A Case Study of Object-Oriented Bio-Chemistry: A Unified Specification of the Coagulation Cascade

Jacqueline Signorini sign@ai.univ-paris8.fr
Patrick Greussay pg@ai.univ-paris8.fr
Artificial Intelligence Laboratory, Université Paris-8, 93526 Saint-Denis, France

Abstract

We propose a case study where a familiar but very complex and intrinsically woven bio-computing system - the blood clotting cascade - is specified using methods from software design known as object-oriented design (OOD). The specifications involve definition and inheritance of classes and methods and use design techniques from the most widely used OOD-language: the Unified Modeling Language (UML), as well as its Real-Time-UML extension.

First, we emphasize the needs for a unified methodology to specify complex enough biological and biochemical processes. Then, using the blood clotting cascade as a example, we define the class diagrams which exhibit the static structure of procoagulant factors of proenzyme-enzyme conversions, and finally we give a dynamic model involving events, collaboration, synchronization and sequencing.

We thus show that OOD can be used in fields very much beyond software design, gives the benefit of unified and sharable descriptions and, as a side effect, automatic generation of simulation software.

Keywords: object-oriented design, UML specifications, blood clotting cascade, proteolytic enzyme networks.

1 Introduction

OOD for non-software complex systems:
Object-oriented programming (OOP) is now the major paradigm for software design. To specify and design very complex programs, OOP requires new methods recently built upon description languages. Prior to programming, these languages yield a complete specification for the states and processes which constitute the planned task. One other reason of the success of these description languages (UML, Real-Time UML) [1, 7, 6] has to do with the power of sharing a prototype description between independent teams working in loose cooperation.

It is now a standard issue that OOP gave birth to a large set of object oriented design (OOD) methods. This structured set of methods has become a recognized craft as well as a technical methodology. The methods give specifications kept separate from the target programs which will be subsequently hand-coded or computer-generated from the description yielded by OOD.

The time has come to consider that OOD has all the power needed to specify complex domains beyond program design. They could not rely until now on: 1/ an unified and stable description methodology; 2/ a way of using concepts such as encapsulation, classes sharing and embedding, static properties and dynamic process inheritance; 3/ an opportunity to share and recombine descriptions built by separate research teams; 4/ the ability of automatically generating the code for simulation software associated with the specified situations. We thus propose, using a typical example in bio-computing - the set of processes of the blood clotting cascade - a case study which hopefully

demonstrate the need for an unified description methodology. For that purpose, our tool will be the most widely used and standardized language today for OOD: Unified Modeling Language (UML)[1].

What is the blood clotting cascade:

The process of blood clotting and the subsequent dissolution of the clot following repair of the injured tissue is termed hemostasis. It is based on an ordered series of proenzyme-enzyme conversions, also referred as the proteolytic cascade [5]. Proteolytic enzymes are proteins that can cut other proteins in pieces. As they can be extremely dangerous, they usually are formed and transported in the plasma as proenzymes (zymogens), an inactive form which on activation undergoes proteolytic cleavage to release the active factor from the precursor molecule. The coagulation pathway functions as a series of positive and negative feedback loops which controls the activation process. It ensures the formation of the cross-linked fibrin clot that plug injured vessels and prevent blood loss through the action of thrombin. All the plasma proteins involved in coagulation, mainly produced in the liver, are clotting factors designated by Roman numeral descriptors. They reflect the order of their discovery rather than their sequence in the clotting cascade. Factors XII, XI, X, IX, VII, prothrombin are proenzymes which are converted to active enzymes during coagulation. In contrast, factors V and VIII are cofactors. When a proenzyme is activated, an "a" is added to the number.

Why using UML to design such a system? First, it provides a structured analysis and object orientation approach that actually fits this information system where objects are specific entities strongly interconnected. Second, the proteolytic cascade which directs the proenzyme-enzyme conversions is actually mapped on a state diagram where the behavior of objects is identified when entering, exiting or existing in a state as well as the events accepted in that state. Third, by activity and sequence diagrams timing tools are introduced as synchronization bars, branch, contact or merge points enabling to sketch the clotting mechanism by message passing and method triggering.

2 OOD Blood Clotting: Classes

In our model, four classes are defined to fit the specific hemostatic stages. Each class is a set of objects and objects (the corresponding blood factors, carrier proteins, cofactors) are instances of a class.

A class is here represented as a three-segment box. The top segment has the class name, the middle segment contains the list of attributes and the bottom segment contains the list of operations or methods. Objects collaborate and exchange messages through the relationships defined between classes and objects. There are five types of relationships: association, aggregation, composition, generalization and dependency.

The four classes designing the complex clotting process (Fig. 1) share three kinds of relationships. The contact phase class, corresponding to the very first moment of clot formation (exposure of blood plasma to collagen in a damaged vascular wall) is connected to the intrinsic pathway class through an aggregation link characterizing its logical and physical dependance to the latter one. The link is shown with diamonds at the owner end of the relationship.

The two following classes, intrinsic pathway and extrinsic pathway, are shown with closed arrowheads indicating a generalization or inheritance relationships with the common pathway subclass. In the UML, generalization means two things: inheritance and substitutability. Inheritance means that a subclass has all its parent's attributes, operations, associations and dependencies. The subclass can extend and specialize the inherited properties. By specialization, the subclass may polymorphically redefine an operation (or statechart) to let be more appropriate to it and even substitute for the inherited behavior a new associated one [11].

This linking type definition agrees with the common pathway process responsible for the production of thrombin. First, through positive feedback loops, thrombin quickly magnifies the hemostatic activity of some procoagulant factors in the intrinsic and extrinsic pathways : factors XI, VIII, V, XIII, X,

[1] We use Rational Rose 2000 UML, Rational Software Corp.

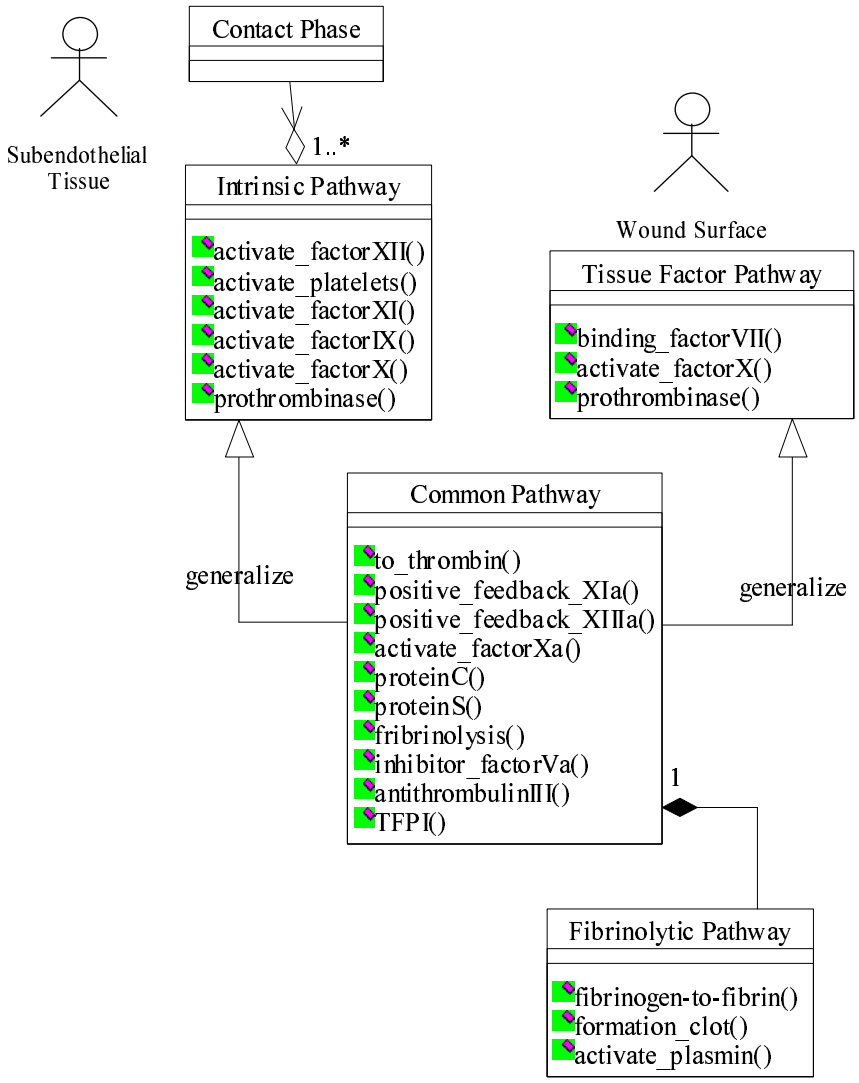

Figure 1: The four classes of the clotting process.

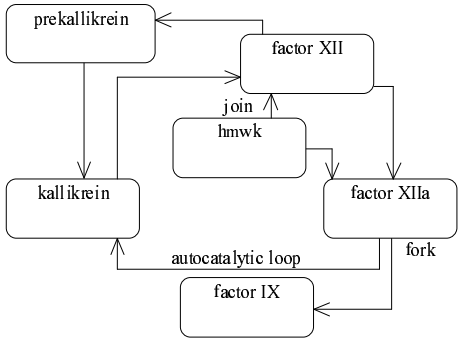

Figure 2a

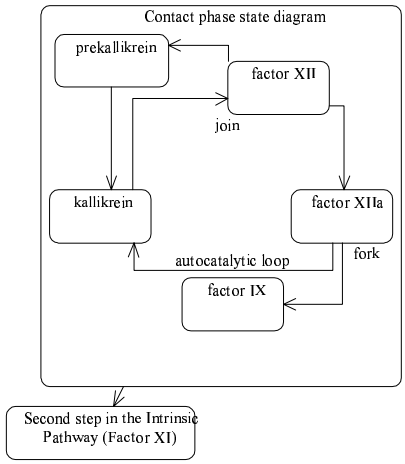

Figure 2b

Figure 2: **2a:** The contact phase state diagram. **2b:** Nested states.

IX, I (fibrinogen). Then, by negative feedback loops, thrombin bound to thrombin inhibitors [3] (antithrombin III, heparin cofactor II, a2-macroglobulin, a1-antitrysin) activates plasma proteins which degrade or inhibit coagulation factors. This endogenous regulation limits the extent of the clotting cascade.

The third relationship connecting the fibrinolytic pathway class to the common pathway class is a composition, a strong aggregation with bidirectional navigability. The fibrinolytic pathway class is a component class within the composite formed with the common pathway class. Components are included in the composite and only shared by the owner. The relationship is shown with a filled diamond at the composite end. Actually, in the clotting process, fibrinolysis is strongly dependent on the activation of thrombin, first step in the common pathway, which operates the proteolytic cleavage of fibrinogen [10, 12].

Thrombin produces fibrin monomers which then polymerize to form fibrin strands and insoluble fibrin by activation of factor XIII, last enzymatic step in the coagulation process that introduces cross-links in fibrin polymers [4, 10].

Two actors, the stick figures, are drawn outside of the scope of the system and depict the interfaces that interact with the static design of the system. The numbers at the end of a relationship line denote the number of objects that participate in the relation at each end. This is called the multiplicity role: an "*" means a cardinality of multiple objects. The value 1 on the composite role designates the necessary role of a composite, solely responsible for the creation of part components.

3 Contact Phase State Diagram

The identification of objects and their changes is concerned with object analysis identifying the essential set of objects and their relationships. The specification of dynamic behavior of those logical objects comes through object behavior designs as statecharts or state diagrams. The *state diagrams* in Fig. 2 provide finite state machines of the contact phase model.

Six states are connected by direct transitions. When an event occurs - the expose of negatively charges membranes to blood plasma - the object transitions to another object or state as a result of accepting the event or performing an action. Soon this new state is entered, an implicit timer regulates the exiting timout transition or amplifies it by self triggering. The transition itself can be

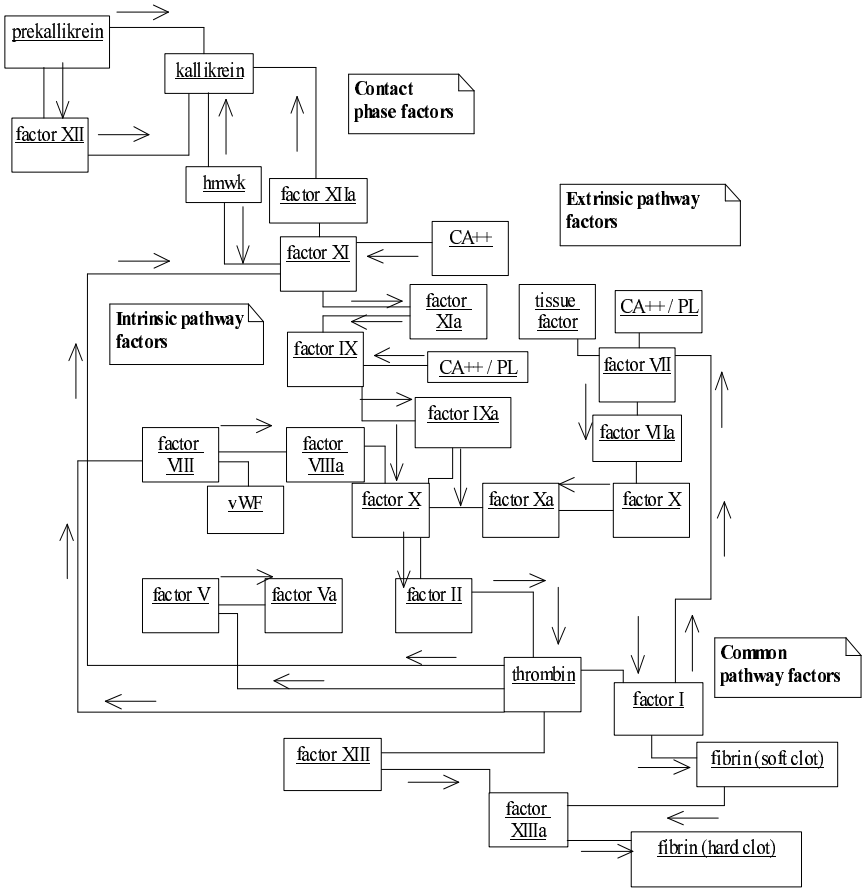

Figure 3: The blood clotting collaboration diagram.

characterized by parameters, guards or boolean expressions as well as action list. The most noticeable feature of the ULM state diagram is the nesting of states within states. As shown in Fig. 2b, the contact phase state diagram, the first step in platelet aggregation, is nested in the intrinsic pathway state diagram.

The state diagram is a dynamic view of the entire state space of a system. In the remaining part of this paper, we describe three more models for the clotting mechanism, each one focusing on a specific view of the entire state space and acting as a constructive to fully define its complete dynamic behavioral design.

4 Blood Clotting Collaboration Diagram

The first model called *collaboration diagram* shows the structural organization of the objects that participate in a given set of messages. Here, it provides the underlying structural context of the clotting process, identifying the object relationships in the four classes previously defined. Early in analysis, the structural context is the use case diagram. Use cases are defined by collaborations of objects grossly identified with actors and the system. Later, analysis captures design details and decomposes the system into objects. Use case scenarios are refined adding new levels of comprehension. Fig. 3 builds the clot formation scenario showing the set of objects working together. It provides an

order-dependent view of how the clotting system is expected to behave.

In the scenario, the extrinsic pathway cascade (right-hand side) is shown drawing an alternative route to factor Xa. When flowing blood is exposed at the site of injury, Tissue factor (factor III) is released. In the precense of calcium and phospholipids, Tissue factor and factor VII form the complex TF-VIIIa that catalyses activation of both factor X and factor IX. Factor Xa and cofactor V react with prothrombin to generate thrombin. Then, the remainder of the cascade is similar to the intrinsic pathway. The TF-VIIa complex is rapidly inactivated by Tissue factor inhibitor (TFPI) [2]. The feedback loops starting from the thrombin notation box with reversed arrows (bottom-up) show the regulatory mechanism of the clotting cascade which serves two functions: 1/ limit the amount of fibrin clot formed to avoid ischemia (blood shortage) of tissues; 2/ localize clot formation and prevent widespread thrombosis.

5 Extrinsic Pathway Activity Diagram

The following diagram called *activity diagram* provides a more refined comprehension of the structural organization of the extrinsic pathway as designed and shortly explained in the previous section (Fig. 4). It shows the flows among activities associated with the objects including transitions, branches, merges, forks and joins. It can be used to specify an algorithm or a detailed computation.

In Fig. 4, four synchronization bars are drawn identified by long, black rectangles. Actions are executed in the same order than states but transitioning between states can be subject to concurrency. Actually, some states behave as and-states requiring synchronization in propagated events. As such for thrombin, a fork bar, that sends stimuli through and-states, active at the same moment, but at some point joins control (join bar) from multiple source and-states to end its activity and prevent bleeding disorders. Hemophilia A, for example, is an X-linked disorder resulting from a deficiency in factor VIII [8] whom activation must occur via proteolytic cleavage by thrombin and factor Xa.

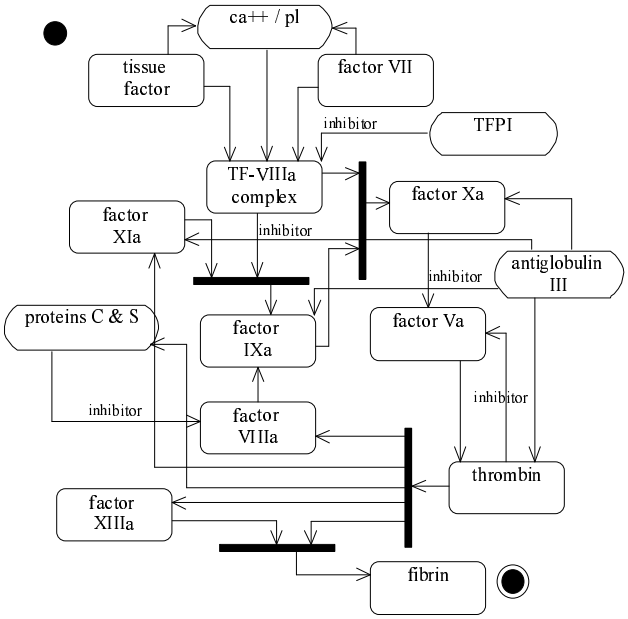

Figure 4: The extrinsic pathway activity diagram.

6 Sequence Clotting Diagram

The collaboration diagram emphasized the relationships between the objects. The *sequence diagram* emphasized the sequence of messages sent between them.

At the top of the diagram, rectangles give the object identity. Their names are underlined to distinguish them from classes. The dashed vertical line is known as the lifeline: the time axis of the diagram downwardly directed. The arrows between the lifelines represent messages being sent between the objects. The white rectangles are called activations and show the duration of a method execution in response to a message.

Fig. 5 shows our comprehension of the sequence of messages and methods between objects. VII-TF complex designates the cofactors for the extrinsic pathway, V-X complex the cofactors for the common pathway and VIII-IX are cofactors for the intrinsic pathway.

The upper part of the diagram exhibits the activation of procoagulant factors. Starting by the activate_factorX() methods in the VII-TF complex, the event cascades triggering new methods in objects up to the point where changes in object activation promote blood clotting by positive feedback loops. The lower part of the diagram is driven by clot restriction messages through methods that inhibit procoagulant factors as well as activating plasmin.

Obviously, detailed timing analysis should add time values to the diagram as constraints or synchronization patterns. However, those time values can only be investigated through coagulation *in vitro* and are themselves elements of a comprehensive model.

7 Conclusion

Object-oriented design (OOD) methods are valuable techniques for planning and prototyping complex software but also for building conceptual domain models.

In the present paper, choosing a very complex bio-computing process as a case study - the blood clotting cascade [5] - we demonstrate the effective capability of object-oriented methodology to specify domain expertise and to investigate through descriptive diagrams the constructive views of the domain.

We have used the core techniques of the standard Unified Modeling Language (UML) [1, 6] from OOD. Each one provides a specific description of the blood clotting process. Class diagrams give a structured organization of blood factors with their relationships. State and collaboration diagrams build the finite state machine of the process. Finally, with activity and sequence diagrams, we describe time-ordered proenzyme-enzyme activations which may support conditional and parallel behavior.

It is well known that commercial versions of UML, e.g. Rational Rose or its freeware substitute Poseidon, can generate Java code, corresponding exactly to the UML diagrams. In that sense, we have available the capacity of automatically generating Java simulators for the process described. Of course, this is not exactly the case: what generators are able to do is to give a Java version of the class-descriptions, the inheritance networks, the local state variables, as well as the headings for the methods: parameters and types. Obviously, the algorithmic parts of the methods (their body) must be provided by a moderately expert programmer. Fortunately, most of the algorithmic structures involved are of the type one-shot or continuous triggering, signal handling, or simple rewriting rules involving single parameters.

We will consider in a further study, the possibility of including specialized sub-languages into UML, which would give a full simulator-generator capability to the specification. In computing terms, this would add a macro-expansion capability to UML.

Now, if a design problem occurs in a biochemical network, its automatic detection, not to mention its automatic correction, is an exciting but difficult Artificial Intelligence problem. Once a standard description is established for bio-networks such issues are very likely to be investigated both by AI researchers as well as biologists.

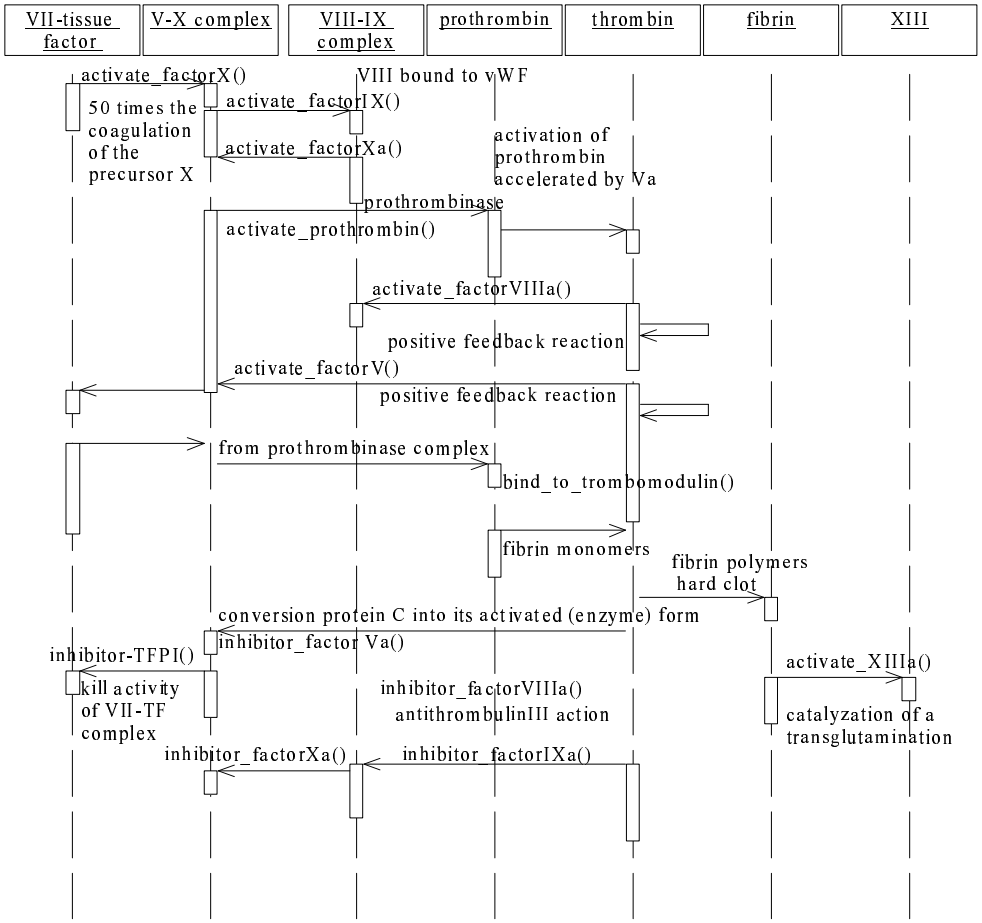

Figure 5: The blood clotting sequence diagram.

How to compare our present approach to the usual pathway diagrams that biologists are already using? An obvious anwer quite usual in software engineering is the fact that once we have several UML specifications available, it is fairly natural to consider the merging of them into a single unified description, which may not be so easy to achieve with standard biochemical diagrams. A less obvious answer would be to consider the class hierarchy as inherent to the bio-processes described and their evolution, and not only just a specification device. Of course, this exciting but very speculative issue is outside the scope of the present paper.

References

[1] Booch, G., Rumbaugh, J., and Jacobson, I., *The Unified Modeling Language User Guide*, Addison-Wesley, 1999.

[2] Broze, G.J., Tissue factor pathway inhibitor and the revised theory of coagulation, *Annual Review of Medicine*, 46:103–122, 1995.

[3] Broze, G.J. and Tollefsen, D.M., Regulation of blood coagulation by protease inhibitors, In *The Molecular Basis of Blood Diseases, 3rd Edition*, 1W.B. Saunders Co., Philadelphia, 657–679,

2001.

[4] Chambers, R.C. and Laurent, G.J., Coagulation cascade proteases and tissue fibrosis, *Biochemical Society Transactions*, 30(2):194–200, 2002.

[5] Davie, E.W., Fujikawa, K., and Kisiel, W., The coagulation cascade, *Biochemistry*, 30(43):10363–10370, 1991.

[6] Douglass, B.P., *Real-Time UML, Developping Efficient Objects for Embedded Systems*, Addison-Wesley, 2000.

[7] Fowler, M. and Scott, K., *UML Distilled, a Brief Guide to the Standard Object Modeling Language*, Addison-Wesley, 2000.

[8] Gilles, J-G., Anti-factor VIII antibodies of hemophiliac patients are frequently directed towards nonfunctional determinants and do not exhibit isotypic restriction, *Blood*, 82(8):2453–2461, 1993.

[9] Halkier, T., *Mechanisms in Blood Coagulation, Fibrinolysis, and the Complement System*, Cambridge University Press, Cambridge (England), 1991.

[10] Kimball, S.D., Thrombin active site inhibitors, *Current Pharmaceutical Design*, 1:441–468, 1995.

[11] Liskov, B., Data abstraction and hierarchy, *SIGPLAN Notices*, 23(5), 1988.

[12] Wang, W., Boffa, M.B., Bajzar, L., Walker, J.B., and Nesheim, M.E., A study of the mechanism of inhibition of fibrinolysis by activated thrombin-activable fibrinolysis inhibitor, *Journal of Biology Chemistry*, 273:27176–27181, 1998.