Java Demo." *GLG Toolkit: High Performance Dynamic Graphics for C/C ,AJAX, Java and .NET.* Web. 02 Nov. 2011. <[http://www.genlogic.com/java2/graph\\_layout.html](http://www.genlogic.com/java2/graph_layout.html)>.
- Gonzalez, Carlos. "Why Securitization?" Stewart title Latin America. Print. Feb. 2012.
- "Graphviz - QED." *Main Page - QED*. Web. 09 Nov. 2011. <<http://qed.princeton.edu/main/Graphviz>>.
- "House of Cards" CNBC. 12 Feb. 2009. Media. Feb. 2012
- Jobst, Andreas. "What Is Securitization?" *Financial & Development*. Print. Sept. 2008. Feb. 2012
- Katz, Jonathan, Emanuel Salinas, and Constantinos Stephanou. "Credit Rating Agencies." *Crisis response*. Print. Oct. 2009. Feb. 2012
- Library of Economics and Liberty. *Takeovers and Leveraged Buyouts*. Retrieved Feb-23<sup>rd</sup> ,2012  
<http://www.econlib.org/library/Enc1/TakeoversandLeveragedBuyouts.html>
- "Linking – SVG 1.1 (Second Edition)." *World Wide Web Consortium (W3C)*. Web. 07 Dec. 2011. <<http://www.w3.org/TR/SVG/linking.html>>.

Lowenstein, Roger. *The End of Wall Street*. New York: Penguin, 2010. Print. Feb. 2012

"Manipulating Database Objects Using Oracle Application Express 4.1." *Oracle / Hardware and Software, Engineered to Work Together*. Web. 01 Nov. 2011. <[http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/apex/r41/apexstart\\_a/apexstart\\_a.htm](http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/apex/r41/apexstart_a/apexstart_a.htm)>.

"Neuron Diagram for .NET." *Neuron Web Examples*. Web. 02 Nov. 2011. <<http://examplesaspnetdiagram.nevron.com/>>.

*Neuron .NET Vision - Online Documentation*. Web. 02 Nov. 2011. <<http://helpdotnetvision.nevron.com/>>.

"Node, Edge and Graph Attributes." *Home / Graphviz - Graph Visualization Software*. Web. 09 Nov. 2011. <<http://www.graphviz.org/doc/info/attrs.html>>.

"Node Shapes." *Home / Graphviz - Graph Visualization Software*. Web. 15 Nov. 2011. <<http://www.graphviz.org/doc/info/shapes.html>>.

O’Kane, Dominic and Stuart Turnbull. “*Valuation of Credit Default Swaps*.” Lehman Brothers, Quantitative Credit Research April 2003: Print.

"Oracle Application Express 4.1." Web. 31 Oct. 2011. <<http://apex.oracle.com/i/index.html>>.

"Oracle Application Express APEX: Calling Reusable Pages in Separate Applications." *Oracle Consulting, Oracle Support and Oracle Training ByBC Oracle Consulting*. Web. 14 Nov. 2011. <[http://www.dba-oracle.com/html/db/t\\_calling\\_pages\\_between\\_applications.htm](http://www.dba-oracle.com/html/db/t_calling_pages_between_applications.htm)>.

"Oracle Application Express APEX: Passing Values between Screens." *Oracle Consulting, Oracle Support and Oracle Training ByBC Oracle Consulting*. Web. 09 Nov. 2011. <[http://www.dba-oracle.com/html/db/t\\_passing\\_values\\_between\\_screens.htm](http://www.dba-oracle.com/html/db/t_passing_values_between_screens.htm)>

"Oracle Application Express." *Oracle / Hardware and Software, Engineered to Work Together*. Web. 31 Oct. 2011. <<http://www.oracle.com/technetwork/developer-tools/apex/overview/index.html>>.

"OTN Discussion Forums : Change Colour of Row - Oracle APEX SQL Report." *OTN Discussion Forums*. Web. 08 Dec. 2011. <<https://forums.oracle.com/forums/thread.jspa?threadID=883110>>.



- "Python: Lambda Functions." Web. 28 Oct. 2011.  
<[http://www.secnetix.de/olli/Python/lambda\\_functions.hawk](http://www.secnetix.de/olli/Python/lambda_functions.hawk)>.
- "Report on securitization incentives". *Joint Forum*. Web. Feb. 2012.  
<http://www.bis.org/press/p110713.html>
- "Scripting – SVG 1.1 (Second Edition)." *World Wide Web Consortium (W3C)*. Web. 07 Dec. 2011. <<http://www.w3.org/TR/SVG/script.html>>.
- Song, Shin Hyun. "Securitization and Financial Stability." *The Economic Journal*, 119 (March), 309-332. Royal Economic Society. Print. 2009. Feb. 2012
- "SQL INNER JOIN Keyword." *W3Schools Online Web Tutorials*. Web. 26 Oct. 2011.  
<[http://www.w3schools.com/sql/sql\\_join\\_inner.asp](http://www.w3schools.com/sql/sql_join_inner.asp)>.
- Stewart, Frances. "The 2008-9 crisis and developing countries: implications for poverty." Print. Feb. 2012
- Stubbs, Paul. "Office Add-Ins: Develop Add-Ins For PowerPoint And Visio Using VSTO." *MSDN / Explore Windows, Web, Cloud, and Windows Phone Software Development*. Feb. 2007. Web. 03 Oct. 2011.  
<<http://msdn.microsoft.com/en-us/magazine/cc163471.aspx>>.
- "Turn Report Rows into a Multi Column Format through Templates!" *APEX Development*. Web. 08 Dec. 2011.  
<<http://application-express-blog.e-dba.com/?p=55>>.
- "URL Parameters." *Oracle Application Express*. 20 Nov. 2009. Web. 09 Nov. 2011.  
<<http://www.oracleapplicationexpress.com/tutorials/55>>.
- "Using GraphViz, a Brief Tutorial | Orient Lodge." *Orient Lodge | An Eclectic News Site*. Web. 26 Oct. 2011. <<http://www.orient-lodge.com/node/3408>>.
- Vardi, Moshe Y., Ian Barland, and Ben McMahan. "Logic and Database Queries." Rice University, 31 Aug. 2006. Web. 4 Oct. 2011.  
<<http://www.cs.rice.edu/~tlogic/Database/all-lectures.pdf>>.
- "Visio 2010 Add-Ins Overview." *Microsoft Vision 2010*. Microsoft Corporation, 2009. Web. 3 Oct. 2011.  
<<http://visiotoolbox.com/2010/PDFs/visio-2010-add-ins-overview.pdf>>.
- Wang, Xiaoyun, and Tu Lai Huan. "BNP Paribas: Equity Smart Order Router." *Electionic Projects Collection*. Worcester Polytechnic Institute, 29 Nov. 2010. Web. 4 Oct. 2011.  
<[http://www.wpi.edu/Pubs/E-project/Available/E-project-112910-121011/unrestricted/Equities\\_Smart\\_Order\\_Router.pdf](http://www.wpi.edu/Pubs/E-project/Available/E-project-112910-121011/unrestricted/Equities_Smart_Order_Router.pdf)>.

Wolf, Patrick. "Opening URL in a New Window." *Inside Oracle APEX by Patrick Wolf*. Web. 28 Nov. 2011.  
<<http://www.inside-oracle-apex.com/opening-url-in-new-window/>>.

Wolf, Patrick. "Oracle APEX 4.0: Cascading LOVs/Select Lists." *Inside Oracle APEX by Patrick Wolf*. Web. 18 Nov. 2011.  
<<http://www.inside-oracle-apex.com/oracle-apex-4-0-cascading-lovselect-lists/>>.

"2008 Financial Crisis." *Concept*. Web. 29 Feb. 2012.  
<[http://www.wikinvest.com/concept/2008\\_Financial\\_Crisis](http://www.wikinvest.com/concept/2008_Financial_Crisis)>.