Novel Analysis Patterns in the Context of the Managed Investments Instruments

Luigi Ubezio[1], Claudia Raibulet[1], and Antonio Carpinato[2]

[1] Università Milano Bicocca, DISCo, Via Bicocca degli Arcimboldi 8, Edificio U7, 20126 Milan, Italy
{ubezio, raibulet}@disco.unimib.it
http://www.unimib.it
[2] Caridata Engineering, Strada 2 – Palazzo D/3, 20090 Assago Milanofiori (MI), Italy
{antonio.carpinato}@eng.it
http://www.caridata.it

Abstract. Traditionally, the investment funds market exploits analysis and design concepts based on the procedural programming approach. We propose a set of analysis patterns which describes the financial products using the object-oriented paradigm. Primarily, the goal of our project has been to provide a solution to calculate the limits related of the managed financial investments. Thus the analysis of financial products has been performed from the calculus of limits point of view. The result is a set of design guidelines that include not only the definition of limits and their calculus formulas but also the definition of products, domains, and portfolios. Our aim has been to provide a model that can be further extended with new types of limits and calculus formulas and to be reused in other financial applications. The paper ends by summarizing the most important implementation issues we have addressed.

Keywords: analysis patterns, finance, investment funds, portfolio management.

1 Introduction

The industry of investment fund and portfolio management has grown over the last decade becoming a key actor in the European capital markets. According to [8], the European fund industry manages over €5 trillion of assets.

Traditionally, the investment fund market exploits designing and programming techniques based on procedural approaches like COBOL/CICS[1]. Nevertheless, it is turning towards new paradigms of developing software, especially after the spread of the Internet and the related Web applications. Hybrid solutions may co-exist, the so

[1] Short for Customer Information Control System, a TP monitor from IBM that was originally developed to provide transactions processing for IBM mainframes. It controls the interaction between applications and users and lets programmers develop screen displays without detailed knowledge of the terminals being used. CICS is also available on non-mainframe platforms including the RS/6000, AS/400 and OS/2 -based PCs.

called web enabled applications: functionalities of legacy systems are simply exposed via a Web graphical interface, without significant modifications in the business logic or the database structure [6], [17], [18] and facing security problems [21]. However, such solutions do not allow for the full exploitation of the new paradigms, being an intermediate, temporal adaptation to the today's requirements. Furthermore, frequently enough, practical considerations such as those related to database obsolescence, lead to the necessity of redesigning and re-implementing the entire application. In this context, the object-oriented paradigm seems to be a valid alternative to the traditional approach, while Java and the J2EE platform a valid implementation support. Such a radical change raises new challenges which may be both technical and organizational, staff skills included.

The contribution of this study may be conceptually identified with the suggestive term of pattern, or, more precisely, of patterns kit, in the context of the investment instruments management. We follow the operative definition of Martin Fowler: "a pattern is an idea that has been useful in one practical context and will probably be useful in others" [11]. Being particularly related to the identification of business processes' conceptual structure, these patterns are more precisely defined as "analysis pattern". We are aware of patterns' presentation in general, and this attempt in particular, cannot have a dogmatic characteristic, but could become an opportunity to share and compare architectural approaches. Being a result of an inductive work, patterns should be mainly considered documentary crystallizations of possible or already adopted solutions and they are finalized by themselves, to deeper studies and discussions.

The work presented in this paper is the result of an actual project developed under the supervision of the Caridata Engineering company for an Italian fund management society. The goals of this paper are to highlight and, if possible, to solve common problems related to the investment instruments management; to open a debate on various possible implementations; to propose a solution path; and to apply and validate the analysis pattern concept in this context.

To achieve our goals we provide a system's functional analysis with architectural considerations, introducing the financial specific products, their composition in portfolios, limits, and domains. This paper focuses on the calculus of limits, which are the expressions identifying the main features a financial portfolio instrument must keep unchanged over time, even in the presence of every day changes in its detailed financial composition. These limits define the main characteristics of the product itself.

We present our work following an iterative and incremental approach by providing one or more solutions to same problem starting from the simplest one and increasing its complexity as performed during the analysis steps.

The rest of the paper is organized as following. Section 2 provides an overview on the terms specific to the financial market. The financial products, portfolios, and domains are presented in Section 3. The description of portfolio limits is provided in Section 4. The details related to the calculus of limits in our case study are described in Section 5. Section 6 presents the main implementation hints. Conclusions and further work are dealt in Section 7.

2 Background

The aim of this section is to provide a brief overview on the managed investment funds and portfolios. Common investment funds allow a group of investors to bring together their funds and to invest them properly. By pooling their funds together, investors can sample a broader range of stocks or bonds that they could address if they were acting on their own.

A fund manager combines clients' money with those of other investors. Taken altogether, those investments are called *fund's assets*. The fund manager invests the fund's assets, typically by buying stocks, bonds, or a combination of the two. Some funds may buy more complicated *security types*. These stocks or bonds are often referred to as fund's *holdings*, and all fund's holdings together form a *portfolio*. The purpose of assets' allocation is to reduce risks by diversifying the portfolio. A fund's type depends on the kinds of securities it holds. For example, a stock fund invests in stocks, while a small-company stock fund focuses on the stocks of small companies. What you get as an investor or shareholder is a portion of that portfolio. Regardless of how much you invest, your shares represent a portfolio.

Funds offer some significant benefits to investors: do not demand large up-front investments; are easy to buy and sell; are regulated; and are professionally managed.

The individual or private portfolio management is characterized by the fact that managed portfolios are clients' oriented meaning that the investor should provide a major capital. However, the investment risk rises because an individual portfolio cannot be divided in quotes that can be sent independently to other investors. Furthermore, the range of the investment titles on which individual portfolios can invest is less wide than that of the collective investors. The advantage of the individual portfolios is provided by the possibility of defining personalized investments and of exploiting a particular management strategy for a portfolio. The disadvantage regards the risks an individual investor should consider.

Regardless the investment type, collective or individual, limits are defined to verify the goals and the features of a product. Typically, these controls are stated by specialized agencies such as banks or national control agencies, or they are part of the objectives of the manager, or they are explicitly required by the investor. Usually, who performs such controls deals both with individual and collective funds requiring a unified vision of the two.

Due to the importance of the funds market in the context of the European reality, there is a great interest and a significant related work to unify the legislation of the investment instruments and their process rules [8]. The goal of this effort is to "identify ways to facilitate the successful development of the fund industry in the short to medium term by building on existing legislation while at the same time guaranteeing the necessary high level of investor protection" [7]. For example, the SWIFT[2] society works on unifying the communication investment messages exploited among investors, investment funds companies, and banks [19], [20].

To better understand the notions presented in this section, see [1], [2], [3],[15],[16].

---

[2] SWIFT is a short for Society for Worldwide Interbank Financial Telecommunication.

3 Products and Composition of Portfolio

The objects of our analysis are those financial instruments which can be sold as products to consumers. The aim of this section is to describe the structure of the financial instruments. Such products may be divided in two categories depending on the portfolio management: collective and individual. Considering their common characteristics, individually managed products are grouped into Lines (see Fig. 1). Their management provides a certain degree of operative flexibility to the buyer. Besides choosing the product line that suits him/her best (considering risks, amount, etc.), the investor may require further constraints such as the type of the companies s/he is interested in (excluding for instance the army industry) or the type of actions of a given typology always present in her/his portfolio.

For each product, the management organization performs daily a set of purchase and sale operations to improve its economic performance. Such operations, which modify the composition of portfolio, use financial instruments not immediately usable without a high risk of capital loss by an ordinary investor. A first version of the structure of products and portfolios is shown in Fig. 1.

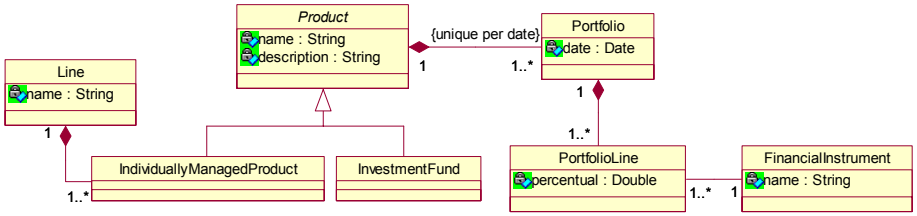

Fig. 1. A First Version of the Structure of Financial Products

Usually, the so called passive management of each instrument such as the numeric consistence of its quota and its real distribution among customers considering different emissions could be represented, as well as the information related to the composition of the financial instruments (owner, number of shares, date of emission, etc.). In the context of this paper, the representation and management of this information is not relevant for the design of products and portfolios.

The Product class is abstract because it must be further specialized as shown in Fig. 1. Note that a Line is merely a container for Individually Managed Products. A Portfolio Line represents an element of a Portfolio.

3.1 Design Convergence and Functional Diversity

If the Investment Fund class does not inherit from Product meaning that the Portfolio of a Product does not contain or cannot contain any Product, the representation is valid. But, this does not model the real case because a Portfolio of an Individually Managed Product may contain Investment Funds or it is advisable to contain those Products for which the Portfolio is analyzed being part of the financial offering of the same manager. Thus, Investment Funds are real financial instruments and they should

belong to a Portfolio. From the analysis point of view not considering the relation between Financial Instruments and Products affects the flexibility of the model. Actually, both Financial Instruments and Products can belong to a Portfolio. The difference is that not for all Financial Instruments the Portfolio consistence is checked. And this is because the portfolio of a Financial Instrument is not always of the competence of the fund manager. These objects belong to the same hierarchy, even if they are exploited in a different way. To express this similarity, the class diagram has been changed as in Fig. 2. The Financial Instrument is considered a Product.

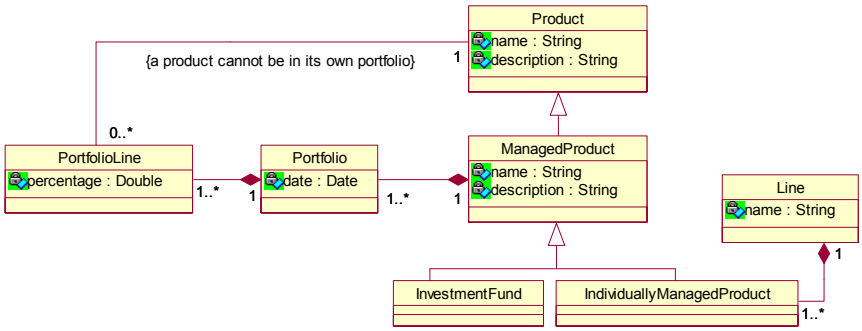

Fig. 2. The Managed Product and the Portfolio

In Fig. 2, Product is no more abstract. This allows the instantiation of all those financial instruments which do not belong to the managed funds. From an ontological point of view, we should further consider separately products containing portfolios and products without portfolios. From our point of view which is a functional one, we are interested in products on which we can analyse their related portfolios. Obviously, if a product is analysed than it has associated a portfolio, but the opposite is not true. To address this aspect, the Managed Products has been introduced to denote those products which have associated a portfolio. The recursion problem of having a Product inside its own Portfolio has been solved by introducing a constraint.

Another important aspect is related to the concept of Domain. This concept is directly connected to the features of Products belonging to a Portfolio. Each Product of a Portfolio is classifiable upon information such as the emission currency, the nation or the related geographic area, and so on (see Fig. 3).

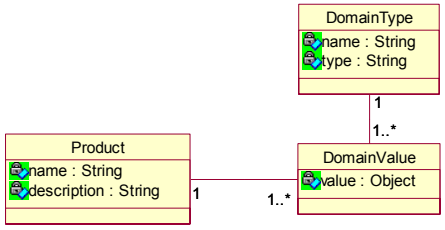

Fig. 3. Products Domain

The representation in this case is very simple: a Product has one or more pertinent domains (Domain Type class), each of them with its proper value (Domain Value class). The Domain Value is simply typed by the *type* attribute according to its Domain Type: in practice it can be a string, a number, or a percent.

3.2 Exploiting Metalevels to Reduce Flexibility Degrees

In the second version of our class diagram, each Product can be inserted in the Portfolio of another Product. This allows an excessive degree of flexibility because it is possible to insert an Individually Managed Product into a Portfolio, but because of its nature an Individually Managed Product cannot be partitioned in shares and sold to different people; hence it cannot be inserted in a Portfolio. This relation should be ruled by a metalevel. The aim is not only to know if a Product may belong to a Portfolio, but also to know which Product type may belong to a Portfolio type. This is achieved by inserting the Portfolio Type and Product Type in a metalevel (see Fig. 4).

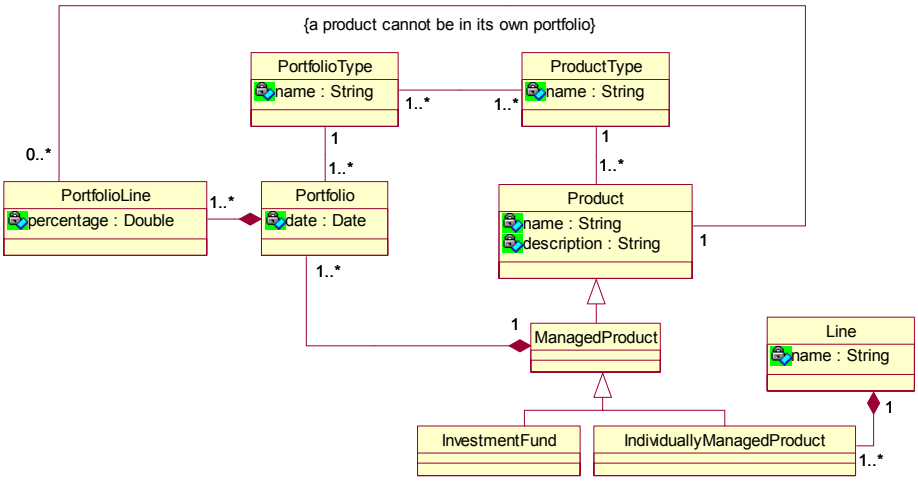

Fig. 4. Portfolio Type and Product Type

Introducing the metaclasses the diagram gains more expressivity than required: there are both Portfolio and Product typologies and every product type may belong to a portfolio type, primarily we have introduced only the Product Type class, considering that Portfolios are of the same type. But the diversity of Portfolios cannot be excluded a priori. The Product subclassing is not required to be equivalent to the Product Type one. Hence, Product Type may have another, even transversal, product classification based on the possibility of belonging or not to a Portfolio.

4 The Portfolio Limits

During the analysis phase, an important issue we have addressed is the definition of portfolio limits. Attention has been focused on the flexibility and reusability of the limits' definitions. Typically, limits are checked through an operation that verifies a set of features defining the portfolio consistence (e.g., it is possible to check whether shares represent 80% of a Portfolio). These features have associated minimum and maximum values that should not be over passed by their current values. The interpretations related to the identification of appropriate ranges are left to the portfolio managers and to portfolio harmonization techniques [9], [10].

Note that a certain level of flexibility should be ensured when computing limits (e.g., duration or risks limits) and thus, we represent the functional feature of a limit by specifying its related parameters without making assumptions on the type of calculus performed using these parameters.

A limit is applicable to a product in the context of a portfolio and the same type of limit may be applicable to more products contemporary with different parameters (in Fig. 5 Limits are associated to a Managed Products). On a Managed Product, zero or more limits may be applied. A limit is characterized by a name and an ordered list of parameters having predefined values. For instance, a limit may establish that, within a portfolio, the products related to the European geographical area should range between 10% and 20%. In this example, the parameters are Europe geographical area and the 10%-20% range.

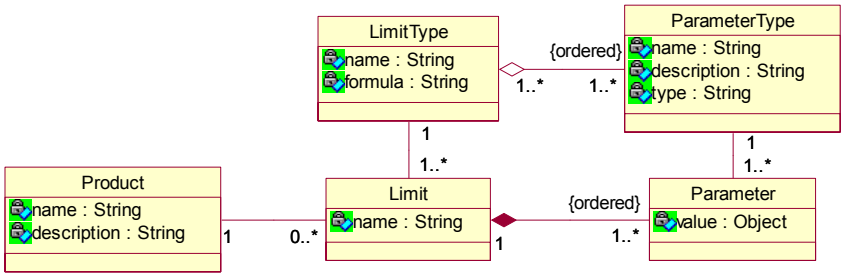

Fig. 5. Products' Limits Definition

To describe formally limits' typologies and their parameters' typologies, we have introduced a metalevel. Conceptually, the metalevel captures the semantic of the control operations: the meaning of the formal parameters of an operation is the same for all the limits of a given type, while the values of these parameters may change from limit to limit. For example, a currency limit has the following parameters: an asset, a currency, and a range. Each implementation of a limit can have different values.

The attribute *type* in the ParameterType class defines the typology of a parameter type. For example, a minimum represents a percentage value, while the currency is defined as a string. Compositions of string parameters are defined through enumerations of possible values, which may be considered as more complex types. For example, the currency values are defined as Euro or Dollar or Pound, etc. A complex type

may be composed of all these values. A current value may be defined IN or NOT IN a set of values. Conceptually, this means that all the products of a portfolio share common characteristics modelled as domains, on which it is possible to define expressions that partition the portfolio in different levels of sections.

The Limit Type class has a *formula* attribute defined by the following grammar:

```
<FORMULA>        := <LIMIT_TYPE><OPEN_PAR><EXPR><CLOSE_PAR>
<EXPR>           := <EXPR> <COMMA> <EXPR> | <QUESTION_MARK>
<LIMIT_TYPE>     := Currency | Nation | ...
<COMMA>          := ,
<OPEN_PAR>       := (
<CLOSE_PAR>      := )
<QUESTION_MARK>  := ?
```

For example, *Currency(?, ?, ?, ?)* is a valid formula.

As currently presented a limit is applicable to a Managed Product, but this does not model the real case: a limit can be defined even upon a Line and automatically it is applied to all the individually managed products belonging to that Line. This is a general rule, but there are also exceptions. Limits can be classified in different families based on the relation to their normative force: so we have some levels of priorities which range from limits imposed by institute's rules (e.g., national banks) to those imposed by the fund managers or the investors. Some limits can be derogated from others. At the moment it is not important to understand why a limit cannot be in force, but that a limit can be derogated, so the interest is on the fact and not upon the reason. Notice that this is a good example of separation of concerns, the limit A is derogated from the limit B. This derogation usually happens on individually managed products.

In Fig. 6 the Limit Family identifies the regulation body which stated the Limit.

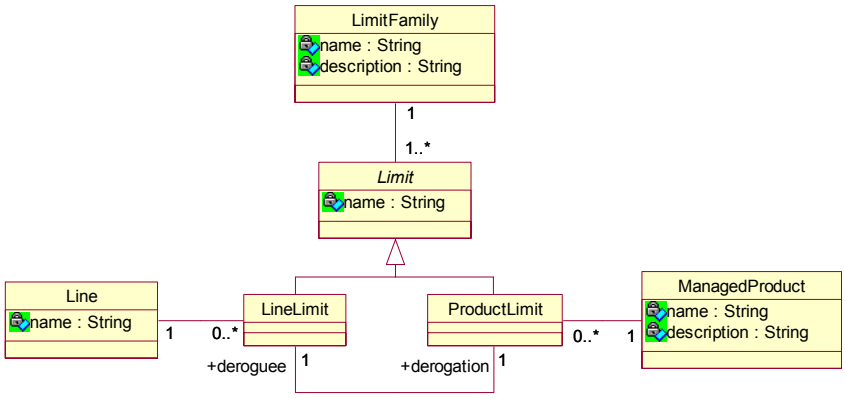

Fig. 6. Limits Families and Derogation

This concept is directly connected to a Limit and not to its type, because the same type can be used differently by different institutes. Furthermore the concept is orthogonal in relation to the derogation; and there is no investigation about the derogation cause. Derogation here can occur between a limit declared upon a Managed

Product and a limit declared upon a Line, even if in practice such derogation has been observed only between Lines and Individually Managed Products.

In this diagram, the Limit concept has been extended by Line Limits and Managed Product Limits. It means that also a Line belongs to the hierarchy of the Managed Product. A Portfolio of a Line is also applicable being defined as a set of Portfolios belonging to a Managed Product. Following this reasoning also a Line by itself can belong to the Product concept.

5 The Calculus

The management and control of limits on individual portfolio management and on collective funds are typical back office operations[3]. They are performed by the fund managers and their aim is to check the so called active management, or more precisely, to verify the coherence of purchase and sale operations of the products themselves in a portfolio. In this case, the availability of these operations over the Internet is not a valuable requirement because there are few expert users that exploit them. Thus, it is obvious to suppose that the application is used in an Intranet, and in addition, it may use rich client applications or smart clients[4].

The calculus is a typical batch operation, which is accomplished after the information about the portfolio consistence has been daily updated. The main operative issue is to achieve time performances: in our case study there are 8,000 portfolios, with an average of about 20 limits for each portfolio; hence there are around 160,000 limits operations to be performed daily.

The diagram in Fig. 7 resumes the information used in the calculus of limits omitting the other elements that describe our solution. More precisely, neither the products hierarchy, nor the distinction between the two types of limits defined upon lines and managed products are included. This ensures a better understanding of the operation sequence. The methods used during the limits calculus have been added.

In the previous sections we have concentrated our attention on the structural aspects of our solution. Now we focus on the behavioural aspects related mostly to the calculus of limits and on the similarities among their operative features. We aim to preserve the expressive potentiality meaning that a parameter within a formula can have various interpretations depending on the current formula that uses it. In the example presented, we use a limit in a so called standard way which corresponds to the 80% of the limits of our case study.

A limit in the standard form calculates the percentage of a specified quota within a portfolio over the entire portfolio or over a part of it (e.g. the stock component of a portfolio) and verifies that this value is within a previously defined range.

---

[3] Back office operations perform tasks dedicated to the management of the company itself. In the banking context, a back office operation is part of the heavyweight IT processing system that addresses clearance and settlement.

[4] Smart clients are easily deployed and managed client applications that provide an adaptive, responsive and rich interactive experience by leveraging local resources and intelligently connecting to distributed data sources.

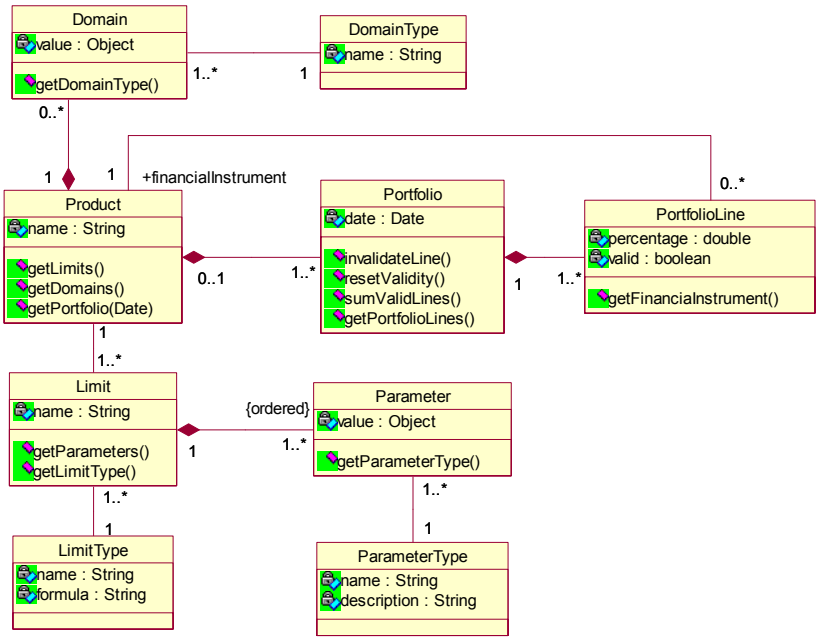

Fig. 7. The information to calculate limits

5.1 The Calculus in the Standard Case

The following operations (see Fig. 8) are performed for each portfolio and for each defined limit (in the case study the component which executes these operations has been called *Sieve*, because in practice the calculus is the iterative application of a filter upon the Products which are in a Portfolio, as the most famous one of Eratostene):

- The typology of limit calculus is checked; in this case only standard limits are considered;
- The parameters and their type are identified.

The operations performed for the calculus of limits depend on the parameters' types:

–If the parameter type identifies the asset, then it must be used to calculate the assets' consistence. The non pertinent lines within a portfolio are invalidated, and a sum is calculated over the field percentage; if the asset considered is related to the entire portfolio the sum is automatically fixed to 1. If, for example, the value of the parameter is the stock asset of the portfolio, all the Portfolio Lines which are referred to non-stock Products must be invalidated. The percentage field of the remaining lines, which are referred to stocks, must be considered together. The value obtained is stored for the calculus; we call this value ASSET.

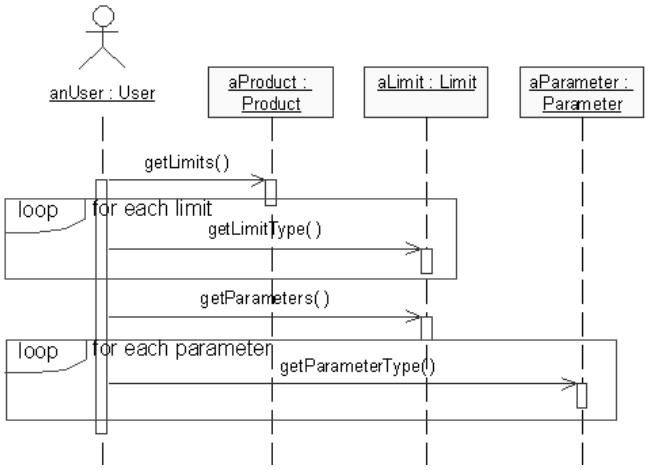

Fig. 8. Getting Limit Types and Parameter Types for each Limit on a Product

–If the parameter type represents a more specific identification of the asset (i.e. a subset of it): the related portfolio lines shall be invalidated and again a sum must be executed over the valid percentage fields. This value is stored for the calculus; we call this value ASSET_PART. For example, a parameter can identify the European currency area within the stock asset, at this point all the portfolio lines related to stock products and not related to the European currency area must be invalidated.

˜ If the parameter type is a minimum or a maximum value it is simply stored for the calculus; we call them MINIMUM and respectively MAXIMUM.

The calculus verifies the following expression:

MINIMUM <= (ASSET_PART/ASSET) <= MAXIMUM .	(**1**)

The limit is verified only if the above expression is true. In this context it is important the way in which portfolio lines are invalidated; the operation by itself is very simple (see Fig. 9). When the sum over the valid lines has been performed the validity flags are reset using *resetValidity()* method.

Attention is paid to the way in which the typology of a product (information related to its domain) is obtained. This operation is accomplished when the parameters related to the identification of the asset and of its subsets is retrieved (the parameter involved in the identification of the ASSET and ASSET_PART variables). To retrieve this information we look for the domains and their values for each product of the portfolio (see Fig. 10). In this way the calculus can be performed virtually over each domain.

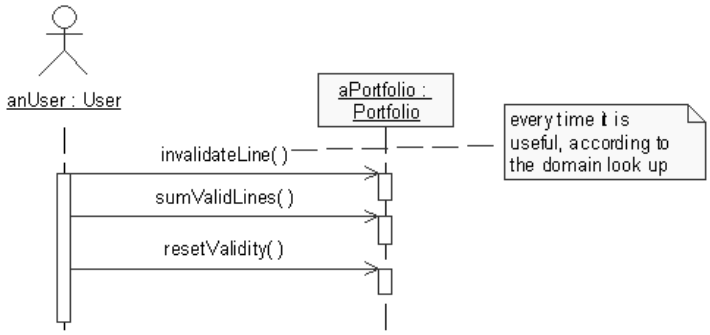

Fig. 9. Invalidate Lines and Calculate the Sum

As mentioned previously, for the evaluation of the parameters different calculus strategies can be applied depending on the nature of the parameter itself. This information can be recollected within the Parameter Calculus class in charge of implementing various calculus strategies (see Fig. 11). This class is abstract and its subclasses implement the *calculate()* method conveniently.

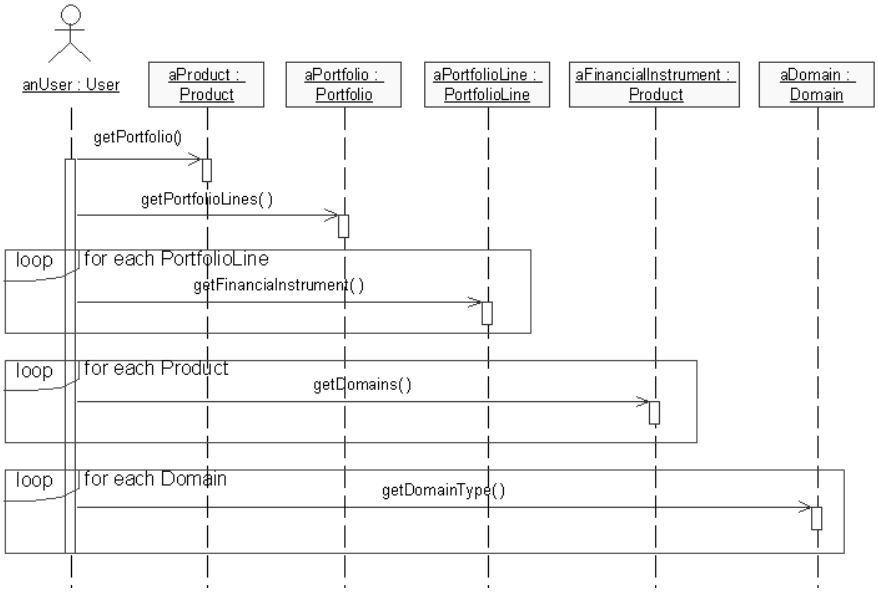

Fig. 10. Looking up for the Domain and Domain Types

To extend and foresee the capability to perform the calculus of limits not only in a standard way, the limit typologies are organized also according to the way they are calculated through an appropriate implementation of the Strategy design pattern [12]. The Limit Calculus class is abstract and all its subclasses implement the *calculate()* method (see Fig. 11).

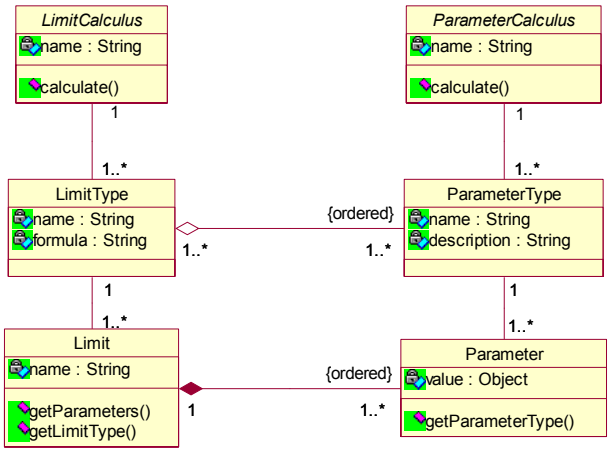

Fig. 11. The Limits' Calculus

At this point we shall not exclude, a priori, a direct link between the strategy of limit calculus and the one of parameter calculus, in that the same parameter may have different calculus strategies according to the limit calculus by which it is hosted. To ensure this capability the class diagram has been modified as follows (see Fig. 12).

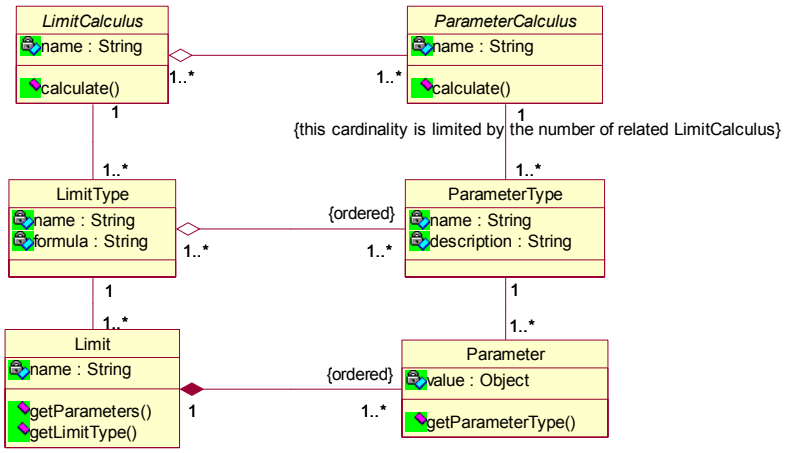

Fig. 12. Contextual Strategies

The modification involved two changes: (1) the introduction of an aggregation between the abstract class Limit Calculus and the class Parameter Calculus, and (2) the change of the cardinality of the existing association between the Parameter Type and Parameter Calculus classes. The multiplicity of the association is limited by the number of calculus implementations. Currently, the cardinality of the parameter calculus is limited superiorly by the cardinality of the calculus of the Limits Type.

6 Implementation Notes

The information related to the limit calculus has been inserted in an ER database, which is updated every day through the flows produced by the active management of the funds, more precisely by the data of selling and buying operations. These implementation details are not addressed within this paper.

The first issue is related to the usage of one or two databases because of the necessity to deal with historical data. The hypothesis of using only one database is supported by the following considerations:

- A unique database is self-contained; all information useful to the calculus relies in one place;
- Data is not duplicated;
- Consequently it is not necessary to provide synchronization mechanisms, which are mandatory when there are two databases.

Meanwhile the following facts support the use of two databases:

- The application that manages limits is an unitary one and exploits historical data;
- The limits calculus by itself could be performed in various ways and it can follow different criteria, useless to mention these implementations should coexists;
- The information related to limits is more dynamic of those related to portfolios, or in other words it has a different dynamicity.

The arguments supporting the existence of two databases were considered more consistent. Also other practical considerations suggested this solution: the implementation team was divided into different working groups that need immediately a historical series of data, even if it was not complete. This information was not affected by the loading operations and the need to manage in a separate way the process of data loading and the check of data consistence.

As the result of this choice, the duplication of information and the synchronization between the two databases have been addressed. The choice has been the one of reducing the number of duplications to avoid errors and unnecessary controls. The duplicated information was useful and necessary to identify the products which have a portfolio. The only verification methods are those related to the maintenance of the information consistence. Theoretically, it is possible to import into the limits database the information related to the values that identify domains, but this information has been included in the formulas (hence they are not present in the database tables). Actually, the interpretation of these values is possible only looking at the historical funds database.

A particular problem was related to the data loading process and to the consistence check: data is loaded through a flow given by the application which manages the operation upon the products portfolios.

To gain more performance the limits calculus has been implemented using stored procedures which work on temporary tables fulfilled with all the needed data. The

usage of such tables leads to a relevant decrease of the operational time[5]. To give more homogeneity to the implemented code, some operations have been factorized by the introduction of ad hoc stored procedures, this process raised slightly the computational time, but at the moment the result is sustainable. Note that the usage of dedicated stored procedures decreases the elaboration time, increasing meanwhile, significantly, the maintainability cost of the code and reducing the generality of the system. In fact the insertion of a new limit type, which uses similar operations imply only the updating of some tables in the database and no modifications of code[6].

7 Conclusions and Further Work

The paper has presented the design concepts and the implementation details related to a set of analysis patterns in the context of the Managed Investment Instruments. To the best of our knowledge, this domain lacks in object-oriented analysis and design approaches, being based on the concepts of the procedural programming features. Today, the advantages provided by the object-oriented approach aims to gain the attention of the financial domain.

The analysis and the design of our solution have been driven by an actual example. To achieve the project goals, we exploited both the iterative development process [13], [14] and the extreme programming approach [4], [5] that allowed us to develop and test in a short period of time (about six months) the qualitative aspects of different design solutions. The validation of the best solution depends on a set of factors including the scope of the application, its running environment, and the performances expected by the customer.

Primarily, our project was focused on the limits calculus of Managed Financial Investments. This has implied the identification and the definition of the financial products and their related portfolios. We have modeled those aspects of products and portfolios which are exploited in the calculus of their related limits. Aspects such as the distribution of the shares and quotes of funds have been ignored. However, our model can be easily extended with these or other additional aspects necessary in financial applications.

During the elaboration of the calculus phase we have validated our solution. Furthermore, the calculus approach is generic and independent to allow the addition of new limit types and calculus without affecting the overall mode. This is mostly due to the usage of strategies.

From the analysis point of view, further work will focus on extending the expressivity of limits by inserting boolean expressions, which provide the possibility to use alternative sequences of limits (e.g., introducing the operator OR) and to indicate

---

[5] The first version of our solution was based on the direct reading of tables, which last for several hours. By using stored procedures and temporary tables we have reduced the operational time from hours to minutes.

[6] The dimensions of calculus are as follows: the portfolio lines are about 30, while the analyzed products (individually managed portfolios included) are about 13,000. For each of these products there are circa 20 limits. The elaboration time is about an hour.

more complex formulas to compose expressions. From the implementation point of view, we aim to extend our approach towards a more object-oriented solution by substituting the current stored procedures and implementing the functionalities they provide by exploiting object-oriented mechanisms in order to compare the two approaches from the performances point of view. .

References

1. ***, Introduction to Fund Management. Prentice Hall (1996)
2. Altman, E., Resti, A., Sironi, A.: Recovery Risk. Risk Books (2005)
3. Antulio, Bomfim, N.: Understanding Credit Derivatives and Related Instruments. Elsevier (2005)
4. Beck, K., Fowler, M.: Planning Extreme Programming. Addison Wesley, 1st ed. (2000)
5. Beck, K., Extreme Programming Explained: Embrace Change. Addison Wesley, 2nd ed. (2004)
6. van den Brand, M.G.J., Sellink, A., Verhoef, C.: Control flow normalization for COBOL/CICS legacy systems. In: Nesi, P., Lehner, F., (ed.): Proceedings of the Second Euromicro Conference on Maintenance and Reengineering (1998) 11-19, available at http://adam.wins.uva.nl/~x/cfn/cfn.html
7. Commission of the European Communities: Green Paper on the enhancement of the EU Framework for Investment Funds (2005)
8. Directive 2004/39/EC of the European Parliament and of the Council of 21 April 2004 on markets in financial instruments amending Council Directives 85/611/EEC and 93/6/EEC and Directive 2000/12/EC of the European Parliament and of the Council and repealing Council Directive 93/22/EEC. In: Official Journal L 145 (2004) 1-44
9. Farrell, J.L., Portfolio Management: Theory and Applications. McGraw Hill, 2nd ed. (1996)
10. Fischer, D.E., Ronald, R.J., Security Analysis and Portfolio Management. Prentice Hall (1995)
11. Fowler, M.: Analysis Patterns: Reusable Object Models, Addison Wesley (1996)
12. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-.Oriented Software. Addison Wesley (1995)
13. Larman, C.: Applying UML and Patterns. Prentice Hall, 3rd ed. (2005)
14. Maciaszek, L.A.: Requirements Analysis and System Design: Developing Information Systems with UML, Addison Wesley (2001)
15. Nelken, I.: Hedge Fund Investment Management. Butterworth-Heinemann (2006)
16. Roman, S.: Introduction to the Mathematics of Finance. Sprinter-Verlag (2005)
17. Sellink, A., Verhoef, C., Sneed, H.: Restructuring of COBOL/CICS Legacy Systems. Third European Conference on Software Maintenance and Reengineering (1999) 72
18. Sneed, H.M.: Architecture and functions of a commercial software reengineering workbench. In: Nesi, P., and Lehner, F., (ed.): Proceedings of the Second Euromicro Conference on Maintenance and Reengineering. IEEE Computer Society (1998) 2-10
19. Society for Worldwide Interbank Financial Telecommunication (SWIFT): SWIFT Standards Modelling Methodology (2004)
20. Society for Worldwide Interbank Financial Telecommunication (SWIFT): Standards High-Level 2006 (2005)
21. Yoder, J., Barcalow, J.: Architectural Patterns for Enabling Application Security. The 4th Pattern Languages of Programming Conference. Monticello, Illinois (1997)