Financial Ratio Analysis: the Development of a Dedicated Management Information System

Voicu-Dan Dragomir
Asistent universitar doctorand
Academia de Studii Economice București

Abstract. This paper disseminates the results of the development process for a financial analysis information system. The system has been subject to conceptual design using the Unified Modeling Language (UML) and has been implemented in object-oriented manner using the Visual Basic .NET 2003 programming language. The classic financial analysis literature is focused on the chain-substitution method of computing the prior-year to current-year variation of linked financial ratios. We have applied this technique on the DuPont System of analysis concerning the Return on Equity ratio, by designing several structural UML diagrams depicting the breakdown and analysis of each financial ratio involved. The resulting computer application offers a flexible approach to the analytical tools: the user is required to introduce the raw data and the system provides both table-style and charted information on the output of computation. User-friendliness is also a key feature of this particular financial analysis application.

Key words: financial ratio analysis; object-oriented development; unified modeling language; conceptual models.

1. Introduction

Financial statements are intended to provide information on the resources available to management, how these resources were financed, and what the firm accomplished with them. Corporate shareholder’s annual and quarterly reports include three required financial statements: the balance sheet, the income statement and the statement of cash flows. Information from the basic financial statements can be used to determine what factors influence a firm’s earnings, cash flows and risk characteristics.

Accounting ratios identify irregularities, anomalies and surprises that require further investigation to ascertain the current and future financial standing of a company. Financial ratios describe the relationships between different items in the financial statements. In order to achieve the desired result, that is the useful information each ratio can convey, we should only compare the company’s financial ratios with ratios for a preceding period and budgeted ratios for the current period. In each case, comparison is possible if an identical basis of compilation is used. There must be conformity and uniformity in the preparation of accounts to ensure a comparison of like with like.

From the computer developer’s point of view, the hundreds of financial ratios that can be computed are a true bottle-neck. The analyst’s tool-pack should not only be standardized, but it should also be subject to a unified treatment from a conceptual perspective. Object-orientation provides an elegant language for framing such problems, and powerful tools for resolving them.

In this paper we will give the reader an in-depth analysis of variation-based methods of financial analysis, followed by the detailed presentation of the unified modeling process applied to financial statement analysis. Considering that object-orientation and the Unified Modeling Language (UML) are practically synonymous, the reader
is presented with a brief overview of the UML and its practical implications on the subject matter. The last two major chapters of this paper offer insights on the structural models underlying the system, followed by a graphical introduction to the software itself.

2. The employment of the chain substitution – ceteris paribus method of financial ratio analysis

For financial ratios that adhere to a deterministic scheme (that includes algebraic operations like multiplication and division), the following steps can be followed in order to achieve a higher level of analytical detail:

a) The specification of the primary form of financial ratios. Considering the computation of Return on Owner’s Equity (ROE), this return would equal:

\[
\text{Return on Equity} = \frac{\text{Net Income}}{\text{Average Total Equity}}
\]

b) The breakdown of the primary form into component ratios or financial elements. In the case of ROE, the application of the DuPont System of financial analysis would generate the following identity:

\[
\text{ROE} = \frac{\text{Net Income}}{\text{Average Total Equity}} = \frac{\text{Net Income}}{\text{Net Sales}} \times \frac{\text{Net Sales}}{\text{Common Equity}}
\]

The breakdown is an identity, because we have both multiplied and divided by net sales. To maintain this identity, the common equity value used is the year-end figure, rather than the average of the beginning and ending value. This identity reveals that ROE equals the net profit margin times the equity turnover, which implies that a firm can improve its return on equity by either using its equity more efficiently (increasing its equity turnover) or by becoming more profitable (increasing its net profit margin).

Moving forward, a firm’s equity turnover is affected by its capital structure. Specifically, a firm can increase its equity turnover by employing a higher proportion of debt capital. We can see that effect by considering the following relationship:

\[
\text{Equity Turnover} = \frac{\text{Net Sales}}{\text{Equity}} = \frac{\text{Net Sales}}{\text{Total Assets}} \times \frac{\text{Total Assets}}{\text{Equity}}
\]

This equation indicates that the equity turnover ratio equals the firm’s total assets turnover (a measure of efficiency) times the ratio of total assets to equity (a measure of financial leverage). This financial leverage ratio is also known as the financial leverage multiplier, whereby the first two ratios (profit margin and total assets turnover) equal return on total assets (ROTA), and ROTA times the financial leverage multiplier equals ROE.

Combining the two breakdowns, we see that a firm’s ROE is composed of three ratios as follows:

\[
\text{ROE} = \frac{\text{Net Income}}{\text{Net Sales}} \times \frac{\text{Net Sales}}{\text{Total Assets}} \times \frac{\text{Total Assets}}{\text{Common Equity}}
\]

c) The computation of every component’s influence on the primary ratio. This step is accomplished through chain substitution of prior-year with current-year values for a specific ratio or financial figure, considering all other terms fixed (ceteris paribus). If we consider the following notations: \( \Delta \) (delta) stands for the particular influence of a given ratio, the subscript 1 (one) for the current-year value and the subscript 0 (zero) for the prior-year value, we would compute the components’ detailed influence on ROE using the algorithm:

\[
\Delta \text{ROE} = \frac{(\text{Net income})_1 - (\text{Net income})_0}{(\text{Average Total Equity})_1} - \frac{(\text{Net income})_0}{(\text{Average Total Equity})_0}
\]

\[
\Delta (\text{Net Profit Margin}) = \frac{(\text{Net Income})_1 - (\text{Net Income})_0}{(\text{Net Sales})_1 - (\text{Net Sales})_0} \times \frac{(\text{Net Sales})_0}{(\text{Total Assets})_0} \times \frac{(\text{Total Assets})_1}{(\text{Common Equity})_0}
\]

\[
\Delta (\text{Total Assets Turnover}) = \frac{(\text{Net Income})_1}{(\text{Net Sales})_1} \times \left( \frac{(\text{Net Sales})_1 - (\text{Net Sales})_0}{(\text{Total Assets})_1 - (\text{Total Assets})_0} \right) \times \frac{(\text{Total Assets})_0}{(\text{Common Equity})_0}
\]

\[
\Delta (\text{Leverage Multiplier}) = \frac{(\text{Net Income})_1}{(\text{Net Sales})_1} \times \frac{(\text{Net Sales})_1}{(\text{Total Assets})_1} \times \left( \frac{(\text{Total Assets})_1 - (\text{Total Assets})_0}{(\text{Common Equity})_1 - (\text{Common Equity})_0} \right)
\]

The deterministic relationship between the original financial ratios is transformed into a stochastic relationship between the variations of these ratios. The individual influences sum up to the amount of variation of the primary ratio:

\[
\Delta \text{ROE} = \Delta (\text{Net Profit Margin}) + \Delta (\text{Total Assets Turnover}) + \Delta (\text{Financial Leverage Multiplier})
\]
This analytical approach offers significant insight into a compound financial ratio’s variation mechanics. However, a particular breakdown of a financial ratio is not unique. Specifically, there are many other applicable breakdowns which meet the needs of one analyst or another, depending on the variations to be computed.

3. The Unified Modelling Language and object-orientation: a brief discussion

Having considered the theoretical implications of chain substitution in the field of financial analysis, this chapter is dedicated to present the process of modelling the financial ratios using the Unified Modeling Language (UML 2.0). This process ensures that the maximum flexibility of the system is attained by considering the latest software modelling paradigm – the object-oriented conceptual framework.

A general-purpose language such as the UML may be applied throughout the system-development process all the way from requirement gathering to implementation of the system. As a broadly applicable language, UML may also be applied to different types of systems, domains, and processes. Therefore, we can use the UML to communicate about software systems and non-software systems (often known as business systems) in various domains or industries such as manufacturing, banking, e-business, and so forth. Furthermore, we can apply the UML to any process or approach. It is supported by various tool vendors, for it is not a proprietary or closed modeling language (Alhir, 2003).

A language is generally based on a paradigm, a way of viewing a subject, which defines the types of concepts that may be used in the language and the principles of why they are useful. A language’s syntax specifies the notation used for communication and is determined by the language’s alphabet. A language’s semantics specify the meaning that is communicated and is determined by the language’s words and sentences. The syntax of the UML involves diagrams, and its semantics are based on the object-oriented paradigm. Therefore, object-orientation (OO) is a powerful design methodology, which has firmly moved into the mainstream of software development (Roussev et al., 2006).

Object-orientation, visually supported by the UML, is expected to give significant results on several key aspects of system development:

- **Strong integration of data and processes**: during object-oriented development, data and processes are kept together in small, easy-to-manage packages. Thus data is never separated from the algorithms. Considering the field of financial analysis we end up with both the computing mechanism of financial ratios and the data itself set up as a stand-alone module, an object as it is called, that exists independently throughout the system’s execution;

- **Intense reusability**: with object-oriented development, we are constantly looking for objects that would be useful in similar systems. Consequently, once an object acquires a conceptual form, it is plausible that it will be included in several different models, with minor changes or no change at all. The financial ratio analysis models can be reused in the context of information systems that serve both the managerial and the investor’s approach to the firm’s profitability. Therefore, business objects like the financial ratios are context-independent and highly reusable, eventually being a stable and reliable source of information for every class of users as defined by the IASB Framework.

- **Dynamic approach**: by applying an iterative approach, any subsets of the lifecycle activities are performed several times to better understand the requirements and gradually develop a more robust system. Each cycle through these activities or a subset of these activities is known as iteration, and a series of iterations in a step-wise manner eventually results in the final system. Particularly, not only that existing sets of financial ratios can be modified and extended, but brand new functionalities can be developed based on the ever expanding requirements of the financial users.

4. Structural modeling of the financial analytical mechanism

This chapter focuses on the class and object diagrams, which depict the structure of the financial ratio system in general and at a particular point in time, respectively. As an architecture-centric process focuses on the architecture of a system across iterations, it is important to understand what elements make up a system and how they are related to one another. That is why we usually apply class and object modeling during analysis and design activities to understand the requirements and determine how a system will satisfy its requirements.

A **class** defines a type of object and its characteristics, including structural features and behavioral features. In a UML class diagram, a class is shown as a solid-outline rectangle with three standard compartments separated by horizontal lines. The required top compartment shows the class name, the optional second compartment shows a list of attributes (the structural features), and the optional third compartment shows a list of operations (the behavioral features).

4.1. The **Item** class

The **Item** class (Figure 1) is considered to be the fundamental class of the system, mainly because it encompasses the largest mass of information and behavioral characteristics. An item may be a financial element (i.e. stocks), a financial highlight (i.e. gross margin) or a ratio (i.e. net profit margin).
An attribute is what an object of a class knows, a piece of data maintained by the object. The Symbol attribute ensures a unitary notation throughout the model database, thus the Symbol serves to pinpoint the identity of a particular financial element.

The most crucial aspect of an object is that it has its own identity. No two objects are the same, even if they have the same values for their structural features. For example, even if two financial ratio objects have the same values for their Symbol and Name attributes, the instances of the Item class are unique and have their own identities.

The Rank of an item signals the order in which every element is introduced into the model. A specific instance of a class, with its structural and behavioral characteristics, is described by the object diagram. Consider the following object diagram for the DuPont System, namely the breakdown of the Return on Equity ratio:

The Predecessor attribute is a necessary backward view of the Rank attribute. For example, the predecessor of Item.Rank = 3 is Item.Rank = 0.

On the basis of the Yes/No value provided for the HasValue attribute, the software asks the user for the amount of a particular financial element (e.g. the Item.Symbol = NetSales should be provided with a value extracted from the income statement, whereas the Item.Symbol = ROE computes the values internally, so there is no need for the user to provide values for this ratio).

The Sign attribute can take the +/- values, whether the financial element enters an addition or a subtraction.

4.2. The Item Collection class

A collection is a set of similarly typed objects that are grouped together. Objects of any type can be grouped into a single collection of a generic type Object to take advantage of constructs that are inherent in the UML. The Item Collection attribute of an Item object is an enumeration of other Items seen as hierarchically inferior to the main Item. An Item Collection is also an object that belongs to an Item and is therefore treated as an independent entity with well-defined attributes and methods.

Consider the following structural diagram:
The class diagram presented in Figure 3 introduces the recursive relationships between instances of the Item class. Associations, like Contains and Enumerates, represent conceptual relationships between instances of classes. The multiplicities of these associations, which are an indication of how many objects may participate in the given relationship, state the following:

- An Item contains a single Item Collection, which in turn is a unique entity belonging to a single Item.
- An Item Collection is an enumerator of multiple Items, possibly none. An enumerator flattens a collection so that the members can be accessed sequentially. Different collection classes might have different sequences. This suggests that the Items in a Collection are not added at random.

The Contains association is an aggregation, that is a “part-of” relationship. An Item, even though it may belong to a Collection, it is still a fairly independent entity, whereas a Collection cannot exist in the absence of a specified Item to whom it is bound to belong.

The Add and Remove methods are self-explanatory. Individual Items are instantiated throughout the program’s execution. They are added one by one to the corresponding Collections, in order to take part in financial computations, while retaining their previously defined attributes.

One delicate aspect of the conceptual model of an Item Collection is its Operation attribute, which refers to the algebraic operation imposed to the Items belonging to a Collection. The following object diagram provides an insight into the mechanism of computing the financial ratios:

![Object diagram for the Item - Item Collection associations for the DuPont System](image)

Starting from the Return on Equity ratio, we follow the branches of the object tree up to the atomic elements that are directly extracted from the balance sheet or the income statement. The object diagram in Figure 4 is sufficiently explicit: ROE ratio is the result of the breakdown and multiplication of three ratios: net profit margin, total assets turnover and financial leverage, respectively. Aside from the two operations considered in the example above, financial elements and ratios can be as well added or subtracted from each other; thus the Operation attribute of the Item Collection class will be defined accordingly.

4.3. The Financial Ratio class

A Financial Ratio (Figure 5) gathers under one conceptual model the Items and their Collections plus the mechanism of computing the period-to-period variations.

![A holistic approach to the structural model of the system](image)
The Financial Ratio class contains the complete Item Collection, in addition to other descriptive attributes, like the model's name and description, the associated chart objects and two data tables, containing the input figures and the computed percentage changes in the financial element. For example, the ROE Financial Ratio computes the year-to-year variation in the ROE Item, plus the percentage change in ROE and the line chart based on the variation.

5. Financial Ratio Analysis – The Implementation

The system had been implemented using the Visual Basic .NET programming environment, which has been endowed with strong object-oriented features, similar to the one's originally owned by C++ and Java.

The system’s functionalities are visually described by the following interface flow diagram:

Whenever the main window of the application is launched:
- The user is asked to browse the model base, for which he is provided model description and the breakdown structure of each financial ratio;
- The system launches the Data Fill-In window, in which the user is required to provide the significant data for the financial elements included in the model;
- The system launches two separate windows, each containing financial information computed from the raw data entered by the user;
- The Year-to-Year Variation window reads the table created on the analytical technique described in Chapter 2. It also offers the possibility to display a custom chart relying on the table data;
- The Percentage Change window shows the percentage raise or fall of the significant elements included in the model. The user can individually select each financial element within the table in order to produce a custom chart in a special window;
- Every piece of information is summed up in the final report, which is to be reviewed, and then printed.

5.1. The model list & details window

The introductory window of the application is designed as follows:
- The left-hand side presents the model list region in the form of a tree view. The root sections follow the guidelines of any comprehensive financial analysis (internal liquidity, profitability, risk assessment etc.), while the two secondary branches exhibit the name of the model (financial ratio) and a brief abstract;
- The upper-right area of the window presents the financial formula upon which the ratio is broken down and variations computed;
- The lower-right list view is filled with the detailed information of the component ratios or financial element of the selected model. A symbol (see Chapter 4.1) is attached to every element, exposing the Symbol attribute of the Item class.
5.2. The Data Fill-In window

In Chapter 2 the reader noticed that the broke down ratios are computed from financial elements extracted from the balance sheet and the income statement. Even though these elements may be found more than one time in a financial model, the user is required to produce their values only once throughout the entire execution of the application. The prior-year and current-year values identified by the element’s symbol are stored in the computer memory, and accessed every time calculations are being done no matter what financial ratio is in use. Therefore the user can work on several models at the same time, while the system automatically transfers the data from one window to another.

5.3. The Computed Variations window and chart

The table presented in Figure 9 is a synthetical view of the algorithm presented in Chapter 2. The figures on the Variation column add up to the amount spelled as “overall variation” of the main ratio, that is, in our case, Return on Equity (ROE). By clicking on the Create Chart button, the system generates the variation histogram, which can be later added to the final report.
5.4. The Percentage Changes window and customized charts

The percentage changes table has proven to be a very useful analytical tool. The underlying table has been extended by customized graphs, both line and bar typed, which are a suitable visual complement to the figures presented in the main table. The user is able to select any financial element to be represented graphically, provided that the scale of the associated values are comparable – a chart showing values under 1 and over 1 million isn’t visually significant.

Figure 10. The percentage changes table and the associated charts

6. Conclusion

Financial ratio analysis is an integral part to the assessment and improvement of company performance. Financial ratios help to direct attention to the areas of the business that need additional analysis. Ratios are very useful when they are used properly and also in conjunction with many other sources of information. Complementary financial ratios can be developed to take into account the share price at closure date or quarterly. The analyst can also compare ratios across a panel of companies in the same industry or against the industry sector averages. Further improvement can be brought to the presented system, undoubtedly. Still, we believe that the conceptual framework developed for the financial ratios, taking advantage of the flexibility and strengths of the object-oriented paradigm, can be considered a reliable starting point.

References

*** Professional ADO.NET Programming, Wrox, 2002
*** Visual Basic .NET Complete, Sybex, London 2002