

# LEMMA: A Language for Easy Medical Models Analysis

Luciano Baresi <sup>\*</sup>, Fabrizio Consorti <sup>†</sup>, Manuele Di Paola <sup>†</sup>,  
Antonio Gargiulo <sup>†</sup>, and Mauro Pezzè <sup>\*</sup>

<sup>\*</sup> Dipartimento di Elettronica e Informazione – Politecnico di Milano  
Piazza L. da Vinci 32. 20133 Milano (Italy)

<sup>†</sup> IV Semeiotica Chirurgica, Policlinico Umberto I – Università “La Sapienza”  
Piazzale A. Moro. 00161 Roma (Italy)

## Abstract

*The quality of health care systems and processes is becoming a prominent problem and more and more efforts are devoted to define methodologies and tools to measure and assure quality of care. New methods are required to optimize health care processes to guarantee high quality standards within (limited) available resources. Resource optimizations able to preserve the quality of treatments require good models of medical processes.*

*This paper presents LEMMA, a new notation to model medical processes. LEMMA provides physicians with intuitive graphical elements to design their models. At the same time an high level timed Petri net corresponding to the designed model is built automatically. In this way, LEMMA models are ascribed formal semantics and can be executed and analyzed automatically. The dual language approach followed in this paper allows physicians to gain all the benefits of formal methods without being proficient in them. Medical users manage simple graphical elements, while Petri nets ensure formality and validation capabilities. In this way LEMMA mixes formal and informal notations, overcoming the problems of both the approaches.*

*The definition of the notation has been supported by the development of an environment to design LEMMA models. The environment, besides letting us experiment with the notation, has been employed to define and analyze real case studies.*

## Topics:

*Definition, Optimization and Analysis of medical processes.*

## 1 Introduction

The quality of health care systems and processes is becoming a prominent problem and more and more efforts are devoted to define methodologies and tools to measure and assure quality of care [5, 6]. In this frame the introduction of practice clinical guidelines is expected to optimize

health care outcomes and costs and to reduce malpractice. Clinical guidelines and protocols – based on scientific evidence – are aimed at supporting physicians in their decisions, even if a clear demonstration of their effects is still lacking [28]. Diagnostic and therapeutic procedures are often complex and involve an increasing number of cooperating different medical specialities, as for example in protocols for cancer patients [23]. These “new” procedures require strict coordination among involved medical scientists and better allocation of available resources. The high costs associated with these medical processes impose the adoption of techniques to optimize – or at least, improve – the diagnostic and therapeutic path of each patient. This can be achieved by both organizational changes and decision support offered to physicians. For example, the use of guidelines led to cost reduction in the management of pediatric emergencies and in the therapy of peptic ulcer [8, 3]. Thus the capability of analyzing diagnostic processes would allow physicians to better allocate and use available resources and to impose a strict control over associated costs. At the same time, they would provide patients with better services by avoiding bottlenecks and unjustified delays.

Resource optimizations, able to preserve the quality of treatments, require good models of diagnostic and therapeutic processes [14]. Medical processes have been modeled in many different ways, ranking from natural language to mathematical (formal) notations. Informal methods, like natural language, are easy to understand and do not force physicians to use unnatural and cumbersome notations. Unfortunately, they retain ambiguities and provide little support to the analysis of modeled processes. Formal methods overcome the problems of informal methods, but are difficult to understand without a specific mathematical background. Physicians take the risk of jeopardizing their effort in describing medical procedures. They cannot concentrate mainly on the problems, but they have to cope with the way to formalize them as well.

This paper presents LEMMA, a new notation for medical processes, which mixes formal and

informal notations, overcoming the problems of both the approaches identified so far. LEMMA is a domain specific graphical language developed by a joined team of computer scientists and professionals of medical science. The notation merges high expressive specific constructs with a formal definition: each element of the notation is ascribed operational semantics by means of high-level timed Petri nets (HLTPNs) [11]. The proposal is based on technology successfully used in software engineering [2, 16]: a user-friendly notation accessible by non experts, that is automatically translated onto a formal model used to execute and analyze user processes. Physicians define clinical and diagnostic models by means of an easily understandable language, and they gain all the benefits of the formal representation to assess their models. LEMMA does not require physicians to be proficient in Petri nets to understand the results. The formal representation, that is, the corresponding Petri net, can automatically be executed and analyzed. The results are translated in visualizations of LEMMA elements.

The paper is organized as follows. Section 2 presents the state of the art in modeling diagnostic processes. The theoretical approach followed while formally defining LEMMA is described in Section 3. Section 4 introduces Petri nets: the formal model for LEMMA. Section 5 describes the notation: it presents each element with the associated formal representation given in terms of Petri nets. Analysis capabilities are described in Section 6. The use of LEMMA to model diagnostic processes is exemplified in Section 8, which proposes a model of the diagnostic procedure of colon-rectal cancer. Finally, Section 9 draws some conclusions and identifies future work.

## **2 Related Work**

Clinical processes are usually defined in large references databases like MEDLINE or Cancer-Net. Procedures are described in plain text. Then, the interpretation of this knowledge has to face all

ambiguities of natural language. The dissemination of WWW technologies will probably enhance the accessibility and value of this kind of informal knowledge [18].

Several research projects in the field of medical informatics have addressed the problem of a more formal representation of clinical procedures to improve automated processing. In most cases used representations are suitable to developers of systems and applications; more friendly notations imply weak formal models. Project PRESTIGE (Patient Record Supporting Telematics and Guidelines) uses networking and knowledge based systems to support the generation, dissemination and application of guidelines and protocols. It relies on an object-oriented model, which defines the kind of entities that belong to a protocol, together with their relations and attributes. This model is employed to release a *Guideline Authoring and Dissemination Tool*, currently under development. The tool will be used to create and edit protocol knowledge bases [13, 12].

The main goal of project IREP was the representation of the cooperation and interaction among different actors involved in a complex clinical process. The model was focused on temporal aspects of the management of a single patient. Even if the model was not fully formal, it was intended to be used by system designers. The methodology was influenced by concepts developed in the frame of workflow management systems [9]. A Workflow Management System (WFMS) is a system which defines, manages and executes workflow processes based on a logical representation according to the Workflow (WF) paradigm [20]. Processes are classified in: *physical processes*, in which physical objects are processed by specific tasks, *information processes*, in which information is produced, managed and processed, and *business processes*, which describe the activities of an organization to satisfy the specific needs of users. A first attempt to represent clinical activities according to WF methodology is documented in [19].

At a conceptual level WF representation is not too complex to be used by rather naive users and many of the available WFMSs provide them with graphical user interfaces to define their processes.

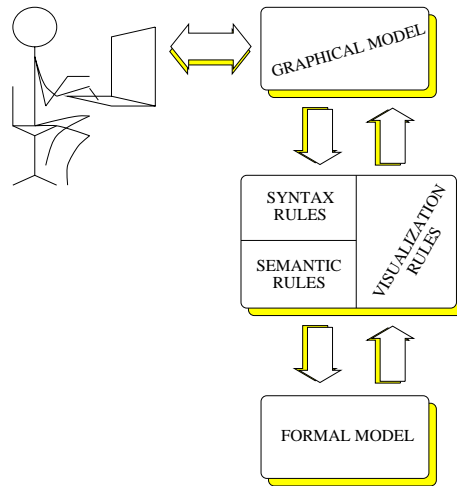
At the same time, the lack of standardization and tools for the analysis, testing and debugging of designed WF still precludes a wider use of WFMSs.

Finally some systems explore the possibilities offered by the WWW environment. FENARETE, GEODE and InterMed are tools to design and distribute clinical protocols over the internet. These systems allow physicians to describe protocols symbolically: they provide graphical primitives that represent the basic elements of a protocol. A protocol can then be retrieved and browsed, but not executed or simulated [7, 17, 24]. SMART represents clinical processes by means of a graphical formalism. The system retrieves a protocol from its knowledge base and monitors its execution step by step for a single patient. An HTML client can query the database via the internet, allowing a remote monitoring of clinical processes [27].

### **3 Approach**

LEMMA is defined formally by means of the methodology presented in [2, 1]. The approach is a rule-based framework that supports the definition of the syntax and semantics of graphical notations. A language is "formalized" by means of three sets of rules:

- *Syntax Rules* specify the syntax of the notation. They define the elements of the notation and the way they can be connected.
- *Semantic Rules* specify the semantics of the notation. They define the semantics of the notation as a translation on a formal model.
- *Visualization Rules* specify the way execution and analysis results are presented. They translate execution and analysis results on the formal model in visualizations of the elements of the graphical notation.



**Figure 1. The approach on which LEMMA is based.**

Figure 1 illustrates the approach. End-users – in this case, physicians – interact with a graphical specification notation (LEMMA) to define their models. Syntax rules are applied on these models to check their syntactic correctness and trigger the associated semantic rules. For each syntax rule there is a semantic rule that describes the corresponding action on the formal model: the high-level timed Petri net (HLTPN).

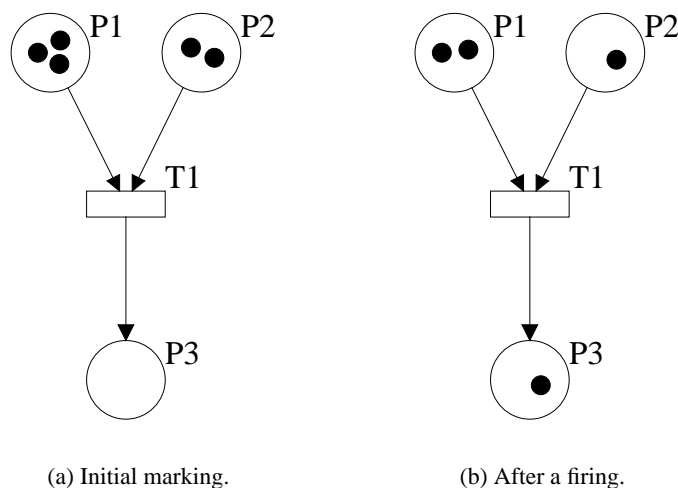
Semantic rules – triggered by syntax rules – build the formal model automatically. The formal representation ascribes semantics to LEMMA models and provides execution and analysis capabilities. In this way, semantic properties of LEMMA specifications can be checked on the formal model by using suitable algorithms, for example: execution or reachability analysis.

Visualization rules translate results of execution and analysis of the semantic model in constructs that can easily be understood by application specialists. Physicians can thus even ignore the underlying formal model, but still benefit from it to analyze and validate their specifications.

## 4 Petri Nets

Petri nets [21] are a graphical formal notation well-suited to model a broad variety of systems. Petri nets combine simplicity (graphical representation) and a rigorous mathematical framework to describe concurrent, asynchronous, parallel, and non-deterministic systems.

A Petri net – Figure 2 – is a directed bipartite graph with an initial state: nodes are divided in places, represented by means of circles, and transitions, drawn as small rectangles. Transitions define the events that characterize a systems; places are pre- and post-conditions associated with events. Arcs connect either places to transitions or transitions to places.



**Figure 2. A toy Petri net.**

The toy net of Figure 2 comprises transition  $T1$  and three places:  $P1$ ,  $P2$ , and  $P3$ . Places  $P1$ ,  $P2$  - the preset of  $T1$  - define the pre-condition that must hold to let event  $T1$  happen. Place  $P3$  - the postset of  $T1$  - identifies the condition that holds after the firing of  $T1$ .

The marking (state) associates each place with a non negative integer. From a graphical view-point, this means that each place contains a non negative number of small blacken circles, called tokens. The arrangements of all tokens in all places of a net is the marking of the net. A transition

is enabled in a given marking, if the conditions associated with all places in its preset hold, that is if each place (in the preset) contains – at least – one token. The firing of the transition removes one token from each place in the preset and produces one token in each place of the postset. If two or more transitions are enabled contemporarily, the one that actually fires is chosen non deterministically.

In Figure 2(a) transition  $TI$  is enabled and its firing removes one token from places  $P1$  and  $P2$ , and produces a token in place  $P3$  (Figure 2(b)). Since place  $P1$  contains three tokens and place  $P2$  has two tokens, transition  $TI$  would be enabled and could fire twice.

Besides a rigorous definition of execution, the mathematical framework allows Petri nets to be analyzed rigorously. Analysis capabilities let designers check automatically their models against specific properties. Two kind of properties can be studied: those which depend on the initial marking, and those which are independent from the marking. The former ones are state-dependent properties, while the latter ones are structural topology-based properties.

The basic Petri net model described so far has been extended to improve modeling power. In this paper, we use a specific extension called *High Level Timed Petri Nets (HLTPN)* [11]. HLTPN associate information with tokens. Tokens are not just markers, but they can carry typed data. Each transition has a predicate and an action. The predicate selects the tokens that enable the transition according to their values; the action defines the contents of the tokens produced by the firing of transitions. Moreover HLTPNs allow users to associate a time interval with each transition. The time interval defines the time frame within which the enabled transition has to fire. The analysis techniques defined for the basic model have been extended and adapted to cope with high level nets as well [15, 10].

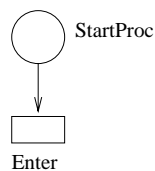
## 5 LEMMA

LEMMA is a graphical operational notation that couples simplicity and formal foundations. Physicians define medical processes by using self-explaining graphical symbols, while the operational semantics supplied by the underlying Petri net ensures simulation and analyzability.

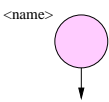
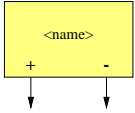
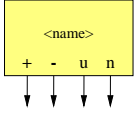
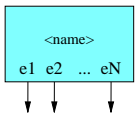
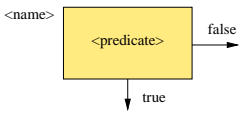
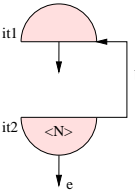

A LEMMA model comprises a topology, that describes the diagnostic paths followed by patients, and a set of symptoms. Symptoms define the cases for which the modeled process is applicable: they identify the domain of the model. In LEMMA, nodes correspond to steps of the diagnostic process; edges model the order in which such steps have to be performed. Patients – the main actors of processes – undergo a set of steps, in the order given by the edges. The path followed by patients in processes depends on their symptoms and the decisions taken by physicians. The results of clinical tests and the symptoms observed when entering the process guide physicians in their choices.

The lexicon of LEMMA is summarized in Table 1. Lexical elements are presented in the following paragraphs by illustrating their behavior and explaining their formal operational semantics, that is, the associated Petri net. Due to space limitations HLTPNs are not described in detail. Readers should get an idea of the behavior of each component: the details are provided in [1].

### Entry Points



**Figure 3. HLTPN associated with *Entry Points*.**

<i>Entry Point</i>		Represents an entrance in a LEMMA model of diagnostic process
<i>Clinical Test (2 outcomes)</i>		Models a “generic” clinical investigation. Possible outcomes are: positive or negative
<i>Clinical Test (4 outcomes)</i>		Models a “generic” clinical investigation. Possible outcomes are: positive, negative, null, or undefined
<i>Set of Clinical Tests</i>		Models a set of clinical tests. The order in which they are executed is not relevant. The result is a combination of the results of all tests in the set
<i>Symptoms Selector</i>		Guides the flowing of patients into LEMMA processes by comparing symptoms associated with patients and predicates associated with the selector
<i>Iterator</i>		Governs the repetition of subprocesses within a LEMMA model: the steps embodied in the iterator are repeated until condition <i>b</i> holds, but no more than <i>N</i> times
<i>Exit Point</i>		Models an exit from a LEMMA model of diagnostic process. It could be the entry point of a different process

**Table 1. The lexicon of LEMMA.**

Patients enter processes through *entry points*. The status of a patient is coded through a “virtual” health record, that describes the patient anamnesis. Entry points represent the phases during which physicians associate patients with sets of symptoms. Usually, a checkup of involved patients helps physicians in filling the “virtual” health record.

A LEMMA model can have more than one *entry point*. Patients can have already done some clinical investigations or present specific symptoms. This means they enter the model from specific *entry points*.

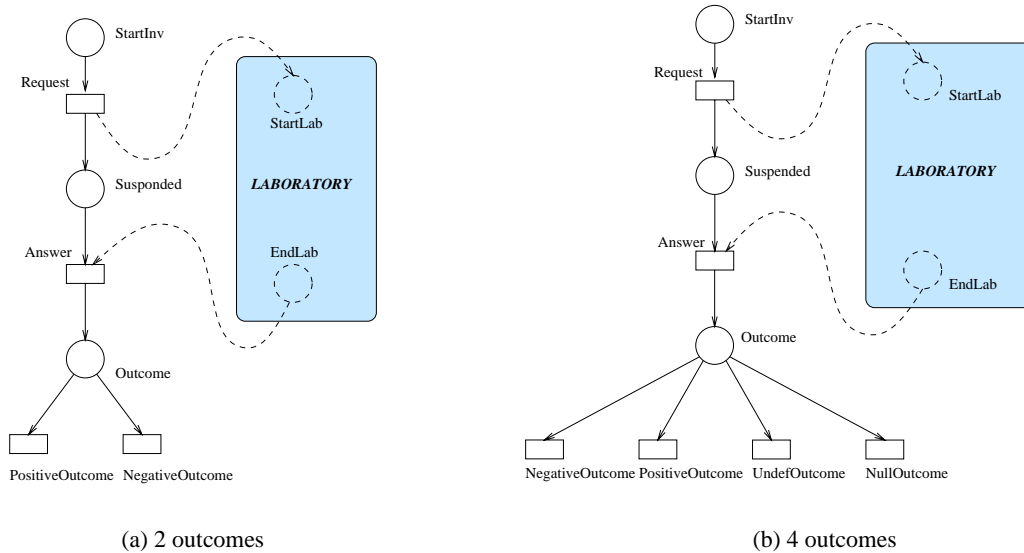
The Petri net of Figure 3 is self-explaining. Place `StartProc` contains “virtual” health records – tokens – associated with patients. Transition `Enter` moves patients (their health records) in the process.

## **Clinical Tests**

*Clinical tests* model the execution of medical investigations. These investigations can be described by using clinical tests with either two outcomes (positive or negative) or four outcomes (positive, negative, null or undefined). Investigations are executed in medical laboratories. In this paper we present a very simple model of *clinical laboratory*. This model can be tailored to fit different needs: optimization of resources or strict control over costs.

Currently, the execution of a clinical test consists in sending a request for a specific investigation to a *laboratory* and getting the result. All requests for a given kind of investigation are served by the same *laboratory*. More specific models could take into account limited resources and specific expertises.

*Clinical tests with 2 outcomes* (Figure 4(a)) model the execution of “generic” investigations. Physicians define the actual clinical test by assigning a label to the graphical symbol (< *name* > in Table 1).



**Figure 4. HLTPNs associated with *Clinical Tests***

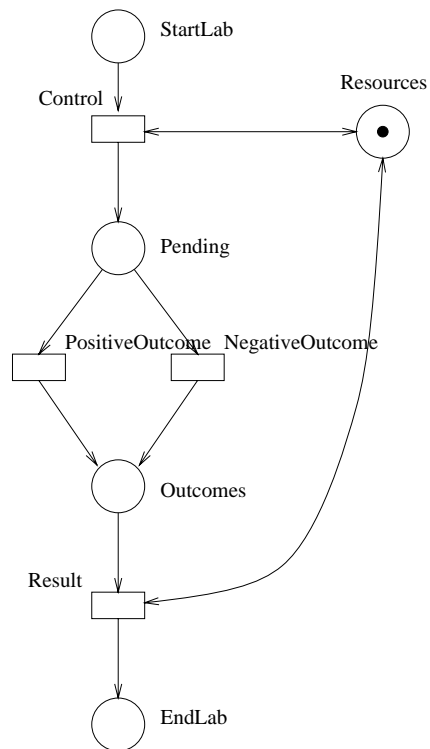
In Figure 4(a) place `StartInv` collects tokens (health records) from previous blocks. Transition `Request` forwards requests for investigations to subnet *laboratory* and puts a token in place `Suspended` to model a patient waiting for the outcome of the laboratory investigation. Subnet *laboratory*, which manages the execution of the investigation, produces results read by transition `Answer`. This transition associates outcomes with suspended patients. The `Outcome` is then read by either transition `PositiveOutcome` or transition `NegativeOutcome`.

*Clinical tests with 4 outcomes* are very similar to the previous ones. They take into account two more outcomes: null and undefined. In this way, physicians handle even these “anomalous” outcomes and define more detailed medical procedures. *Clinical tests with 4 outcomes* (Figure 4(b)) add two more transitions to the net of Figure 4(a), whose meaning is obvious.

## Laboratories

Figure 5 presents a simple model of a *laboratory*. The model collects all the requests for a given investigation in place `StartLab` and controls the availability of resources by means of transition

Control and place Resources. Accepted requests are stored in place Pending.



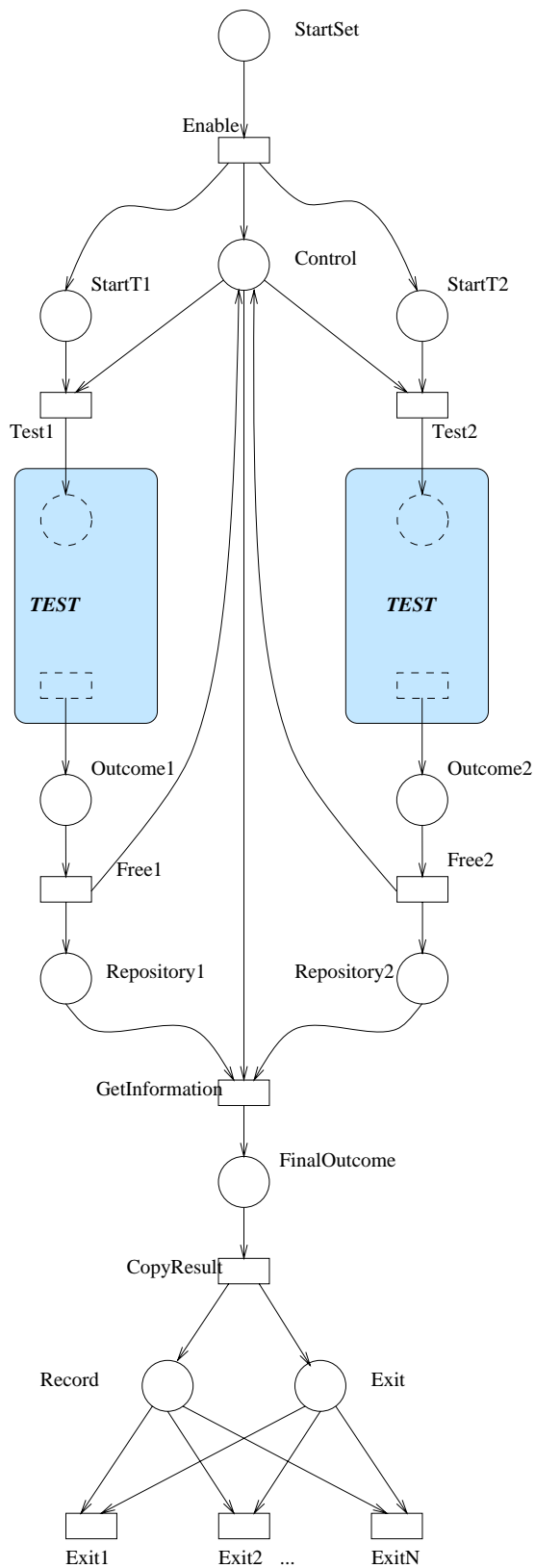
**Figure 5. HLTPN associated with *Laboratories***

The model of Figure 5 computes the result for *clinical tests with 2 outcomes*. To cope with four outcomes we should modify the subnet that defines the outcomes: currently only transitions NegativeOutcome and PositiveOutcome are modeled. The actual outcome is written in place EndLab by transition Result, which also frees allocated Resources.

This simple subnet can be extended and modified to handle specific cases and consider more sophisticated controls. The only constraints are the two places (StartLab and EndLab) that define the interfaces of element *laboratory*.

### Sets of Clinical Tests

Clinical tests can be grouped in *sets of clinical tests* to model the execution of sets of investigations in unspecified order. The number and the meaning of exit points of *sets of clinical tests*



**Figure 6. HLTPN associated with *Sets of Clinical Tests***

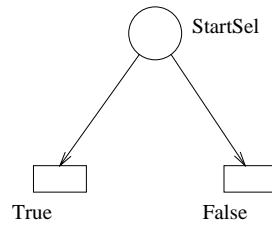
are defined by physicians. Since the number of combinations grows exponentially in the number of investigations, physicians can define rules to simplify the definition of the meaning of each exit point. For example, physicians can consider positive outcomes dominant on other outcomes. In this case, a single positive outcome determines a positive result regardless of the outcomes of the other investigations.

The Petri net of Figure 6 defines a framework to control the execution of a set of – two, in this case – clinical tests. Transition `Enable` initializes the subnet by writing one token in places `Start1`, `Start2` and `Control`. Place `Control` prevents patients from doing two clinical tests at the same time and stores their health records. Transitions `Test1` and `Test2` synchronize the executions of the set of clinical tests by enabling the subnet corresponding to one of the investigations. These subnets are the models of the clinical tests described in Figure 4. At the end of each single investigation, transitions `Free` store the results in places `Repository` and enable further tests (place `Control`).

Only at the end of all the clinical tests (i.e., all places `Repository` contain information on the same patient) place `FinalOutcome` contains the comprehensive result of the whole set. The rules used to compute the result are part of the action of transition `GetInformation`. Finally, transition `CopyResult` defines which exit has to be enabled. It produces a token that identifies enabled exit (transition) in place `Exit` and stores the health record of the patient in place `Record`.

## **Symptoms Selectors**

*Symptoms selectors* are split points of diagnostic paths. They determine the path to be followed by patients. The decisions are taken by comparing the symptoms of patients with the symptoms associated with the selector. *Symptoms selectors* are defined by means of predicates, that describe the condition that leads to exits *true* and *false*. These predicates are coded by listing the symptoms



**Figure 7. HLTPN associated with *Symptoms Selectors***

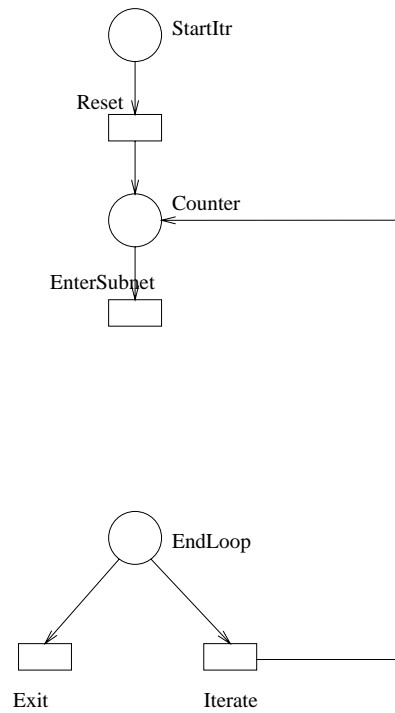
that must be present, and the ones that must be absent.

The Petri net that defines a *symptoms selector* is shown in Figure 7. When patients – tokens – are in place `StartSel`, the information in their health records is used to evaluate the predicates of transitions `True` and `False`. Since the predicate of transition `False` is the negation of the one associated with transition `True`, the two transitions (exits) are enabled in mutual exclusion.

### **Iterators**

*Iterators* govern the flow of patients through subprocesses. They indicate the maximum number of times a sequence of steps has to be undertaken to complete the process. Patients have to repeat the subprocess within element `it1` and `it2` (Table 1) for – at most –  $n$  times. Patients can exit earlier if exit conditions are satisfied.

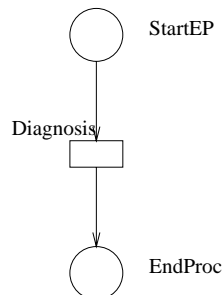
The Petri net of Figure 8 consists of two parts. The first one starts from place `StartItr`. A token (patient) in this place triggers transition `Reset`, that resets the counter. Place `Counter` accounts for the number of iterations and enables transition `EnterSubnet`. The execution of the subnet (not reported here) writes a token in place `EndLoop`. The contents of this place (number of performed iterations) triggers either transition `Exit` or transition `Iterate`. In the former case, patients would leave the loop, while in the latter case they would go on iterating.



**Figure 8. HLTPN associated with *Iterators***

**Exit Points**

Patients leave processes through *exit points*. The arrival of a patient at an exit point is associated with the release of a diagnosis. This diagnosis can either identify a precise problem, and thus lead to a therapeutic process, or recommend further clinical tests not modeled within the current process, and thus lead to another diagnostic process. In this case, the diagnostic process highlights either lack of information or new pathologies.



**Figure 9. HLTPN associated with *Exit Points***

Figure 9 – the semantics of *exit points* – is self-explaining: place `StartEP` receives tokens (patients) from previous blocks. Transition `Diagnosis` evaluates whether the diagnosis corresponds to the one identified by place `EndProc` and, if it is the case, writes a token in this place.

## 6 Analysis Capabilities

LEMMA models can be validated through the execution or the analysis of the underlying HLTPNs. The results of executing or analyzing the HLTPNs are suitably displayed in LEMMA. The executions of the underlying HLTPNs are displayed through the *LEMMA Editor* by associating firings of HLTPN transitions with animations of the corresponding LEMMA elements. The executions (simulations) are initiated by inserting patients with specific symptoms in the model. Such initializations are translated into corresponding initializations of the underlying HLTPN. Animations are given by moving patients through the paths of diagnostic models according to the firings of the corresponding transitions. Transition firings are driven by the symptoms of patients, the outcomes of clinical investigations and the choices of physicians. An example of animation is presented in Figure 11, which shows the process for the diagnosis of colon-rectal cancer with a patient at exit point *diagnosis* and the diagnostic path followed by the patient highlighted suitably.

Syntactic and semantic correctness of LEMMA models can formally be analyzed by checking the syntax and the semantics of the corresponding HLTPNs. Syntactic analysis checks syntactic consistency and completeness of LEMMA elements and their connections. It can reveal the absence of required elements (e.g., *entry* and *exit points*), open input/outputs (e.g., unconnected outputs of clinical tests), and wrong connections (e.g., wrong sequences of clinical tests within iterators). Semantic analysis can identify incorrect states or paths that could wrongly be reached or executed. Examples of semantic errors, that can be identified on the underlying HLTPNs, are dead paths

and wrong chains of selectors. Dead paths, that is, paths that can never be taken, correspond to inaccuracies or errors in the model. Incorrect chaining of selectors can introduce unforeseen or wrong combinations of symptoms that would wrongly guide patients in the model.

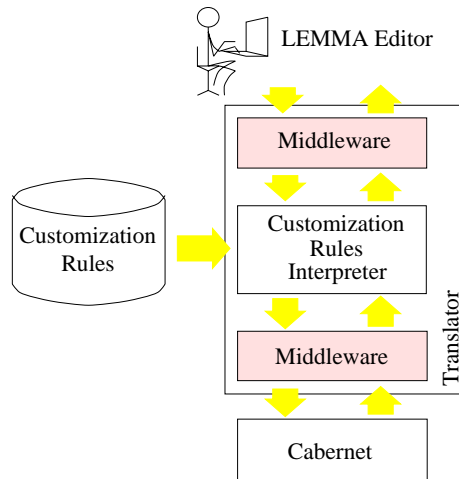
Cognitive analysis allows physicians to retrieve information about their diagnostic models: specific sequencing among sets of clinical tests, the required number of specific clinical tests, the average time spent by a patient in the model. Statistic and timing analysis can provide optimization criteria, for example, they can suggest the optimal order of clinical tests to minimize the hospitalization of patients and reduce costs without affecting the quality of the process. A detailed explanation of available techniques is given in [1].

## 7 Prototype Environment

In parallel with the definition of the notation, we started the implementation of an environment supporting LEMMA. The goal was the creation of a prototype system where to experiment with LEMMA and demonstrate its capabilities to physicians.

The toolset supporting LEMMA was released as a customization of a more general environment, which is aimed at providing graphical notations with formal semantics and analysis capabilities [2, 1]. The logical design of Figure 10 recalls the layered organization of the approach described in Section 3.

Physicians interact with the environment through a dedicated graphical editor called *LEMMA Editor*. The editor, besides providing users with all functionalities of graphical editors, allows them to validate (i.e., execute and analyze) designed models. It is implemented in *tcl-tk* [25] to widen the set of possible users. *Tcl-tk* ensures portability among different platforms (UNIX, Mac, WindowsNT, Windows95) and does not constrain physicians to specific systems to run the editor.



**Figure 10. The high-level design of LEMMA environment**

*LEMMA Editor* communicates through a middleware with the *Customization Rules Interpreter (CRI)*. This software component carries out two different activities. It governs:

- The creation of the formal representation (i.e., the Petri net) of a LEMMA model. The *CRI* interprets the syntax of LEMMA models according to *Syntax Rules* (Section 3) and produces inputs to build the Petri net by applying *Semantic Rules*. These rules - syntax and semantic rules - are defined as graph grammar productions [22].
- The visualization of the execution and analysis of LEMMA. It translates states and events of the formal model into suitable animations of LEMMA elements. *Visualization Rules* - textual rules defined in a procedural C-like language - map Petri net markings (states) into proper visualizations suitable to physicians and transition firings (events) into animations of corresponding LEMMA elements.

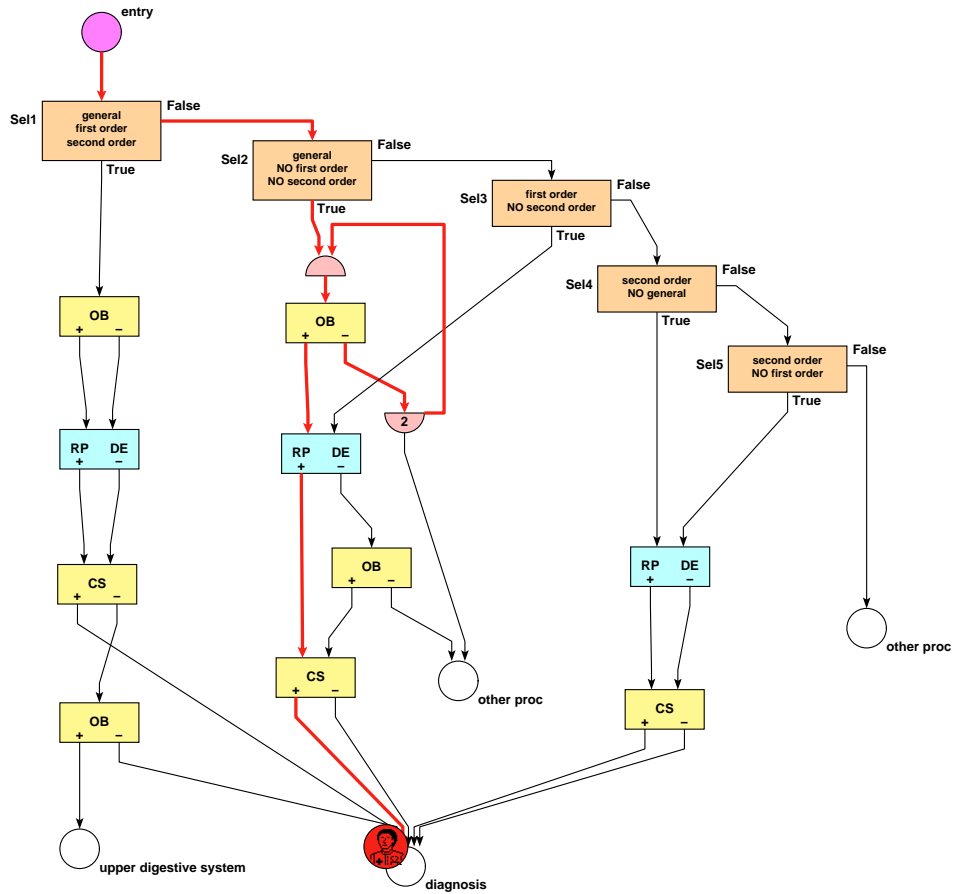
The last component identified in Figure 10 is *Cabernet* [26]: a Petri net-based CASE tool, developed at Politecnico di Milano, that allows users to define, execute, and analyze high-level timed Petri nets. In the environment described so far, it acts as the formal engine, that supplies execution and analysis capabilities. *Cabernet* receives from *CRI* the inputs to build the Petri net and communicate

to *CRI* the results obtained by executing and analyzing it. Even if this component is the means by which LEMMA gains formal foundations, it remains hidden to physicians. The translation activity performed by *CRI* lets physicians ignore *Cabernet* and its internal notation (Petri nets), but – at the same time – they can exploit all the benefits rising from its use.

## 8 Case Study

The prototype was installed on a HP workstation at the 4th Institute of General Surgery in Rome. LEMMA has been used during three months by different physicians with various skills. Few hours of training were required to be able to manage the language in a confident way. Although a formal evaluation was not carried out, LEMMA proved to be intuitive and expressive enough to represent different kind of procedures.

Moreover, the prototype has been employed to validate a protocol of the diagnosis procedure of colon-rectal cancer [4]. The original description in natural language was “interpreted” according to daily practice to solve ambiguities. Symptoms are grouped into three classes, general, first order, and second order symptoms, according to their probabilistic specificity for rectal cancer. Each group or combination of symptoms can act as a trigger to start a specific procedure, as suggested in the original documentation. The diagnosis of colon-rectal cancer may require the following clinical tests: occult blood testing, rigid proctoscopy and double contrast barium enema – performed together – and colonoscopy. The order of execution of proctoscopy and double contrast barium enema is irrelevant from the diagnostic viewpoint. The overall evaluation of the two aforementioned clinical tests is given by the dominance of the positive result. Even if colonoscopy could lead immediately to a diagnosis, it is an expensive, time- and resource-consuming exam and implies also some small risks for the patient. This is why it is usually preceded by the other tests, which must



**Figure 11. LEMMA model of the diagnosis procedure of colon-rectal cancer**

be completed by further clinical investigations.

The LEMMA model of the diagnosis procedure of colon-rectal cancer is shown in Figure 11. The corresponding Petri net is presented in Figure 12. Even non expert readers can get an idea of the different levels of complexity at a first glance.

Patients enter the process from the only available entry point, and can follow either one of the four different diagnostic paths, identified by the four main streams in the model. The first path corresponds to patients with general, first and second order symptoms. It can result in either a diagnosis or a request for further investigation with respect to upper digestive system. The second path corresponds to patients with only general symptoms. This path, besides releasing a diagnosis, can move the patient to another unspecified diagnostic process. The third path – patients with



## 9 Conclusions

This paper presents LEMMA, a new notation for diagnostic and therapeutic processes. LEMMA is given formal semantics through the dual language approach described in [1, 2]. In this way the notation is ascribed formal semantics by means of high-level timed Petri nets, which remain completely hidden to medical users. Physicians gain all the benefits of formal models, without the need to become proficient in them.

LEMMA could be useful to solve different problems. It can be employed to optimize protocol models, study the allocation of available resources, assist physicians and patients in their choices, and teach medical protocols.

LEMMA has been validated on real case studies, that show its suitability to model diagnostic processes and its usability in a medical environment. The analyses performed on the case studies have been extremely useful to check the correctness of the models.

We are currently improving LEMMA by adding new elements to widen the set of processes that can be described and improve its modeling power. Typical examples are *extended selectors* and hierarchical models. Extended selectors should substitute current chain of selectors and thus should improve the expressive power of LEMMA. Hierarchical models are a good means to structure diagnosis models and relate different protocols into a “complete” diagnostic process. We plan also to increase analysis functionalities by integrating the environment supporting LEMMA into a more general framework to include additional information to drive diagnostic processes.

## Acknowledgments

The authors are grateful to Fabrizio Denna and Antonio Re for their hard work on LEMMA. Luciano Baresi and Mauro Pezzè are also indebted to the “Cab Crew” for their daily help.

## References

- [1] L. Baresi. *Formal Customization of Graphical Notations*. PhD thesis, Dipartimento di Elettronica e Informazione – Politecnico di Milano, 1997. (in Italian).
- [2] L. Baresi, A. Orso, and M. Pezzè. Introducing Formal Methods in Industrial Practice. In *Proceedings of the 20th International Conference on Software Engineering*. IEEE-CS Press, 1997.
- [3] L.R. Burns, B. Denton, S. Goldfein, L. Warrick, B. Morenz and B. Sales. The Use of Continuous Quality Improvement Methods in the Development and Dissemination of Medical Practice Guidelines. *Qual. Rev. Bull.* 18:434-39 1992.
- [4] CNR - oncologia. *I tumori del colon-retto*. Ministero per la ricerca scientifica e tecnologica - Ministero della sanità, September 1985. (in Italian).
- [5] A. Donabedian. The Quality of Care: How Can It Be Assessed? *JAMA* 260(12), 1988.
- [6] A. Donabedian The Seven Pillars of Quality. *Arch. Pathol. Lab. Med.* 114:115, 1990.
- [7] A.M. Errera, P. Merialdo, A. Orsano, G. Sindoni and G. Rumolo. Clinical Protocol Development Using Inter/IntraNet Technology: The FENARETE System. *Medical Informatics Europe '97 IOS press* 1997.
- [8] S.A. Finkler, D. Schwartzben. The Cost Effects of Protocol Systems : The Marginal Cost - Average Cost Dichotomy. *Med.Care* 26:894-906, 1988.
- [9] A.M. Florit et al. Context Trees: Representing Co-operative Healthcare Activities. In *IREP Proceedings of MIE 94*, 1994.
- [10] C. Ghezzi, D. Mandrioli, S. Morasca, and M. Pezzè. Symbolic Execution of Concurrent Systems using Petri Nets. *Computer Languages*, 14(4):263–281, 1989.
- [11] C. Ghezzi, D. Mandrioli, S. Morasca, and M. Pezzè. A Unified High-Level Petri Net Model For Time-Critical Systems. *IEEE Transactions on Software Engineering*, 17(2):160–172, February 1991.
- [12] C. Gordon, et al., Care Protocols and Healthcare Informatics. In *Artificial Intelligence in Medicine*, (S. Andreassen, et al. Eds.), IOS Press, 1993.
- [13] C. Gordon, I. Herbert, P. Johnson, P. Nicklin, D. Pitty and P. Reeves. Telematics for Clinical Guidelines : a Conceptual Modelling Approach. *Medical Informatics Europe '97 IOS press* 1997.
- [14] Institute of Medicine. *Guidelines for Clinical Practice*. National Academy Press, 1992.
- [15] K. Jensen. Coloured Petri Nets. In W. Reisig and G. Rozemberg, editors, *Advances in Petri Nets*, LNCS 254-255. Springer-Verlag, Berlin-New York, 1987.
- [16] C. Kronlöf, editor. *Method Integration - Concepts and Case Studies*. John Wiley & Sons, 1993.
- [17] E.B. Liem, J-S. Obeid, E.P. Shareck, L. Shato and R.A. Greenes. Representation of Clinical Practice Guidelines Through An Interactive World-Wide-Web Interface. *SCAMC* 19:223-27, 1995.
- [18] H.J. Lowe, E.C. Lomax and S.E. Polonkey. The World Wide Web : A Review of An Emerging Internet-Based Technology for the Distribution of Biomedical Information. *JAMIA* 3(1):1-14, 1996.
- [19] D. Luzi et al. Shared Care: Workflow Management for Medical Treatment Improvement.

Proc. of Health Telematics '95, 1995.

- [20] R. Medina-Mora et al. The Action Workflow Approach to Workflow Management Technology. In Proc. of CSCW'92 Conf., ACM Press, 1992.
- [21] T. Murata. Petri Nets: Properties, Analysis, and Applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.
- [22] M. Nagl. Set theoretic approaches to graph grammars. In H. Ehrig, M. Nagl, G. Rozenberg, and A. Rosenfeld, editors, *Graph Grammars and Their Application to Computer Science*, volume 291 of *Lecture Notes in Computer Science*, pages 41–54. Springer-Verlag, 1987.
- [23] NIH Consensus Conference Adjuvant Therapy For Patients With Colon and Rectal Cancer. *JAMA* 264(11):1444-1450, 1990.
- [24] D.E. Oliver, M.R. Barnes, G.O. Barnett, H.E. Chueh et al. InterMed: An Internet-Based Medical Collaboratory. AMIA Fall symposium 19:1023, 1995.
- [25] J.K. Ousterhout. TCL and the TK Toolkit. Professional Computing Series. Addison Wesley, 1993.
- [26] M. Pezzè and S. Silva. Cabernet User Manual. Technical Report 47-94, Politecnico di Milano, May 1994.
- [27] D.M. Pisanelli, F. Consorti and P. Merialdo. SMART: A System Supporting Medical Activities. In Real Time Medical Informatics Europe '97 IOS press 1997.
- [28] S.H. Woolf. Practice Guidelines, A New Reality in Medicine. Impact on Patient Care. *Arch.Int.Med* 153:2646-55, 1993.