# Supporting PCR-related activity with Implicit Culture

*Andrei Papliatseyeu*

Advisor: Prof. Enrico Blanzieri

Master of Science
Department of Information and Communication Technology
University of Trento
2007

# Abstract

Polymerase Chain Reaction (PCR) is a very common kind of experiment conducted in laboratories over the world. However, the procedure is highly sensitive to the choice of parameters and varies depending on the local practices of each laboratory.

In this paper, we present an analysis of the PCR-related activities in a biological laboratory and identify the information needs of the biologists performing PCR. Then, we produce a set of requirements to a system for PCR support and describe an architecture of such system using Implicit Culture concepts. Finally, we introduce a working prototype of the system, its design, implementation and evaluation results.

This work is the result of close collaboration with biologists, individual interviews and on-site observations in laboratory.

# Acknowledgements

I would like to thank:

- My supervisor Enrico Blanzieri for his patient and helpful guidance during the project;
- Marcello Sarini for the valuable advices and discussions;
- Claudio Moser and his colleagues for the precious information about internal organization of a biological laboratory and useful feedback about the project.

Finally, I am grateful to my parents for their endless support and encouragement.

# Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Andrei Papliatseyeu*)

# Table of Contents

# Introduction

Polymerase Chain Reaction (PCR) has been taking an increasingly important role since its invention in early 1980's by Kary Mullis [39]. Today it is used in many research areas and practical applications, such as genetic fingerprinting, paternity testing, detection of hereditary diseases, analysis of ancient DNAs, etc [46, 33].

PCR has been patented, commercialized and to a great extent automated, and there are efforts being made to represent it as a simple procedure [16, 7, 4]. However, the technique is still complicated, contradictory and often confusing and cannot be considered as a "cookbook recipe" [36, 29]. Even strict adherence to the protocols provided along with DNA kits does not guarantee the success [40]. A large number of alternative approaches, modifications and applications of PCR makes it virtually unique for each laboratory. As Jordan and Lynch pointed out:

> There is no set list of variables that describes the sources of variation: instead, countless local adaptations, tunings, and tweakings go into the development and standardization of the technique as it is integrated into particular medical, legal or disciplinary contexts [36].

Therefore, the experiments that succeeded in one laboratory are not guaranteed to succeed in another [29, 26]. Another reason for this often is inability of the researchers to describe their protocols in a comprehensive way [26].

Heavy dependence of the PCR implementations on the local practices makes the experiment problematic for new laboratory workers even if they have had a reasonable experience with it in another lab.

Surprisingly, only few works have been dedicated to support biologists during PCR experiments in the laboratory [14, 42, 18].

This project continues and extends the work of Sarini et al. [42] and addresses the needs of the biologists working in a laboratory performing PCR experiments.

The objectives of the project are the following. First of all, we want to acquire an understanding of the processes occurring in a biological laboratory and to find the ways in which the biologists performing PCR could benefit from information technology. The second objective is to develop a system for PCR-related activity support and use it to test validity of the implications inferred from the previous step. The final goal of the project is to evaluate applicability of the Implicit Culture (IC) approach [22] to the design of such systems.

The project has been done with close cooperation with biologists of the genomics laboratory of the Instituto Agrario di San Michele all'Adige, Italy. The methodological approach of the work has been based on regular site visits to the laboratory combined with collective conversations as wel as individual interviews with the biologists.

This paper presents our findings concerning support of PCR experiments using Implicit Culture approach. We have provided an analysis of the domain of PCR-related activities in a biological laboratory. Then, basing on this analysis, we produced a set of requirements to a system for PCR support and described the architecture of such system using IC concepts. The system makes the personal knowledge of each biologist explicit and available for others by inferring patterns from the history of the actions observed in the laboratory. Finally, we have implemented a prototype of the system and evaluated it with biologists.

The results of prototype evaluation support the hypothesis about applicability of the IC approach to the biological laboratory work domain. They has also discovered the need of the biologists for more advanced algorithms for accurate prediction of the experimental parameter values.

The paper is organized as follows. The first chapter provides an overview of the background material. The next chapter presents activity analysis, requirements and the architecture of the system for PCR support. Finally, two last chapters contain the prototype design, implementation details, evaluation results and their discussion.

# Chapter 1

# Background material

This chapter sets the context of the present work by providing the necessary background information.

## 1.1 Polymerase Chain Reaction

Polymerase Chain Reaction (PCR) is a DNA replication method of molecular biology and biochemistry [46]. It has been invented by Kary Mullis in 1986 [39]. The reaction turned out to be so easy and much cheaper than previous methods (like chemical synthesis) that the inventor was rewarded with a Nobel prize in chemistry in 1993 [46].

PCR is a very common procedure conducted in biological and medical laboratories over the world. It is used for various tasks, such as genetic fingerprinting, gene cloning, paternity testing, etc [46]. The number of PCR-related publications since the invention (figure 1.1) proves the popularity of the procedure.

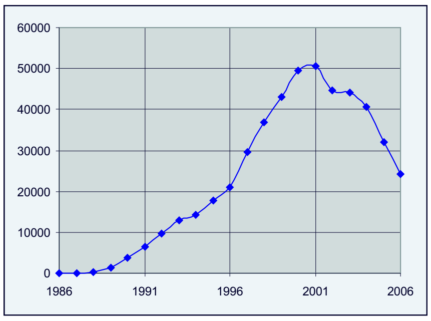


Figure 1.1: Number of PCR-related publications per year (estimated by Google Scholar [1])

The aim of PCR is amplification (replication) of a part of DNA, i.e. given a few samples of DNA, one may apply PCR to receive multiple copies of a specified region of the source DNA. The process is exponential and thus very fast.

The main components of PCR are [46]:

**DNA template** containing the region to be replicated.

**Primers** (one or more) which are relatively short (usually 18 to 25 base pairs) nucleotide sequences.

**Polymerase** which synthesizes the DNA copies. This may be Taq polymerase or other DNA polymerase.

**Deoxynucleotide triphosphates (dNTPs)** as a building material for the new DNA.

**Buffer solution** providing a suitable chemical environment for the DNA polymerase.

The PCR is performed within special machines called thermal cyclers or thermocyclers. These are used to heat and cool the reaction tubes to the temperatures defined by the appropriate experiment program. Usually, PCR contains about 20 to 35 repeated cycles. Most commonly, procedure for PCR is performed as follows [46].

1. Initially, the reaction is preheated to a high temperature (94–96°C) which is held for 1–9 minutes. Thus one insures that most of the primers and templates have denaturated. After this, the temperature cycling begins.

2. The temperature is hold at 94–96°C for 20–30 seconds (denaturation step)

3. At this step (annealing) the temperature is lowered, and the primers attach to the single strands of DNA template. The temperature value is usually chosen between 50–64°C (depending on the $T_m$ of the primers, see below) and hold for 20–40 seconds.

4. At this step (expansion), the polymerase copies the DNA strands, starting from the annealed primers. The temperature depends on the used polymerase and usually lies around 72°C. The timing of this step depends on the polymerase and the length of the region being replicated. An empirical estimate for Taq polymerase is 1000 base pairs per minute.

5. At the last step, expansion is prolonged for up to 15 minutes in order to ensure that all remaining single-stranded DNA are copied.

The overall success of the experiment highly depends on the right choice of temperature values, especially the annealing temperature, which depends on the melting temperature $T_m$ of the primers (usually it is about 5°C lower that $T_m$).

The $T_m$ of a primer is defined as the temperature at which half of the primer binding sites at the DNA template are occupied [46]. $T_m$ increases with the length of the primer.

However, shorter primers lack specificity and may anneal at several positions on the template, thus causing non-specific amplification. Very long primers, in turn, require very high annealing temperature (over 80°C), which affects the polymerase. Therefore, an optimal length of primer is about 15–40 nucleotides, with annealing temperature from 50°C to 75°C.

For calculation of the melting temperature of primers, a number of empirical formulas and rules-of-thumb are used. Two of them are [27]:

$$T_m^\circ\mathrm{C} \simeq 2(C+G)+4(A+T) \tag{1.1}$$

$$T_m\ ^\circ\mathrm{C} \simeq 64.9+41\frac{C+G-16.4}{N} \tag{1.2}$$

where letters *C,G,A,T* represent the number of the corresponding nucleotides in the primer's sequence; *N* is the length of the primer. The first formula is valid for primers with up to 14 nucleotides, the second is used for longer primers.

However, none of such formulas provides better than $\pm 5\ldots 10$°C $(8\ldots 17\%)$ accuracy [25] and the proper values of the melting temperatures of primers and annealing temperature for the PCR must be chosen empirically for specific conditions. Also, due to the large number of other parameters influencing PCR, the software tools for primer design (like Primer3 [6]) can only provide approximate values, and it is generally suggested to determine optimum conditions for the specific environment empirically [46].

## 1.2 Implicit Culture and SICS

When a person comes to a new and unfamiliar environment, it is sometimes quite difficult for him/her to make the appropriate choices and decisions, and to act consistently with the environment. Thus, the novices' behavior is not optimal. However, this is not necessarily the case for the people who have an experience of similar situations, as they have obtained some knowledge related to the effective behavior in the environment. Such knowledge is generally implicit and corresponds to what can be called "community culture" [20].

Implicit Culture (IC) is based on the assumption that it is possible to elicit the community culture by observing the interactions of people with the environment and to encourage the newcomer(s) to behave similarly to more experienced people. Implicit Culture assumes that agents perform actions on objects in the environment (see [23] for more details). The actions are considered in the context of situations, so agents perform situated actions. The “culture” contains information about actions and their relation to situations, namely which actions are usually taken by the observed group and in which situations. This information is then used to provide newcomers with information about the others behavior in similar situations. When newcomers start to behave similarly to the community culture, it means that we have a knowledge transfer. “Implicit Culture is a relation between a set and a group of agents such that the elements of the set behave according to the culture of the group” [23]. This transfer of knowledge is performed by a SICS [23].

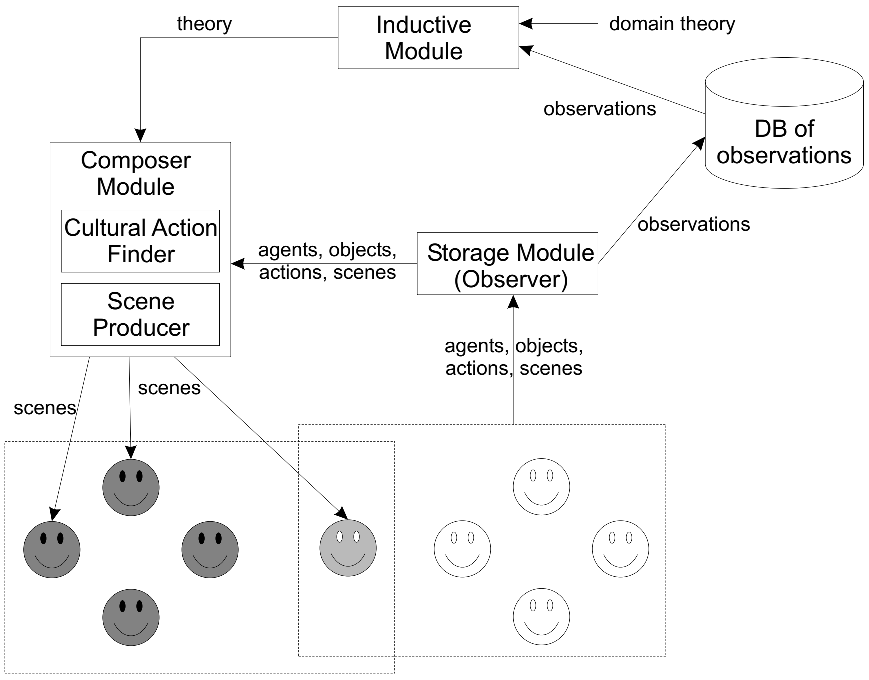


Figure 1.2: The general architecture of the System for Implicit Culture Support [20]

The general architecture of the SICS is depicted in Figure 1.2. The SICS consists of three components: the Observer, which uses a database of observations to store information about actions performed by users in different situations; the Inductive Module, which analyzes the stored observations and applies data mining techniques to find a theory about the community culture; the Composer Module, which exploits the observations and the theory in order to suggest actions in a given situation. The Composer Module operates with the architectural abstraction of the situation - the scene, which contains a set of objects and a set of actions that can be performed on these objects. The Composer Module includes the following two sub-modules: the Cultural Action Finder(CAF), which filters those actions that satisfy the theory; and the Scenes Producer, which searches for scenes where the actions (found by the CAF) are likely to be performed.

Figure 1.2 shows that the theory (representing the community culture) is extracted from the observations on the set of agents and is then used to produce recommendations for another set of agents. These two sets may be disjoint, may overlap or coincide. The theory is rule-based and contains two parts. A part of the theory used by the Composer Module can be specified a priori (domain theory), while the other part is learned by the Inductive Module and can evolve over time. For instance, for the general problem of providing people with recommendations the domain theory may say that the system must recommend items which are likely to be accepted by the user, while the theory learned by the Inductive Module may contain information about which items are accepted.

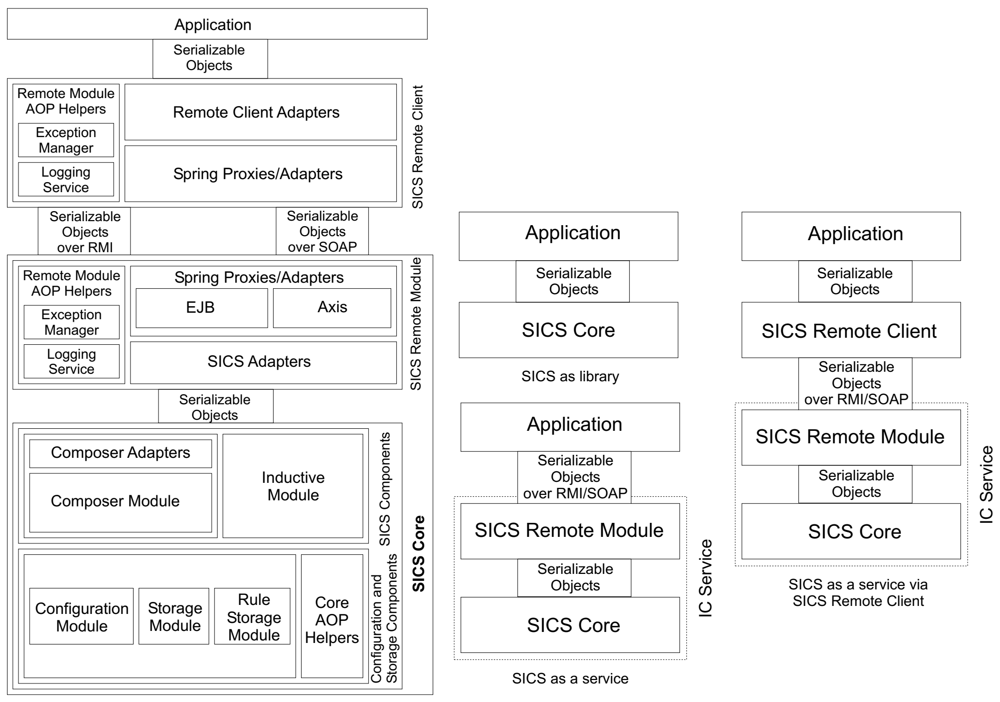
As an example of Implicit Culture, one can consider a person who would like to use a book-selling service, but does not know which service is used by other people in the same location. Obviously, people who used to sell books know the name of the service and it may be the case that they have tried several services before choosing the best one. If the system is able to use previous history to suggest that the person access the service used by others and he/she actually does it, then it is possible to say that he/she behaves in accordance with the community culture and that the Implicit Culture relation is established.

The SICS core functionality is implemented in a form of a generic library [20]. The detailed architecture of the SICS is presented on figure 1.3(a).

An application may utilize the SICS functionality in three ways: as a library; as a web-service; as a web-service via SICS remote client (figure 1.3(b)). The first one is suitable for the applications that are not distributed and allow tight-coupling with the library. It also provides best performance, as there are no losses on serialization, transfer and reconstruction of the objects. Distributed applications may use SICS either as an ordinary web service via SOAP protocol or via SICS Remote Client. The second case is simpler, as Remote Client hides technical details of communication, but it is also more resource demanding [20].

## 1.3 Software tools for PCR

This section presents an overview of three different systems for augmenting biological laboratory by means of information technology, with either explicit or possible relation to PCR.



(a) SICS Architecture (b) Invocation scenarios

Figure 1.3: SICS architecture details and invocation scenarios [20]

## 1.3.1 Labscape

Labscape [15, 14] is a pioneering project aimed to support biologists in a laboratory by converting it into a smart environment. Its goal, as the authors state, "was to simplify laboratory work by making information available where it is needed and by collecting and organizing data where and when it is created into a formal representation that others can understand and process" [14].

The authors have performed a thorough analysis of the application domain; many consequent projects, including the present one, have been using the results of this analysis. The project design approach was the user-centered one, and included extensive observations of the laboratory environment, analysis of long video recorded in the laboratory, interviewing biologists, etc.

The design of the system has taken into consideration the following observations [14]:

- Biologist's tasks during an experiment involve both mental and physical activity.
- The biologist may need information support in various areas of environment.

- Laboratory work requires interaction with different kinds of equipment.
- Biologists perform a limited types of operations, but in a variety of ways.

The design guidelines included the requirements of compatibility with almost any biology laboratory, i.e. reliance on a simple equipment, and sufficiency of interaction only with basic human-computer interfaces like mouse, keyboard or touch screen.

The authors have grouped the information needs of the biologists according to the three phases of workflow:

**Preparation** In this phase, the biologist prepares a plan of the experiment, usually basing on a previously completed one and recording only differences between two. Labscape provides such functionality, but in contrast with the paper-based system provides a complete experiment plan.

**Execution** During this phase, the system must:

- Provide the biologist with a plan of experiment. Labscape provides an interface to a database and represents the plan in a comprehensible form of a graph (figure 1.4).
- Support some means for keeping track of progress. With Labscape, the user records the progress by simply touching the appropriate elements of the plan. The system also allows the user to choose own level of details.
- Record data and observations during the experiment. Labscape supports various types of captured data (text, images, audio, video) and allows the user to attach captured data to the appropriate components of the experiment plan.

**Documentation** This phase may occur only several days after the previous one. In this phase the biologist makes a formal record to the personal notebook. For paper-based process, some details may be omitted as unimportant; however, Labscape records of the experiment may themselves be a useful part of the report.

While Labscape makes emphasis on the explicitness of user interface, the computing infrastructure is hidden and biologists do not need to be aware about technical details.

The system has been evaluated in a number of laboratories with different settings [14, 13] and proved itself valid as a smart environment for a cell biology laboratory. Currently, Labscape has become a commercial product offered by Teranode [8].

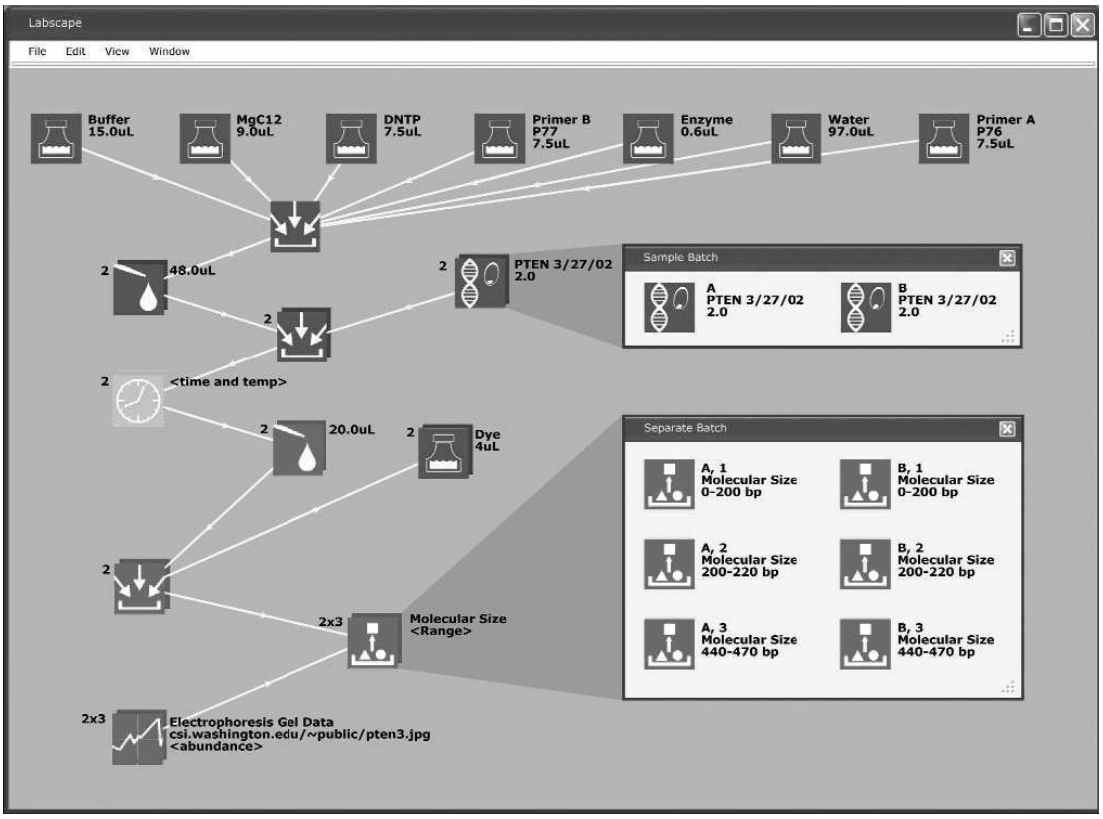

Labscape
File
Edit
View
Window
Buffer 15.0uL
MgC12 9.0uL
DNTP 7.5uL
Primer B P77 7.5uL
Enzyme 0.6uL
Water 97.0uL
Primer A P76 7.5uL
2 48.0uL
2 PTEN 3/27/02 2.0
Sample Batch
A PTEN 3/27/02 2.0
B PTEN 3/27/02 2.0
2
2 <time and temp>
2 20.0uL
2 Dye 4uL
Separate Batch
A, 1 Molecular Size 0-200 bp
B, 1 Molecular Size 0-200 bp
A, 2 Molecular Size 200-220 bp
B, 2 Molecular Size 200-220 bp
A, 3 Molecular Size 440-470 bp
B, 3 Molecular Size 440-470 bp
2
2x3 Molecular Size <Range>
2x3 Electrophoresis Gel Data csi.washington.edu/~public/pten3.jpg <abundance>

Figure 1.4: Labscape user interface [14]

### 1.3.2 BioSICS

BioSICS project [42] uses the concepts of Implicit Culture (section 1.2) to provide biologists with suggestions which may help them in their work.

Basing on the observations of biologists working in laboratory and the experience of one of them, the authors have described the environment of the biological laboratory and the artifacts used by the biologists. A special attention has been dedicated to laboratory notebooks as one of the most important artifact the biologists use in their work.

BioSICS project aimed at the satisfaction of the information needs of biologists. The following groups of information needs have been identified [42].

**How** to perform an experiment or its specific action? These needs are caused by variations of personal experience among the biologists.

**Where** to find needed materials? Different reagents are stored in different places and often there is a problem to find required reagent, especially for a new member of the laboratory.

**Why** has a specific experiment succeed or failed? This corresponds to the motivations of the actions to be taken.

**When** a particular required machine is available? This is related to the efficient use of shared equipment.

The proposed solution has been a combination of situation-based and history-based recommendation system. The main idea was to store and combine past actions of the biologists in order to infer suggestions for the current situation. The actions were to be observed indirectly via personal notebooks augmented with some of digital input technologies, like Anoto [32] or Paper++ [5], or via electronic versions of these notebooks.

BioSICS system has been based on the SICS architecture and contained four main modules (figure 1.5):

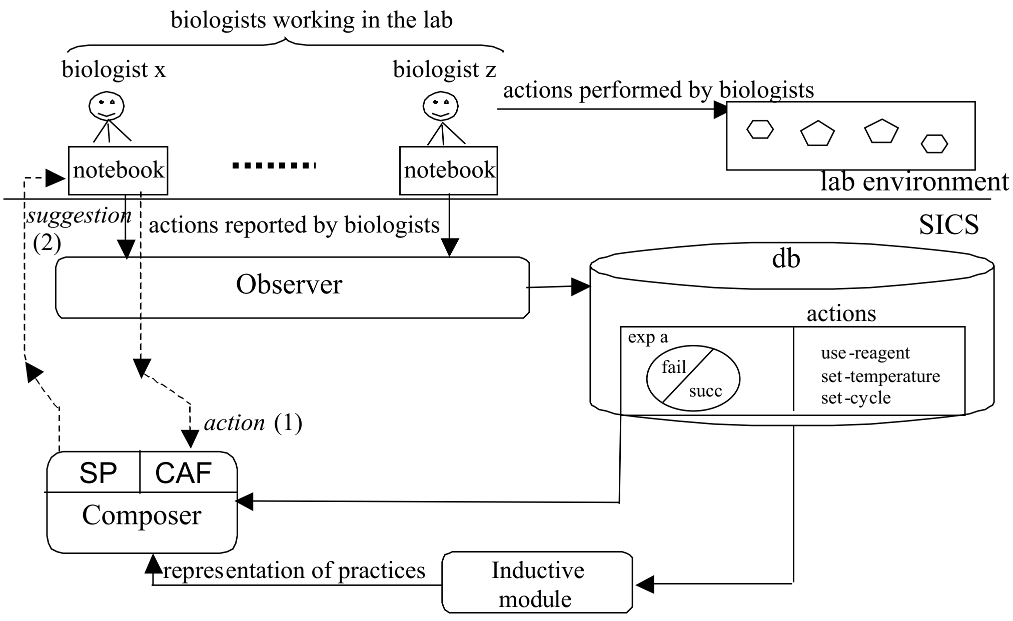


Figure 1.5: BioSICS architecture [42]

**Observer** This module uses the notebooks in order to collect the data about the actions performed by the biologists during experiments and puts them into the database.

**Database** contains the history of the actions taken during the experiments and specifies which of the experiments were successful.

**Inductive module** It uses data mining algorithms and, basing on the stored observations, creates a set of rules which make up the cultural theory for the group.

These rules are expressed in the first-order form:

$$a_1 \wedge a_2 \wedge \ldots \wedge a_M \rightarrow a_N$$

Thus, if the scientist who asks for a suggestion "what should I do next?" has already performed actions $a_1, \ldots a_M$ – the system will use the above rule and suggest the action $a_N$.

**Composer** This module fills the template rules provided by IM with some real parameters (e.g. the actual temperature value in the `set-temperature` action) from the similar actions performed before and provides the complete suggestion (i.e. the most probable, according to the cultural theory, action with fully specified parameter values) to the user.

The authors have proposed a number of possible algorithms to use in the Inductive Module. The main one was Apriori [11] by Agrawal and Srikant. However, Apriori suffers from so-called "cold start" problem, because it requires a large amount of data which is not the case when the system has just been installed and only few observations have been stored. To alleviate the "cold start" problem the authors suggested to use FOIL [41] data mining algorithm, which can work with small amounts of data.

To summarize, BioSICS project has addressed the problem of making the data kept in personal notebooks both available and usable by providing the biologists with suggestions about current experiment basing on the colleagues' experience with the past ones.

### 1.3.3 Virtual Lab Dashboard

Virtual Lab Dashboard [18] project focuses on building an infrastructure for the Smart Biological Laboratory of the future.

The authors have addressed the goal of creation of a pervasive laboratory information management system (LIMS) by means of providing the users (biologists) with a tool for remote and local monitoring and control of the laboratory equipment.

The project is focused on the two issues [18]:

**Heterogeneous lab instruments** The equipment used in a laboratory is usually from different vendors and utilizes different communication and control protocols. Therefore, the system should integrate different hardware platforms into a unified one.

**Mobile scientists** The biologists do not usually stay at one site in the laboratory, it is more common to change location frequently. Therefore, the system must provide such biologists with access to the (relatively) remote equipment.

To solve the first issue, the authors have build a dedicated hardware, called Bio-Bridge Device which purpose was to connect various lab equipment to a wireless ad hoc network and to provide an interface to access bridged devices remotely in a uniform fashion.

The architecture of the system is presented on figure 1.6. As one may see, the

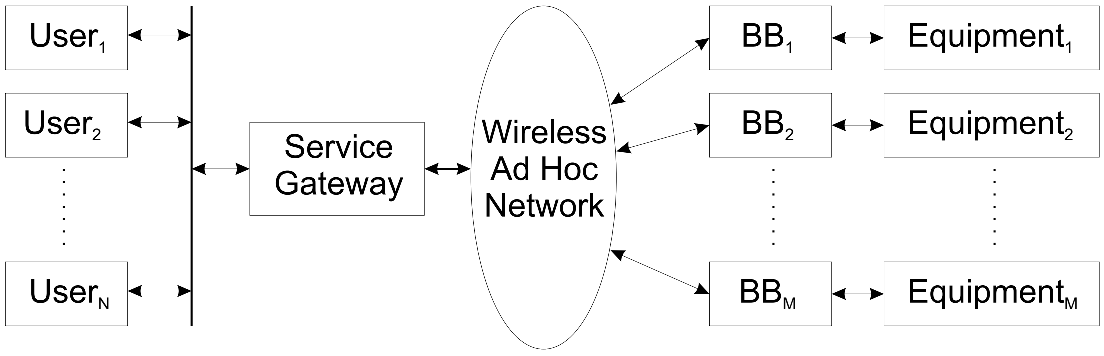


Figure 1.6: Virtual Lab Dashboard system architecture [18]

users can access the equipment through the service gateway which is represented by a wireless LAN card, a PC running Apache web server, and a GSM modem. The gateway provides two alternative ways of working with the system: via web interface or via mobile Short Messaging Service (SMS).

The authors have also considered the problem of locating the machine the user is near to. The proposed approach was to use signal strength patterns of IEEE 801.11b wireless LAN to infer user's location, as described in [17]. That would make the system context-aware.

Although the project primarily concerns with the design of the embedded system and lacks comprehensive study of the laboratory domain and real-life evaluation, the proposed system seems to be a good middleware solution for a laboratory management system.

## 1.4 Summary

In this section we have outlined the principles of PCR and Implicit Culture, and briefly described the existing systems related to PCR experiment support. The present project has a number of distinctive features compared to the others.

First of all, in comparison with the ancestor BioSICS project [42], the present work delivers a number of novelties and enhancements. We have performed our own analysis of the domain and information needs of biologists and elaborated a detailed set of requirements to the system for PCR support. We have also designed an integral architecture of such system, which extends the one presented by Sarini et al. [42]. Finally, we have designed and implemented a working prototype of the system and performed a qualitative evaluation of it, while BioSICS project was purely theoretical.

What concerns Labscape [14], there are two main distinctions.

First, we focus on the collection of system input data via transparent and non-intrusive observation of users' actions from a possibly large number of different sources, while Labscape requires direct interaction with computer. It is more probable that the biologists would easier accept a system that does not interfere with the existing practices.

Second, Labscape provides a visual tool to prepare, conduct and record experiments. However, the experiments' templates must be predefined in advance. The system we propose takes an advantage of the collaboration principles: the experiment templates are induced from the history of observations and thus it is colleagues' experience that is used to infer suggestions. Thus, the system will automatically and dynamically adapt to the local rules of the laboratory.

Virtual Lab Dashboard [18] is more concerned with the practical aspects of building a smart bio-laboratory environment and issues related to remote monitoring and control of such environment. However, it lacks a study of the domain and evaluation by prospective users. In turn, our work has been performed with close interaction with biologists, with particular focus on PCR experiments and features both domain analysis and prototype evaluation.

# Chapter 2

# System for PCR support

This chapter presents an analysis of the domain, requirements and design of the system for PCR support.

## 2.1 Domain analysis

### 2.1.1 Methodology

From the methodological point of view, the work has been done with a composite approach: active collaboration with biologists in form of interviews and site visits to Instituto Agrario di San Michele all'Adige, Italy during the whole time of the project combined with extensive literature review on the subject.

The initial idea of enriching the work of biologists with computer support has primarily been discussed in a private manner with Claudio Moser, the head of the Genetics & Bioinformatics research unit of the aforementioned institute. As the idea seemed to be sound, the authors undertook the first site visit to the laboratory of genomics, in order to get a more detailed personal view of the laboratory's internal organization than it is possible to grasp from the relevant literature.

Site visits to the laboratory were accompanied with naturalistic observation of the work of biologists in their usual environment. Although the authors attempted to be as unobtrusive as possible, some influence of their presence might have caused variations in a usual behavior of the laboratory workers. However, this is an inevitable pitfall of the naturalistic observation method applied in laboratory [45].

The summary of our observations and interviews taken are presented below in section 2.1.2.

After the development of the first prototype, the classical iterative development scheme has been followed: evaluation of the prototype with biologists; improvements based on the acquired feedback; re-evaluation, etc. All the communications with biologists have been performed on site in their working environment, either in the laboratory or office.

After implementation of the most interesting functions, the final prototype evaluation has been done, its results are presented in the section 4.4.

### 2.1.2 Activity analysis

#### 2.1.2.1 Laboratory layout

The lab is represented by two adjacent rooms, one of which contains long tables with working places and shelves, refrigerators with reagents and smaller pieces of equipment while the other one is full of large and noisy machines and is visited only when there is a need to check or use one of them. In the lab, there are three computers connected to the LAN.

In total, there are six working places, plus those in front of computers (these are shared). Both rooms of the laboratory have a large number of PostIt notes hung above working places and on the machines. They are also used to manage usage of the shared equipment (reservation, current task description, etc).

There are various kinds of machines and tools used in the laboratory, including a number of thermocyclers, machine for electrophoresis (with digital camera over it), centrifuge, thermostats, scales, timers, chambers, pipettes, tubes, 8-tubes strips, 96-well plates for PCR, other laboratory glassware.

Reagents are stored on the personal shelves in refrigerator. There is also a common stock, managed by a responsible person. When somebody needs a specific reagent, he/she requests it from the person who manages the stock.

#### 2.1.2.2 Personnel

There are different kinds of staff working in the laboratory:

**Senior researchers** determine the directions of the work in the laboratory, assign tasks to technicians and students. However, most of the time they spend in their offices instead of laboratory.

**Technicians** perform general routine experimental work according to researchers' assignments. Usually it is they who conduct PCR experiments.

**Students, PhDs and postdocs** carry out their own research in the laboratory, acquiring or further deepening the important hands-on experience about the procedures of experimental biology.

There is no evident general distinction between experts and plain biologists in the laboratory; every member of the staff has keen knowledge in his/her specific areas.

#### 2.1.2.3 Notepads

The results of the experiment are stored in the personal notebook of the biologist in a free form. Some of the results are hand written, some are printed from a computer and glued in (e.g., tables, photos of gel electrophoresis, etc). There is also an established report form with predefined fields (see also [42]).

A typical PCR experiment report contains a description of the reagents usage details (concentrations and amounts), a pattern of DNA layout in the wells, the thermocycler's program, a photo of the gel electrophoresis result with hand-made marks, an interpretation of this result (e.g., the specificity of the product), other personal notes related to the experiment.

Generally, notepads are used to store only the results of the experiments, but not the progress.

#### 2.1.2.4 PCR timing and pricing

As the project is particularly focused on the PCR, we have interviewed the biologists on the details of the organization of this experiment.

The following is an approximate example of a PCR timing:

- Primers design – 30 minutes
- Ordering of primers – few days
- PCR itself
  - Thermocycling – 2–3 hours
  - Gel electrophoresis – 30–60 minutes, depending on the size of the product
- Writing a report – 30 minutes

Thus, an experiment may take from few hours (with ready primers) to few days (including primers order and delivery).

A typical PCR reaction costs approximately \$1.0–1.5 per sample (excluding primers) [9]. Primer synthesis is about €0.4–0.5 per base pair. Given a usual total length of 40 base pairs (both forward and reverse primer) this makes €20. The amount of synthesized primers is usually enough for approximately $10^4$ reactions.

### 2.1.3 Information needs

We have identified the following information needs of the biologists.

#### 2.1.3.1 History of experiments

The results of the experiments are recorded in the personal notebooks, which are hard to read for everybody but the owner. This makes it difficult to utilize the experience of a colleague who could probably has already done some experiments similar to the one being planned. Moreover, with a collection of separate notebooks it is difficult to keep a common history of experiments.

Currently, the laboratory in which we conducted our on-site research does not have any common database on PCR experiment results. We faced this fact ourselves when we asked for some real experimental data for our prototype (section 3). It took several phone calls and a number of days to find some data. It was in the form of Excel spreadsheet compiled by one of the biologists for individual use.

#### 2.1.3.2 Choice of experimental parameters

Another need identified during interviews, is the difficulty of selection of the appropriate experimental parameters. There are some rules-of-thumb used in the selection process, but they are not very reliable. The researchers usually start from some default values and if they do not work out, the experimenters have to change them slightly, sorting out the proper ones. Unfortunately, after the correct parameters have been found and the experiment succeeded, the parameter values usually do not become known to other lab members, but are put away in the personal notebook instead.

#### 2.1.3.3 Primer design

For PCR experiment, primer design is one of the key processes. Experiment's success largely depends on the designed primer. There are some tools available for this task (e.g. Primer3 [6]), but they are complex to use and dispersed. During the prototype development, the biologists asked us to add a simple tool for evaluation of melting temperature for arbitrary primers.

Thus, there is a need for integration of external services available online and providing them possibly with a more convenient interface.

#### 2.1.3.4 Shared resources management

Biological laboratory is an environment where many machines are shared among all the lab members. This causes need for synchronization of activities and reservation of the machine time. Currently it is being done by means of PostIt stickers on the machines. However, they tend to drop and cannot give a notification when the machine becomes available.

The same is valid for shared reagents. Once somebody has used a reagent, he or she does not necessary put it back from where it has been taken. This raises a problem of location of a specific reagent in the laboratory (see also [42]).

Summarizing the above, there is a need for efficient scheduling and management of shared resources.

## 2.2 Requirements analysis

### 2.2.1 Goals of the system

The system should address information needs of the biologists performing PCR experiments. The following design goals have been set:

- non-intrusively keep track of the progress;
- give the experimenter a plan to follow;
- provide suggestions in complex situations;
- record experimental data and observations;
- manage shared resources for effective use;
- integrate with external services.

## 2.2.2 General requirements

As the system is to be used by a group of collaborating people, the design should follow both the general requirements for a Computer Supported Cooperative Work (CSCW) system and the specific requirements for the biological laboratory and PCR experiment support.

A biological laboratory obviously has a clear spatial structure, thus the CSCW system must be space-aware. However, the processes in the laboratory follow some either explicit or implicit protocols. Experiments themselves are continuous in the time. Therefore, the system for experiment support must also have a notion of workflow and be context-aware.

The basic requirements to CSCW systems are listed below.

**Awareness** Users should be aware of the activities of the others. In the context of a biological laboratory, it is required for shared resource (machines) usage.

**Flexibility** The system should support various working styles and easily adapt the changes of these styles through the time.

**Access control** The system should provide different access rights for different groups of users. This implies creation of a user profile management subsystem, and distinction of the user role when providing an access to the system functions.

**Availability** A user should be able to access the system in any place of the laboratory where he/she needs to. Also, the system must be able to track the user when he/she moves from table to table, which is very common for the researchers in the biological laboratory.

Bogia and Kaplan [24] have also approached the problem of maintaining dynamically-changing, flexible workflows. Here are some of the design goals they have used [24]:

**Support for a range of specifications** One should be able to define the workflow at an arbitrary level of detail.

**Support for modifications** One should be able to modify workflow specification after deployment.

**Inheritance of existing definitions** One should be able to define new workflows, using previous as a template.

**Record of history** Actions performed with or by the system should be recorded and logged for possible future review (see also [38]).

### 2.2.3 Functional requirements

The functional requirements for the system conveniently fall into a number of groups in accordance with the design goals. In this section we present a list of these requirements and the respective design goals they address.

#### Goal: non-intrusively keep track of the progress

- In order to be able to perform its functions, the system should continuously observe the actions done by the biologists in the laboratory environment. The actions should be observed from a number of different sources, possibly with some redundancy, so that the system can get the details of each action.
- Some of the experiments do not have a straight "success/fail" outcomes. Therefore, the system must have a notion of "degree of successfulness" of the experiment and use it in the processes of suggestions inference.
- The experimenter might find some of the previous actions to be erroneous and want to retry the experiment from some previous state. The system must support such a revision function.

#### Goal: give the experimenter a plan to follow

This and the following goal correspond to the "Choice of experimental parameters" information need of biologists (section 2.1.3.2).

- The system must provide suggestions on the experiment workflow, proposing the experimenter the actions that can be taken next in the current situation.

  A discussion with the biologists has shown that the sequence of actions to take during PCR is usually known by the scientists, even newcomers. However, as Jordan and Lynch [36] state, PCR is a very lab-specific procedure with a wide variety of local practices. Thus, the function of suggesting the user the next action to take seems to be appropriate.
- The suggested actions should be displayed with their respective contexts, like details of the situation that the action has been taken in and the name of the person

who has done this action. The importance of providing a context for recommendation systems has previously been emphasized by Ackerman and McDonald [10].

- An experimenter should also be aware of the progress of his/her experiment. Thus, the system must display the current experiment's history so far, so that the user may estimate the current state of the procedure.

**Goal: provide suggestions in complex situations**

- Information needs of biologists include the problem of correct selection of experimental parameters, like reagents concentrations, temperature values, etc. Therefore, the system must also provide suggestions concerning the parameter values of the actions it recommends.
- For these suggestions, the context should also be available, as we have already stated for the workflow-related recommendations.

**Goal: record experimental data and observations**

This goal corresponds to the "History of experiments" information need (section 2.1.3.1).

- All the experiment-related observations should be saved in a persistent storage. These stored observations enable experiment history viewing and suggestion providing functionality of the system.
- The system should enable the user to give a textual description for the experiment.
- As biological experiments may be lengthy (from hours to months), the system should support possibility to conduct such a long-term experiments involving multiple participants.
- The system must provide an access to the history of saved experiments, with sufficient search/filtering ability and adjustable level of details.
- Taking into account the important role of experiment reports in biological research, the system should be able to generate a printable report about selected experiment, using local formatting templates of the laboratory.

**Goal: manage shared resources for effective use**

This goal corresponds to the “Shared resources management” need (section 2.1.3.4).

- In order to facilitate effective use of the shared resources and equipment, the system should provide means to enable users awareness about current state and reservations schedule of the equipment.
- The system should provide the users with the hints concerning the location of specific reagents or tools in the laboratory, basing on the recorded observations.
- Taking into account the increasing amount of computer-controlled laboratory equipment, the system should support automatic operation of such equipment (for example, automatic loading and running of a thermocycler program or shutting down the machine when it is neither used nor reserved).
- When the system observes a machine breakdown, it should automatically send an appropriate notification to the technical support.

**Goal: integrate with external services**

- In order to enhance its functionality, the system should integrate with the available external services, such as sequence databases, primer design tools, sequence matching services, etc. This requirement also addresses the “Primer design” need (section 2.1.3.3).
- Due to the fact that the biologists use various types of data (images, spreadsheet, etc), the system must support integration of such data with the history of experiments. For example, a record about PCR experiment might have a gel electrophoresis photo attached.
- Whenever the system detects a shortage of some reagent or equipment, it should send an appropriate notification to the person responsible for shared reagents' stack management.
- If the laboratory already uses some kind of Laboratory Information Management System (LIMS), the proposed system should also integrate with the installed LIMS, if possible. This would allow both systems to benefit from information exchange and sharing of data.

**PCR-specific functional requirements**

The following requirements are related to the specifics of PCR procedure. The system should:

- take into account similar primers, when producing suggestions. However, it should only consider highly similar ones (for details see section 3.3.5).
- consider the experiment type (e.g., SNP or Microsatellites);
- support ability to work with a single primer (provide a list of primers that have previously been used with the current one);
- observe reagents' concentrations;
- store at least the following information about the experiment:
  - Gene name;
  - Primer name;
  - Primer sequence;
  - Author name.

### 2.2.4 Interaction requirements

This section describes important requirements regarding the user interaction with the system (see also section 3.3.6).

**Transparency** The system must not intrude in the existing workflow, nor require continuous *direct* interaction with users.

One solution is to observe the machines the user interacts with anyway. There is a problem of few machines having an interface to computer. However, the tendency is that more and more machines are to be produced with such interfaces. Another solution is the integration of input from different sources (Anoto paper [32], touchscreens, RFIDs, barcode scanners, cameras, keyboards).

**Easy and intuitive user interface** This continues the previous requirement of transparency. Biologists need to concentrate on their work and not on the interaction with the system. Moreover, the possibility of various types of client devices must be considered. Therefore, the interface must be simple, have shortcuts for the most common tasks and be adaptable for different kinds of output devices (displays, data projectors, PDAs, cell phones, pagers, etc).

### 2.2.5 Non-functional requirements

For the CSCW in biological laboratory the above should be extended by the following non-functional requirements.

**Minimal cost of suggestion** As Ackerman and McDonald pointed out in the requirements for Answer Garden recommendation system, “Cost of authoring must be minimized” [10]. Ideally, the system should provide suggestions without explicit “question answering” by users.

**Extendability** Some laboratories might need additional functionality from the system. Thus, there should be a possibility to add new modules to the system and configure the existing ones (e.g., modification of the list of possible actions and parameters).

**Portability** The system should be easily portable in order to use it with different hardware platforms and operating systems.

**Easy localization** As the system is supposed to be used in different laboratories, it should be easily adaptable to the local practices of any laboratory.

**Openness of the architecture** The system should have an open architecture and be easily integrable with other systems [28]. This is achieved by adaptation of the service-oriented architecture and interaction via SOAP protocol.

## 2.3 System architecture

The general architecture of the system is presented on figure 2.1. Its main components and their functional roles are described below.

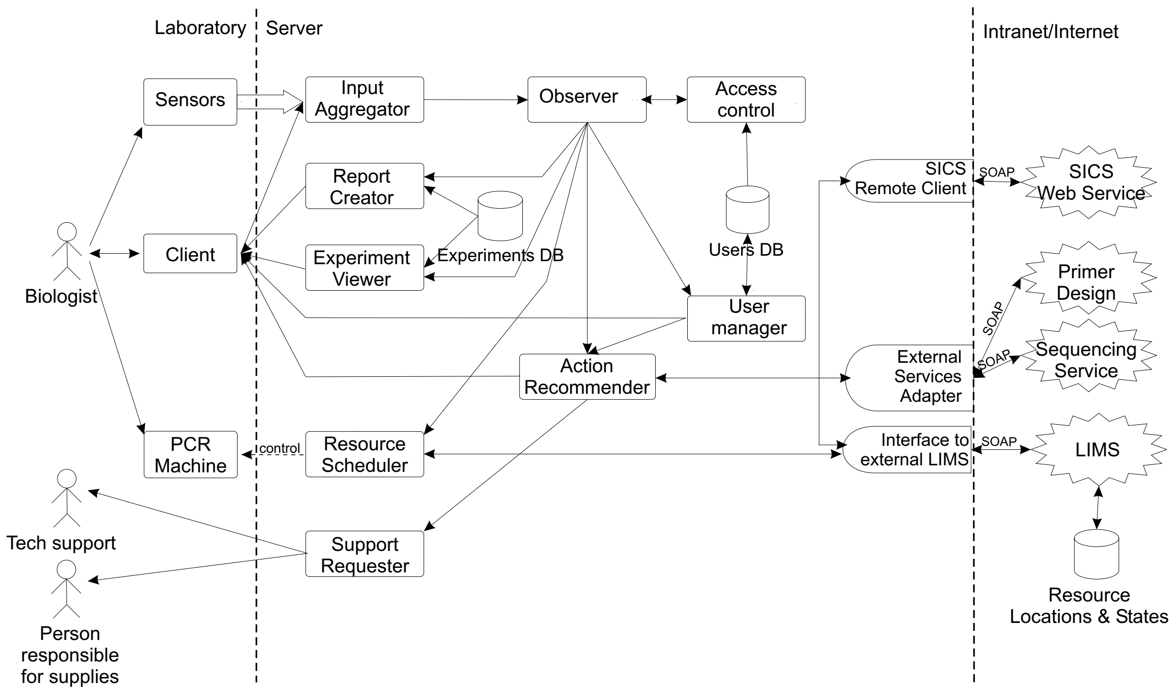


Figure 2.1: System architecture

### 2.3.1 Components of the system

#### 2.3.1.1 Sensors and Input Aggregator

Sensors in our architecture are all devices that provide the system with the information about environment's state, its changes over time, user actions and locations. They include, but are not limited to: temperature, light, pressure, resistance, radiation, sound, video, still images, acceleration, humidity sensors, RFID receivers, microphones, cameras, keyboards, touchscreens and other means of human-computer interaction (like Anoto paper [32] or augmented notebooks [37]).

Obviously, such a wide variety of sensors raise the problem of unification and integration of the information gathered from them. A sensible solution is to adapt the layered architecture for context-aware systems presented by Ailisto et al. [12]. The layered architecture provides high flexibility and extensibility of the input infrastructure.

There are five layers in the framework [12]:

**Sensors.** There can be three types of sensors [35]:

> **Physical sensors** are sensors in their usual understanding, i.e. hardware for determining the value or magnitude of a physical quantity or variable.
>
> **Virtual sensors** obtain the information from software systems. For example, such a sensor may report user's interaction with external LIMS.
>
> **Logical sensors** provide higher-level information by aggregating data from a number of physical and logical sensors, enriched with additional information from databases. For example, such a sensor might provide the information whether the machine the user wants to use is free, by combining information from the sensors of user location and database of machines usage.

**Raw data retrieval.** This layer contains drivers for physical sensors and libraries for logical and virtual ones. It converts raw data from the sensors to a standard format (E.g., it returns a temperature value irrespectively of the way it has been measured - whether it is an infrared photodiode or a thermocouple).

**Preprocessing.** This raises the level of abstraction once step higher by adding contextual information.

**Storage/Management.** Provides a public interface to the obtained data.

**Application.** This layer represents the client application.

On the figure 2.1, the Sensors component represents the first two layers, i.e. sensors and raw processing, and Input Aggregator corresponds to the preprocessing and management layers.

#### 2.3.1.2 Observer

The Observer component represents a part of the management layer and the entry point to the application. It transforms integrated sensors' data to a high-level observation (see section 1.2), checks whether the observation complies with the access constraints (see the Access control component description below) and then, depending on the observation, activates the component responsible for further processing of the observation.

#### 2.3.1.3 Action Recommender

This is one of the key components of the architecture. It is responsible for the "Suggestions" part of the functional requirements, namely – providing suggestions about experiment workflow (next action to take), reagents to be used and their location, values of the experimental parameters (e.g. melting/annealing temperatures of primers). It also makes use of external services (like primer design tools, sequence matching algorithms, primers' databases, etc) that are available as web services [30, 31] or could be integrated "as is"(e.g., open source primer design tool Primer3 [6]).

There might be a number of possible solutions for the Action Recommender. We propose to base it on the IC-Service, a web service implementing the concepts of System for Implicit Culture Support [20].

SICS has originally been designed for gathering and effective utilization of implicit knowledge in a diverse community and thus represents a good choice for the core of a recommendation system (e.g. [21]). It is able to provide suggestions at no extra cost, without explicit question answering from users. Users just perform their usual work, and the system transparently collects patterns of the actions and thus is able to provide reasonable suggestions basing on the previous observations.

Using SICS as a remote module allows a number of system instances running in similar laboratories across the world to share the same recommendation core and thus to stick to common uniform working practices.

#### 2.3.1.4 Experiment viewer

The Experiment viewer allows users to access the records of previous and ongoing experiments, providing detailed information about experiment date, biologists involved in the experiment, actions taken, parameters used, and the experiment results. The user should be able to search for and/or filter a specific information in the history of experiments.

The information provided by the Experiment viewer can be used for reference, auditing or study purposes.

#### 2.3.1.5 Report Creator

This component implements reports-related functionality. It generates a printable report about selected experiments adapting local laboratory templates that specify formatting and level of details. Support of various report templates increases the system's flexibility and allows its usage in laboratories with different requirements to experiment reports (e.g. biological research lab reports are obviously different from these of a forensic lab).

#### 2.3.1.6 Experiments DB

High-level observations regarding the experiment are stored in a dedicated database. It is used by the Report Creator and Experiment Viewer components.

Here is a minimal list of required DB fields, suggested by biologists:

- Name of the person conducted the experiment;
- Experiment description (e.g. Single Nucleotide Polymorphism or Microsatellite; related gene name);
- Primers used;
- Polymerase used;
- PCR machine used;
- Cycling details (temperatures, timing);
- Reagents concentrations;
- Description of the template in a free text form;

- Associated files (e.g. electrophoresis photo);
- Experiment result (specificity of the product).

#### 2.3.1.7 User Manager

The User manager allows for access to the user profiles data stored in the Users DB. It is responsible for registration of new users, viewing and modification of the existing user profiles.

Moreover, the User Manager receives all observations that contain user location data and continuously updates the Users DB with this information, so that the Action Recommender is always aware of the spatial context of the user and can give suggestions taking into account user's location.

#### 2.3.1.8 Access Control

This component checks every observation perceived by the Observer for its eligibility. All the user-related observations, like request for some specific information, are verified against user's access rights. If the user is not allowed to access the requested information, the observation is forwarded only to the User Manager in order to update user's location and then ignored.

#### 2.3.1.9 Users DB

Users DB is a database containing user profiles information, roles and access rights, and last observed location of each user. This data is shared by the Access Control and User Manager components.

#### 2.3.1.10 Resource Scheduler

The Resource Scheduler component is dedicated to automatic control and reservation of laboratory equipment. Obviously, it requires cooperation with existing LIMS in order to avoid reservation conflicts. If there is no LIMS installed, the Resource Scheduler uses just a database containing information about reserved time periods and current states of the machines.

The Resource Scheduler provides information about reserved machine time to the Action Recommender, so that suggestions are automatically adjusted with respect to availability of required equipment.

The Resource Scheduler is also used for automatic control of the machines equipped with appropriate interface to computer (see, for example, Virtual Lab Dashboard project in section 1.3.3). For example, it may be ordered to switch off PCR machine once cycling has been done, or remind technicians about planned periodic maintenance of the equipment.

#### 2.3.1.11 Support Requester

The Support Requester is a simple component that sends notification messages (emails, SMS, etc) with requests to the people who do not use the system in their daily activity, e.g. technical support staff. This component is used for automatic reporting about failures and reagents base refill requests.

## 2.3.2 Actors, actions and objects

From the point of view of Implicit Culture [22], a biology laboratory is an environment in which actors (biologists) observe scenes (parts of the lab with objects) and perform situated actions (work) with objects (notebooks, tools, machines, reagents, etc).

Implicit Culture (IC) considers two sets of actors [22]:

$G$ contains the actors who act according to their cultural theory.

$G'$ contains the actors who do not know the cultural theory of G, but need to respect it, i.e. their actions should not violate it.

Intuitively, $G$ corresponds to experienced biologists and $G'$ corresponds to novices. However, in a laboratory, it is often difficult to divide all the actors in these groups; generally, scientists have different areas of competence and no one could be "The expert". Therefore, $G$ and $G'$ partially overlap in the context being discussed. Partially, because there still are some people who definitely belong to $G'$, e.g. students.

There are the following kinds of actors, with their own possible actions:

**Guest** These actors are only allowed to view the history of experiments.

**Technician** Functions available for technicians include in-experiment support, getting suggestions, using external services and, obviously, access to the history.

**Researchers** These actors have the same access rights as technicians, but are distinguished as a separate category due to higher assumed level of competence.

**Administrator** This actor manages the system and has a full control over it.

A list of PCR-related actions is provided below.

**`use primer`** This action corresponds to the primers choice for PCR. It takes one or two primers as parameters which are used as a main similarity criteria between all PCR actions. The user can also provide the concentration of primers' solution.

**`use polymerase`** This action takes polymerase name and concentration as the parameters.

**`use reagent`** This action relates to the choice of additional components of PCR, like buffer solution. It takes reagent name and amount as parameters.

**`set temperature`** This action is related to the thermocycler program. It allows the system to observe and store settings of melting/annealing temperature. It takes temperature value as a parameter.

**`wait`** This action also relates to the thermocycler program. It reflects the pause times between temperature changes. It takes time interval as parameter.

**`set cycle`** This action also relates to the thermocycler program. Its only parameter corresponds to the number of cycles the thermocycler must perform during the experiment.

Obviously, this list is not complete and should be extended with the actions specific for laboratory and equipment used there.

The objects to be considered in a cultural theory also heavily depend on the specific settings of the concrete laboratory. Here are just some of them related to the actions listed above: primer (name, sequence, size, concentration), polymerase (name, concentration), reagent (name, concentration), temperature (value), delay (interval). In the brackets there are possible attributes of the objects.

### 2.3.3 User interface

There is a growing number of research papers related to human-computer interaction (HCI) and its specifics in a biological laboratory [18, 47, 14, 37]. They address various aspects of HCI, including improvements of notebooks, adaptation of mobile computing technologies, etc.

Here we would like to stress only some of the key points. The main requirement for the organization of a user interface to the system is its transparency for the user, i.e. the interface should be either a part of the environment (like PCR machine) or occupy no working space (a good idea is to use a projection apparatus to display the data right on the table without obstructing it [34]). On the other hand, the system should be accessible wherever it is needed.

Interactivity and clearness of the interface are self-evident requirements.

One of the not-so-evident requirements is the need of context. Provided with the context of suggestion, the user may better understand it and make an own decision about its relevance. For example, having known that the author of the suggested action is a recognized expert in the area, the user is more likely to accept the suggested action than in the case when the suggestion comes from a third year student experiments.

## 2.4 Discussion

The system overcomes a number of well-recognized common drawbacks of CSCW systems (as of Trevor et al. [43]):

**Unrealistic models of the real-world** The presented system dynamically follows the changes of the workflow due to use of users' experience as the source of suggestions. The Action Recommender module seamlessly adjusts to the current real-world practices in the laboratory.

**Constraining models of control** Similarly with the argumentation above, the proposed architecture does not put requirements on users to behave methodically and strictly adhere to the workflow. It just provides a hint of a "probably best" action to take, leaving, however, the final decision to the user.

**Lack of awareness of "groups"** ("People are unaware on who is cooperating on what" [43].) This is overcome by providing a list of on-going experiments.

**Limited support for "sharing"** This one is not addressed by the system. The users can use conventional means of communication: email, instant messaging, etc.

For a workflow-based CSCW system it is important to allow initial specifications with different degrees of completeness [24]. The system uses a set of SICS-rules as the base-level workflow specification. Having collected more observations during its work, the system is able to induce more rules and thus provide the user with more up-to-date workflow. Put in other words, the system produces a flexible workflow dynamically following the current practices of the users. It is an advantage compared to the conventional workflow-based CSCW systems with predefined and rigid workflow specifications.

Compared to the Labscape [14], the proposed system has the following distinctive features.

- The system is PCR-specific.
- IT provides extensible functionality via integration with third-party web-services.
- It does not use predefined templates, but automatically reflects the laboratory's specific working styles and procedures instead.
- It may easily be shared by similar laboratories worldwide, which helps to maintain common working practices in remote laboratories.

Unfortunately, the suggested system suffers from the cold-start problem (as has been mentioned in [42]). However, this has only a partial and temporary impact on the system, resulting only in the inability to provide SICS-based suggestions while other features work properly. The cold-start problem disappears as the users continue to work with the system and the observation database grows.

# Chapter 3

# Prototype design

In order to test and verify the design of the proposed system, we have created its prototype with a particular focus on the recommendation subsystem. This chapter describes the prototype's design, architecture and functionality.

## 3.1 Choice of the requirements

For the prototype design we have selected a range of functional requirements from the list elaborated for the system for PCR support (section 2.2). The selection criteria were to choose as many of suggestion-related requirements as possible, keeping the prototype feasible and implementable in the time available and given the real-world data provided by the biologists.

The core requirements for the prototype are the following.

- Provide suggestions on the experiment's workflow (next action to take).
- Provide suggestions on the experiment parameters.
- Assign each experiment a description in a text form.
- Display current experiment's progress so far.

The initial requirements to the suggestions, namely the history-based one provided by SICS and that for similar primers, has been extended by the biologists' request. The extension allows the prototype to enhance suggestions related to the melting temperature of primers with values calculated basing on the rule-of-thumb formulas 1.1 and 1.2.

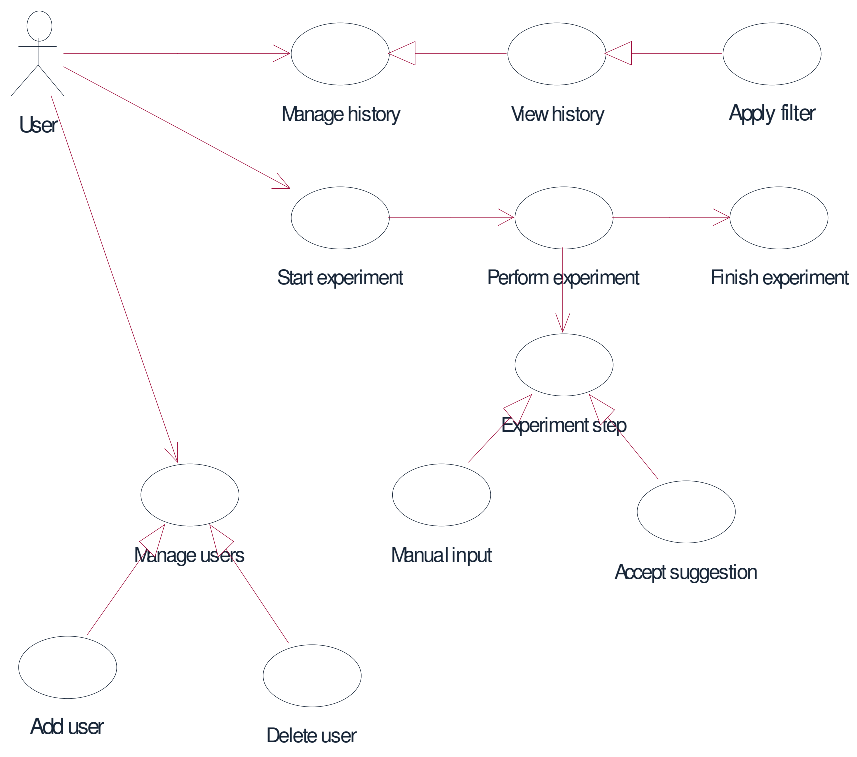


Figure 3.1: Use cases of the prototype

We have also included some of the requirements which would enhance prototype's functionality and do not require significant implementation efforts:

- Provide an access to the history of experiments with search/filtering ability.
- Observe the "degree of successfulness" of the experiment.
- Provide user management functionality.

Although there was no need for access control with different rights, the prototype needs basic user management functionality because the experiment records must be assigned the name of the biologist who conducts the experiment.

The use case diagram for the prototype is presented at figure 3.1.

Biologists have suggested that they would prefer to have not a simple success/fail outcome of PCR, but rather something more detailed instead. Thus, the prototype supports the following kinds of PCR results:

- Specific product;

- Unspecific + specific products;
- Unspecific product;
- No product.

Regarding the experiment types, the prototype supports two of them: Single Nucleotide Polymorphism (SNP) and Microsattelite. This choice has been motivated by the availability of the real experimental data only on these kinds of PCR experiments.

As the main goal of the prototype was testing of the recommendation subsystem, we gave up the transparency requirement for the sake of feasibility. Development of the multiple-sensor input would have taken unaffordable amount of time while not being critical for the main goal. Therefore, executed actions should be directly entered by the user instead of being transparently observed.

For the user interface we have chosen a web-based one, which has been proven to be a good choice for building CSCW applications (for example, see [19]). Web interface has a number of features important for the PCR support system:

- Minimal equipment requirements. The only thing needed to access the system via web interface is a computer with a web browser, which is included by default to any modern operating system.
- Platform independence, which ensures portability of the system across the different client platforms.
- Worldwide access. The system with the web interface may be accessed worldwide in a uniform manner. This allows for joining the experience of a number of laboratories around the world.

However, the web-based interface has also a number of limitations, some of which are listed below.

- No possibility to work offline. This, however, does not affect the prototype as it is a network-based application and is not supposed to be used offline.
- Simplistic interface. This limitation is alleviated by a number of modern technologies like flash and JavaScript, which allow implementation of client-side interactivity.

- No support for server-initiated actions. Although there is no direct support for such actions, there are at least two solutions. A straightforward one is automatic periodical update of the current page; a more complicated implies usage of client-side interactive component maintaining a direct communication with the server.

## 3.2 Cultural theory

The prototype is based on the System for Implicit Culture Support (SICS) library [20].

The set of actions possible during PCR is based on the key article about PCR (Mullis et al. [39]) and the BioSICS project ( Sarini et al. [42]), an ancestor of the present one. For the sake of feasibility of the prototype testing, the set of actions has been limited, correspondingly to the available real-world data. These actions are self explanatory and kept as general as possible:

**`use primer`** This action corresponds to the primers choice for PCR. It takes one or two primers as parameters which are used as the main similarity criteria between all PCR actions. It can also take two optional parameters that were available in the real data: primer's name and real size of the primer.

**`use reagent`** This action relates to the choice of additional components of PCR, like buffer. It takes reagent name and amount as parameters.

**`set temperature`** This action is related to the thermocycler program. It takes the value of melting/annealing temperature as the only parameter.

The format of the actions' parameters and some examples are given in the table 3.1. Optional parameters are surrounded with square brackets.

| Action | Parameters | Example of parameters |
|---|---|---|
| `use primer` | sequence1[(name1, realSize1)]<br>[sequence2[(name2, realSize2)]] | CTATAC<br>TCGGAG(VI.a12f, 250) |
| `use reagent` | name[(amount)] | MgCl2(1.5mM) |
| `set temperature` | value | 59 |

Table 3.1: Actions and their parameters

The cultural theory for the prototype is simple and straightforward. First of all, the user is suggested to specify primers he/she is going to use in the experiment (`use primer` action). Once the primers are specified, the user is free to specify either components of the buffer solution (`use reagent` action) or the temperature for the reaction. Last two actions are suggested interchangeably thereafter. The suggested parameters are based on the previous action taken and the primers used in the experiment (see section 3.3.2 for more details). As the rest of the experiment may vary widely, the system provides possible parameters for both `use reagent` and `set temperature` actions.

Formally, the cultural theory is the following.

- `UsePrimer`(_actor; primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time1; (;))
  $\rightarrow$ `UseReagent`(_actor; reagent(name="*", amount="*"), primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time2; (;))
  $\wedge$ `SetTemperature`(_actor; temperature(value="*"), primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time2; (;))

- `UseReagent`(_actor; reagent(name=_name,amount=_amount), primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time1; (;))
  $\rightarrow$ `SetTemperature`(_actor; temperature(value="*"), primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time2; (;)) $\wedge$ `UseReagent`(_actor; reagent(name="*",amount="*"), primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time2; (;))

- `SetTemperature`(_actor; temperature(value=_t1), primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time1; (;))
  $\rightarrow$ `UseReagent`(_actor; reagent(name="*",amount="*"), primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time2; (;))
  $\wedge$ `SetTemperature`(_actor; temperature(value="*"), primer1(sequence=_seq1, name=_name1, realSize=_size1), primer2(sequence=_seq2, name=_name2, realSize=_size2); ; time2; (;))

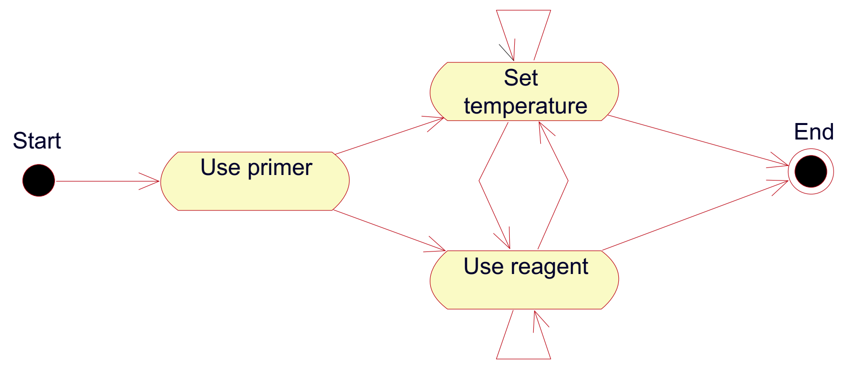


Figure 3.2: PCR workflow

Graphically the workflow the prototype suggests to follow during PCR is presented by the activity diagram at figure 3.2.

## 3.3 Design decisions and considerations

### 3.3.1 SICS module

For the prototype, we used the SICS module adapting a direct way to interface it, that is as a library, not a web service (see section 1.2). This does not have any adverse impact on the functionality of the prototype, but results in a higher runtime performance and easier implementation instead.

Also, we decided to exclude the inductive module from the SICS library. This has been motivated by two reasons. The main one is that the inductive module was still at the development stage and therefore unreliable at the time of prototype creation. Another reason is the fact that the basic cultural theory is already known and the need to discover new rules dynamically is not a crucial one.

For the data storage we have decided to use internal storage of SICS library. Although it has a reduced functionality, this allowed us to save some time on development of a dedicated storage. Therefore the SICS library along with few adapters provides data storage functions for observations and user profiles.

### 3.3.2 Context independence

While testing one of the first versions of the prototype, it has been observed that the suggestions the SICS library gives out do not depend on the parameters of the previous actions, or, which is the same, the experiment's context.

The worst side of this was the fact that the suggested temperature value did not depend on the primer's sequence – the SICS simply suggested the most prevalent temperature value from the previous observations.

The solution has been found in assigning (transparently for the user) two additional parameters to `use reagent` and `set temperature` actions, namely primers. Indeed, the primer's sequence is the most unique parameter of the experiment (the least recurring one, more exactly), which allows the SICS to distinguish contexts.

### 3.3.3 Aggregation of observations to experiments

The domain of the activity of biological laboratory implies the concept of experiment as a set of actions continuous in time and leading to some result, either positive or negative. Unfortunately, the SICS has no hierarchy of observations, they all are saved in a 'flat' storage. Therefore, aggregation of observations (that is executed actions) to an experiment should be done at the application level.

In order to do this, two auxiliary actions has been defined:

**startExperiment** This action designates the start of the experiment and is sent to the Observer before any other action of this experiment. It contains attributes specifying the type of the experiment (see section 1.1) and a free-text description.

**endExperiment** This action designates the end of the experiment and is sent to the Observer when the experiment is done and its result is known. The result is saved in an attribute of this action.

Thus, whenever the prototype needs to convert a list of observations to a list of experiments (for example, in *View History* use case) it scans the observations, considering *startExperiment* and *endExperiment* as the boundaries.

### 3.3.4 Failed experiments processing

One of the issues of the prototype design and SICS library usage was handling of the failed experiments' observations.

The problem is that both successful and failed PCRs are in the same storage, and there is no direct way to tell for any arbitrary action whether it belongs to a failed or successful experiment.

We could adopt the approach used to solve the context problem (section 3.3.2) and assign each action an additional attribute designating success of the experiment in whole. But ultimately, the problem is that the SICS scene producer cannot distinguish ‘good’ and ‘bad’ actions, i.e. to give a negative weight to the suggestions which are similar to the actions of a failed experiment. We would simply result in setting up the similarity evaluation module to exclude ‘bad’ actions from consideration at all.

Because of the mentioned reasons we have taken a simplifying, although not a pretty one, decision to save only successful experiments in the SICS observations storage. This allowed us to simplify the code without loss of functionality.

Due to the fact that the experiment result is unknown until it is finished, the observations belonging to the current experiment are not sent to the storage one-by-one but kept in the memory instead. When the experiment is finished positively, all of its observations are stored at once (retaining their real timestamps).

### 3.3.5 Primers’ similarity

Generally speaking, evaluation of the degree to which two sequences of nucleotides are similar, is not a trivial task.

First of all, when comparing two primers, i.e. nucleotide sequences, one should take into account the properties which are being compared, that is define a similarity criterion. This can be, for example, as simple as the amount of “A” nucleotides in the sequence or as complex as the distance between primers in a specific gene. Obviously, in the latter case knowing just two sequences is not enough, one needs to know also the structure of some part of the gene.

What concerns PCR, the task is complex, because even two primers having just a couple of mismatches (different nucleotides in the same positions) might be considered either very different or very similar, depending on the target DNA sequence being amplified.

Having discussed the issue with biologists, we adapted the following simplistic approach. Two primers are considered similar, if there are no more than 3 mismatches. Possible gaps are considered as mismatches. If the primers have different length, a sequence of “gap” symbols is attached to the shorter one in order to make the length equal. For more details about primer similarity evaluation algorithm please refer to section 4.2.2.

The similarity measure is just the frequency of matches calculated by the following formula:

$$similarity \equiv 1 - \frac{number\ of\ mismatches}{length\ of\ the\ primers} \tag{3.1}$$

A typical length of primer is around 20 nucleotides. Having 3 mismatches allowed, gives us a maximal frequency of mismatches equal to 15%. That is, two primers are considered similar, if the frequency of mismatches is less or equal to 15%.

The prototype provides suggestions for similar primers along with their sequences and mismatches highlighted. The ultimate decision regarding suitability of the proposed actions and similarity of primers is thus made by the expert, i.e. biologist who uses the application.

### 3.3.6 User interface

The user interface design has passed a number of stages due to the biologists' feedback. The initial guide was the mockup presented by Sarini et al. [42]. However, even a prototype needs an elaborate interface due to the well known fact that the quality of the interface can bias users' opinion about functionality of the prototype.

Therefore, we dedicated a special effort to the development of the user interface, including client-side interactivity. More details about that are presented in section 4.3.

## 3.4 Prototype architecture

The architecture of the prototype is based on that of the system for PCR support described above (see section 2.3). The components which are not critical for meeting the requirements, have been removed. The prototype's design architecture, slightly rearranged for clarity, is shown on figure 3.3.

However, figure 3.3 is too abstract for implementation. A more practical architecture is presented at figure 3.5, while the transition from design to implementation architecture and their interrelation is shown at rather overloaded figure 3.4.

The main components of the architecture and their functional roles are listed below.

**Web frontend** represents the entry point of all the user requests to the server-based application. It implements part of the Observer's functionality. Namely, it parses the HTTP requests from the user's Web Client to the internal application's format and then forwards processing to the appropriate component, in accordance with the request. When the result is ready, the web frontend transmits it to the client transparently for the component which has provided this result.

**Input Adapter** is responsible for the part of the Observer's functionality related to the conversion of raw observed data to a high-level observation.

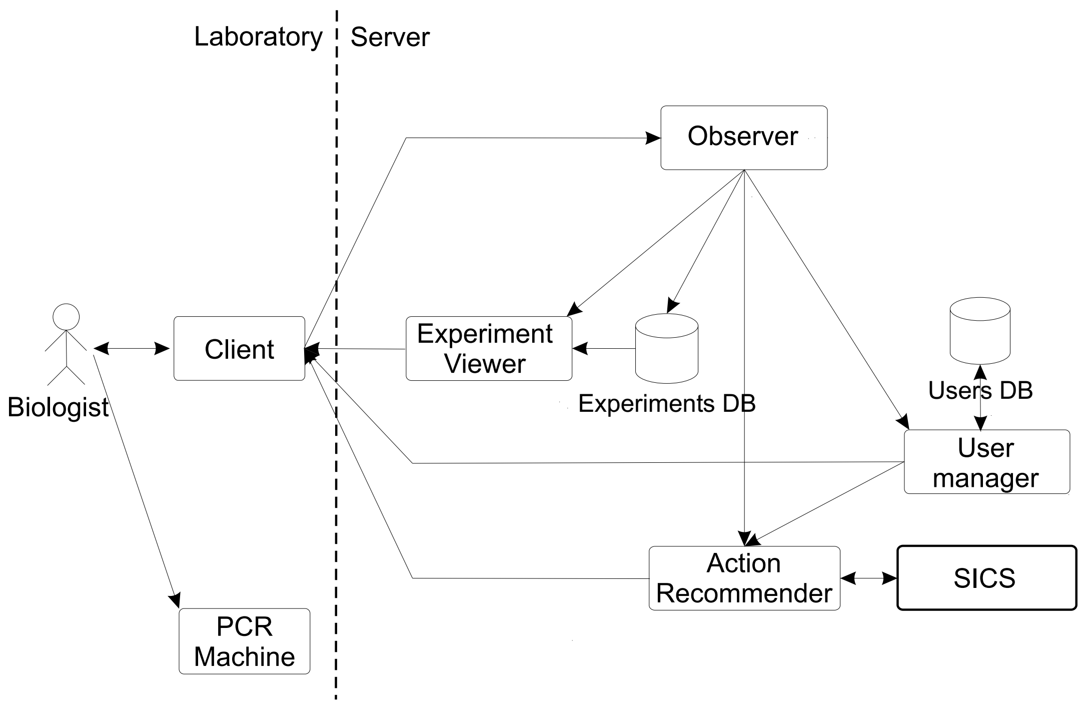


Figure 3.3: Prototype's design architecture

**Suggestion Manager** is the key module of the prototype; it represents the Action Recommender. It functions in the following way.

First, the Suggestion Manager receives an observation from the Input Adapter. Then, it performs a query to the SICS library for an appropriate suggestion. In the next step, it extracts from the observation the information about the primers used in the experiment, and using the Primer Similarity Evaluator makes a number of SICS queries for the similar primers. Finally, if one of the suggestions gathered so far contain `set temperature` action – the suggestions are enriched with yet another `set temperature`, with a value provided by the Temperature Calculator.

Thus the Suggestion Manager provides recommendation about next action to take and its parameters.

**Primer Similarity Evaluator** provides the Suggestion Manager with a list of the known to the prototype primers that are similar to those currently used in the experiment. More details about primer similarity evaluation are given in section 4.2.

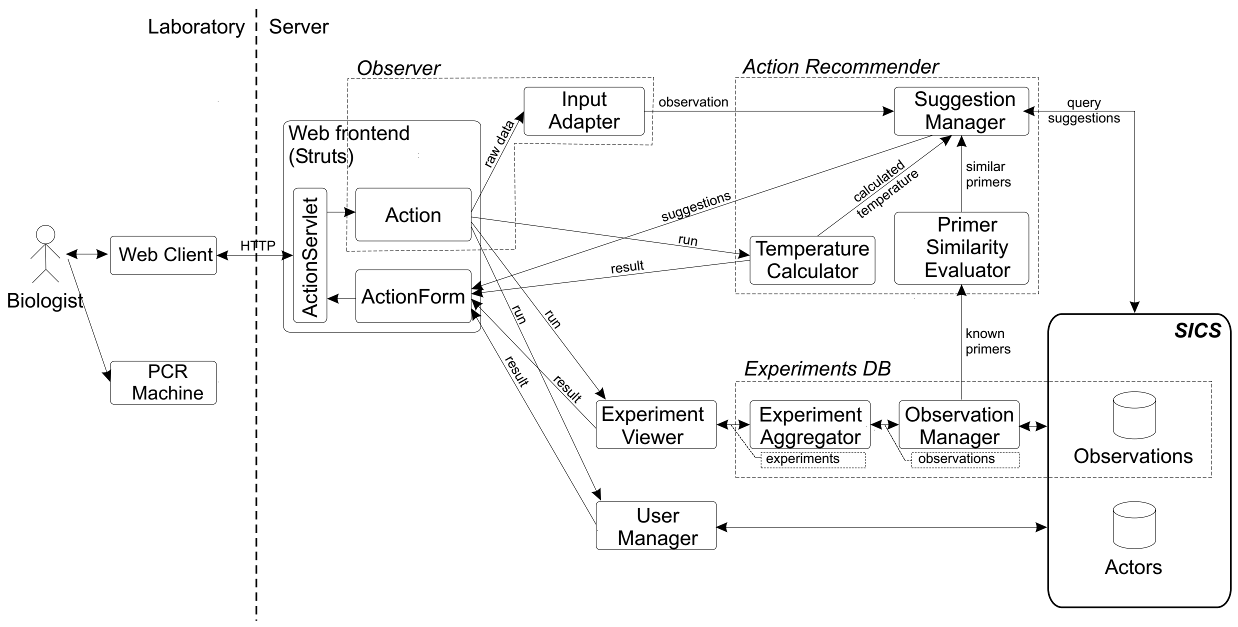


Figure 3.4: The relation between design and implementation architecture of the prototype

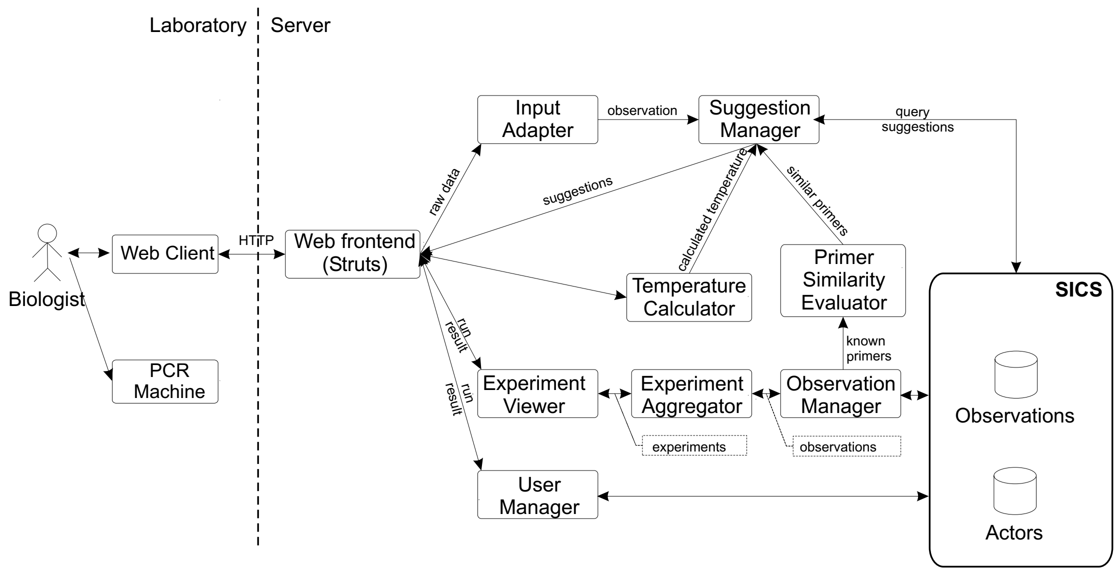


Figure 3.5: Prototype's implementation architecture

**Temperature Calculator** is a simple utility implementing formulas 1.1 and 1.2 for estimation of melting temperatures of primers. It is used by the Suggestions Manager and also can be accessed directly via dedicated interface.

**SICS library with data storages** is the core of the prototype. It integrates the recommendation system functionality and data storages of observations and actors (used instead of the Experiment DB and User DB components on figure 3.3)

**Experiment Viewer** is the component providing access to the history of experiments. It also implements some basic filtering of the history. Further details are provided in section 4.3.3.

**Observation Manager** is an extension for the storage module of the SICS, providing some application specific functionality, like getting a list of known primers, etc.

**Experiment Aggregator** implements a workaround for the SICS' lack of hierarchy of observations (see section 3.3.3). It converts a set of observations to a set of experiments. With regard to the system architecture, Experiment Aggregator, Observation Manager and SICS data storage correspond to the Experiments DB.

**User Manager** is the same as in the system architecture, with the only exception that it uses SICS data storage instead of the Users DB.

## 3.5 A typical workflow

Let us consider a typical interaction of prototype's components using "Experiment step" use case as an example.

The sequence diagram of the actions is shown on figure 3.6.

At the beginning, the request from the biologist's web browser is received by the prototype's web frontend. The frontend forwards the request to the corresponding processing element, namely, NextStepAction class, which is responsible for the "Experiment step" use case.

Then, the application transforms the raw data from parsed HTTP request to a high-level observation in terms of Implicit Culture concepts (section 1.2).

Due to the previously discussed fact that the experiment result is unknown until the end of the experiment and the prototype does not store observations for failed experiments (section 3.3.4), the application uses a buffer in the server's RAM, which stores

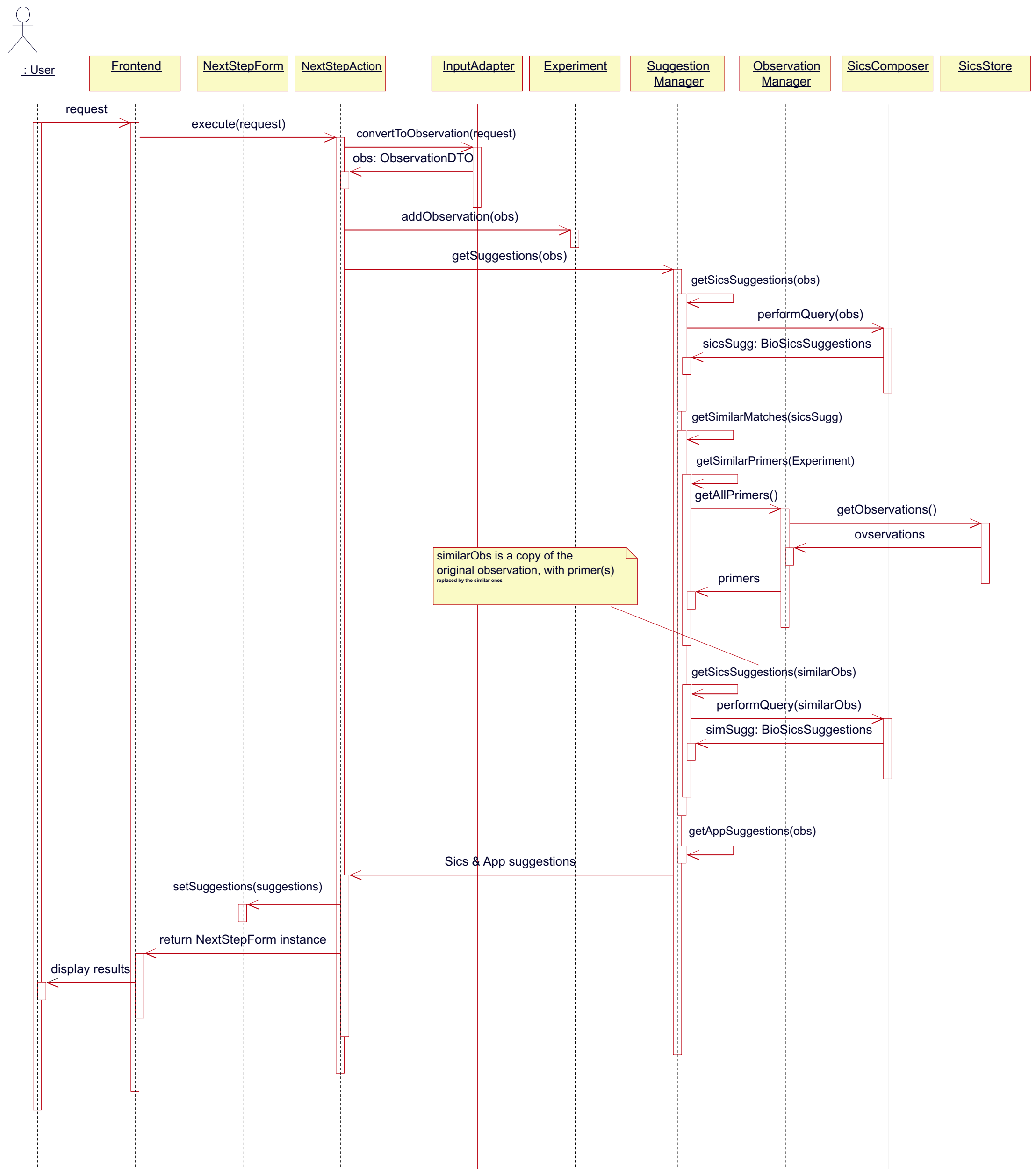


Figure 3.6: Sequence diagram of the “Experiment step” use case

the experiment's observations. All executed actions observed during the experiment are added to this buffer.

After saving the observation, the application requests the suggestions from the Suggestion Manager. Firstly, the Suggestion Manager performs a query to the SicsComposer, passing the current observation to it.

Then, the Suggestion Manager obtains a list of the primers known to the system (i.e. observed before) which are similar to the ones being used in the current experiment. This is done by means of receiving all the primers from the SICS observations store via the Observation Manager, and then filtering it by primers' similarity estimation algorithm (for more details about filtering algorithm see section 4.2). Then, the SICS library is queried again with almost the same observation except that the primers pair in it is replaced with just one, similar to either of the original.

The suggestions obtained in the above steps are called SICS-based recommendations and saved in a separate list.

There is also a list of application-based recommendations (returned by the getAppSuggestion() method; see figure 3.6). These represent the suggestion which are received from the sources different from the SICS library. In the prototype, there is only one such suggestion – the one related to the `set temperature` action. If the list of SICS-based recommendations contains any suggestion of this action, the application adds another one `set temperature` with the suggested value proposed by the Temperature Calculator component (see figure 3.5).

At this step, all the suggestions have been produced and they are filled into the NextStepForm – a special component which serves as a data transfer object for the frontend. Finally, the page with the suggestions is displayed to the user.

## 3.6 Summary

This section has introduced a design of the prototype of a system for PCR support. It describes the cultural theory of a PCR experiment in terms of Implicit Culture; the choice of requirements, design decisions and the proposed prototype architecture has also been presented.

# Chapter 4

# Prototype implementation

This chapter contains the details of prototype implementation and evaluation results.

## 4.1 Choice of development tools and technologies

Development of a complex software system requires application of models of high level of abstraction and effective solutions for components interaction. The required properties are provided by object-oriented development approach.

For the implementation of the prototype, we have chosen Java2 Enterprise Edition platform [3]. For the web frontend we adapted a well-known Struts framework [2]. Recommendation system core has been implemented on the base of System for Implicit Culture Support(SICS) generic library [20]. The prototype has been deployed on Tomcat servlet/JSP container from Apache Jackarta project.

The motivation for this selection of technologies is presented below.

### 4.1.1 Java, Servlets and Java Server Pages

Java technology includes not only a powerful object-oriented language which allows creation of platform independent extensible applications, but a set of various tools and libraries simplifying development of complex software systems. It provides the following features:

- Portability, extendability and modularity. Platform independence meets the important requirement for Internet-based system by allowing the application to be installed on different operating systems. There are Java Virtual Machine versions for all most popular OS (Windows, Unix/Linux, MacOS, etc).

- New features such as Servlets and Java Server Pages(JSP) technologies allow for more time-effective and easier development of web-based software systems. All the functions related to interaction between server and Java-application are performed by servlet container.

- Java Server Pages(JSP). Utilization of JSP as the next evolutionary step of Servlet technology allows one to divide an application into two loosely coupled parts: interface and business logic. Thus, the same functionality can be provided via different interfaces which provides high implementation flexibility.

- Openness of architecture and predictability of the further development. Directions of the platform development are clearly defined and take into account large number of participating projects while the future development of competitive Microsoft's .NET platform is not transparent for those who do not work in the company and is presented as just a general planning.

High functionality of Java, platform independence, intrinsic security and free of charge availability of advanced libraries makes this platform one of the best for Internet-based solutions.

### 4.1.2 Struts framework

Struts framework has been created to ease development of components of a web application which are responsible for processing of user requests, and make this processing more flexible [2].

The main architectural style Struts stimulates to use is the standard "Model–View–Controller"(MVC) design pattern. Here, view is presented by JSPs, and controllers by servlets. The main differences between Struts and standard J2EE approach are related to higher specialization of servlets, and standardization of data exchange between processing servlet and JSP page presenting the results.

Main components of the framework include:

**Action servlet** It represents single entry-point of the application and dispatches the requests to their respective processors (actions).

**Action** This is a base class for the components which perform request processing. Usually, one or a few actions correspond to a single use case of the system.

**ActionForm** This is data transfer object used in interaction between actions and JSPs showing the results.

General workflow of a system designed using Struts framework is the following.

- HTTP request is received by the entry-point servlet and parsed to form a logical request. Parsed data is filled into corresponding ActionForm instance.
- Request parameters are checked for validity and if any problem is found, the user is notified about it in a most informative manner.
- The correct logical request and its parameters are transformed to calls of the appropriate actions which implement business logic.
- Actions process the requests and put the results to display in the corresponding ActionForm.
- The ActionForm containing the processing results is transferred to the components responsible for building of their representation.
- Created page is displayed to the user.

The main advantages of the Struts framework are its high flexibility, which implies easy architecture extensibility, and appreciable reduction of the development time.

## 4.2 Primers' similarity processing

### 4.2.1 Suggestions with similar primers

Due to rather specific nature of primer sequence matching algorithms and limited flexibility of the SICS library concerning similarity configuration, the primers' similarity evaluation has been implemented at the application level. This also allowed us to achieve some extra functionality, like highlighting of mismatches in the user interface.

The algorithm is the following.

```
Input: observation;
Output: suggestions for similar primers;

allPrimers = list of all primers ever used;
for (each obsPrimer: primer in the current observation) {
  for (each primer: element of allPrimers) {
    similarity = getSimilarityScore(primer, obsPrimer);
    if (similarity > 85%) {
      tmpObs = copy of observation;
      replace primers in tmpObs with the similar one;
      similarSuggestion = getSicsSuggestions(tmpObs);
      add similarSuggestion to the list of results;
    }
  }
}
```

### 4.2.2 Primer's similarity evaluation

As it has already been said (see section 3.3.5), the evaluation of primers' similarity is generally not a trivial task. For the prototype, we have implemented a simplistic approach based on calculation of mismatch frequency (see section 3.3.5). The final decision concerning primers' similarity is taken by the biologist who uses the prototype.

For better visual perception of the comparison results, the algorithm also produces a list of mismatches' positions. This list is then used to underline corresponding symbols in the primers.

The pseudo code for the similarity calculation algorithm is the following.

```
Input: primer1, primer2;
Output: similarity value, list of mismatches' positions;

//Ensure that the primers have equal length by adding
//GAP symbol to the end of the shorter primer
while (primer1.length > primer2.length)
  primer1.append(GAP);
while (primer2.length > primer1.length)
  primer2.append(GAP);

primerSize = primer1.length;
//At this moment both primers have length primerSize
```

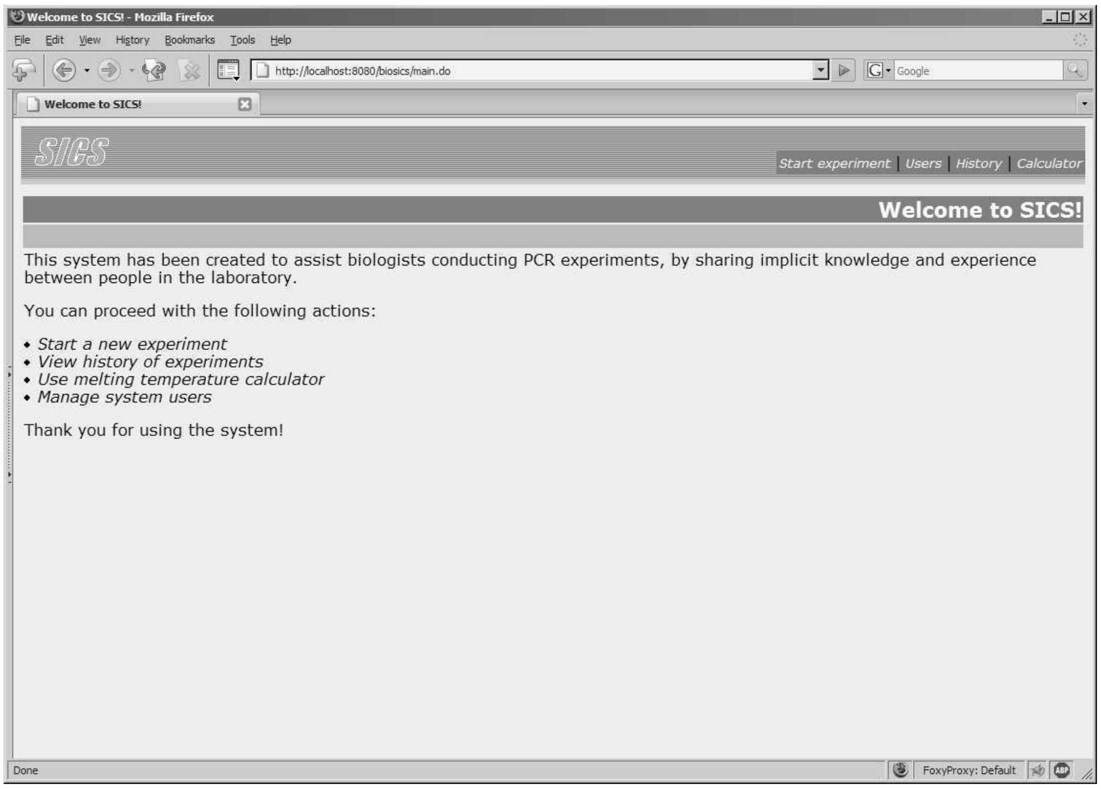
```
mismatchPositions = new list of integers;
int matchesCount = 0;
for (position from 1 to primerSize)
  if (primer1[position] == primer2[position])
    increase matchesCount by one;
  else
    add position to mismatchPositions;
similarity = matchesCount/primerSize;
```

## 4.3 Working with the prototype

This section contains a description of the user interface of the created prototype and general instructions on how to use the application in PCR-related activities.

### 4.3.1 Getting started

The start page of the application is displayed on figure 4.1. In the top part of the page


Welcome to SICS! - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8080/biosics/main.do
Google
Welcome to SICS!
SICS
Start experiment | Users | History | Calculator
Done
FoxyProxy: Default

Welcome to SICS!

This system has been created to assist biologists conducting PCR experiments, by sharing implicit knowledge and experience between people in the laboratory.

You can proceed with the following actions:

- *Start a new experiment*
- *View history of experiments*
- *Use melting temperature calculator*
- *Manage system users*

Thank you for using the system!

Figure 4.1: Start page of the application

there is a logo of the prototype and main menu items. Below the menu there is a line, which displays information about the current module, so that the user always knows where he or she is. The rest of the page displays the user's workspace.

Main menu is one of the means of interaction with the prototype. It provides instant access to the different application functions. There are the following items in the main menu.

**Start experiment** This item allows user to initiate a new experiment and get an in-experiment support.

**Users** This item provides access to the user management module.

**History** This item provides access to the history of experiments.

**Calculator** This item forwards the user to the melting temperature calculator utility.

The start page displays the links duplicating the items of the main menu, but having an extended description, so that even a new user can easily decide which one to pick.

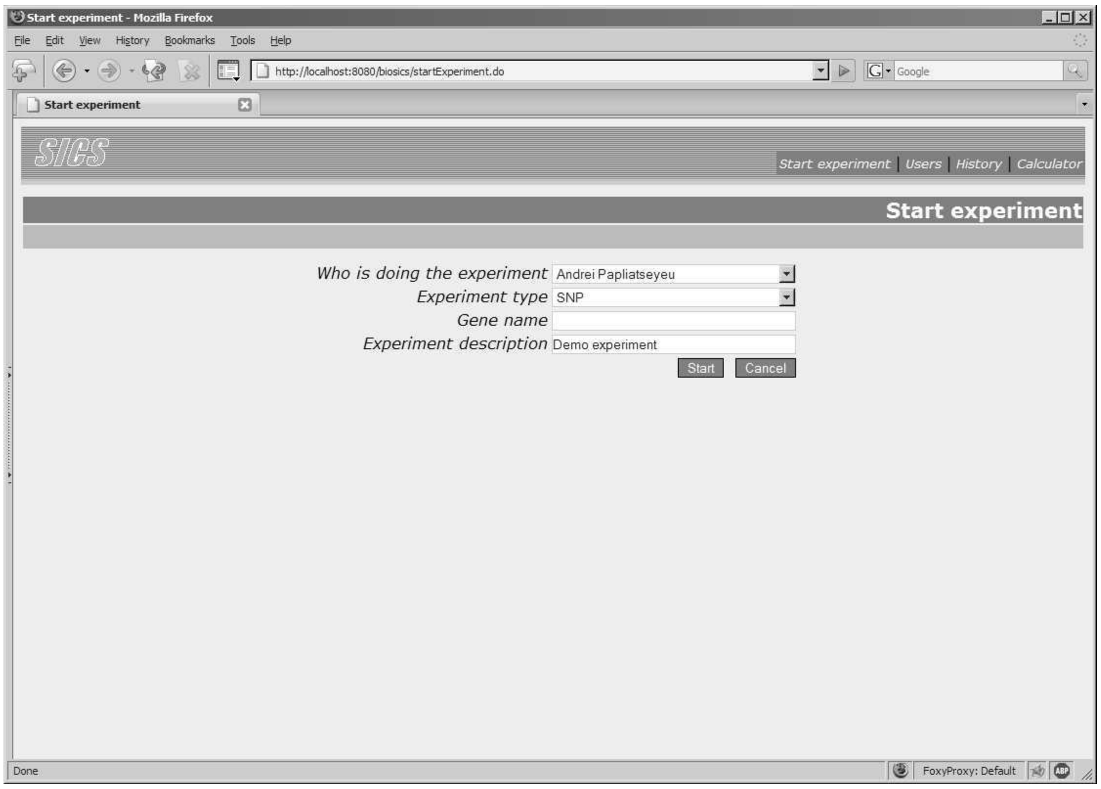
The user can get to the start page from any module of the application by clicking the prototype's logo. This will cancel any non-saved changes, which is useful if the user is not familiar with the application and simply explores its functional abilities.

### 4.3.2 In-experiment support

In order to start a new experiment within the application, the user has to click "Start experiment" item in the main menu. On the next screen (figure 4.2), he or she defines experiment properties: selects his/her name from the list of laboratory staff, chooses the experiment type (SNP or Microsatellites), can optionally set the name of the gene the experiment relates to, and give a free-text description of the experiment. When the new experiment is confirmed by pressing 'Start' button, the user comes to the 'Experiment step' page which represents the key functionality of the prototype (figure 4.3).

In this state, the workspace is divided into two parts. The left side of the page contains some guiding information, suggestions and controls required to record the experiment's actions. The right part displays the flow of the experiment, namely, the actions that were taken and the respective parameters, along with timestamps.

Lets take a closer look at the left part of the page. It is divided vertically into three logical groups: list of suggestions, action input controls and experiment result controls.



Start experiment - Mozilla Firefox

http://localhost:8080/biosics/startExperiment.do

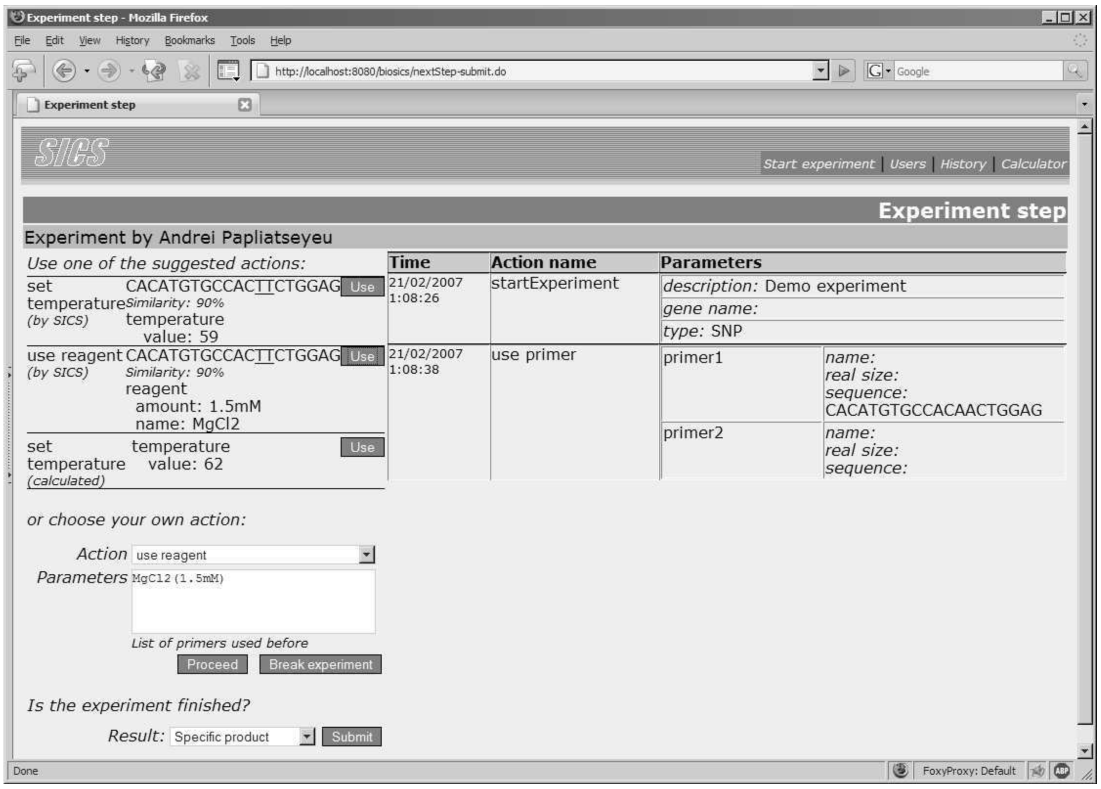
Start experiment | Users | History | Calculator

**Start experiment**

| | |
|---|---|
| *Who is doing the experiment* | Andrei Papliatseyeu |
| *Experiment type* | SNP |
| *Gene name* | |
| *Experiment description* | Demo experiment |

Start Cancel

Figure 4.2: "Start experiment" interface



Experiment step - Mozilla Firefox

http://localhost:8080/biosics/nextStep-submit.do

Start experiment | Users | History | Calculator

**Experiment step**

Experiment by Andrei Papliatseyeu

*Use one of the suggested actions:*

| | | |
|---|---|---|
| set temperature *(by SICS)* | CACATGTGCCACTTCTGGAG *Similarity: 90%* temperature value: 59 | Use |
| use reagent *(by SICS)* | CACATGTGCCACTTCTGGAG *Similarity: 90%* reagent amount: 1.5mM name: MgCl2 | Use |
| set temperature *(calculated)* | temperature value: 62 | Use |

| Time | Action name | Parameters | |
|---|---|---|---|
| 21/02/2007 1:08:26 | startExperiment | *description:* Demo experiment | |
| | | *gene name:* | |
| | | *type:* SNP | |
| 21/02/2007 1:08:38 | use primer | primer1 | *name:* *real size:* *sequence:* CACATGTGCCACAACTGGAG |
| | | primer2 | *name:* *real size:* *sequence:* |

*or choose your own action:*

*Action* use reagent

*Parameters* MgCl2 (1.5mM)

*List of primers used before*

Proceed Break experiment

*Is the experiment finished?*

*Result:* Specific product Submit

Figure 4.3: "Experiment step" interface

#### 4.3.2.1 List of suggestions

Each of the items of the suggestion list contains:

- The name of suggested action followed by indication of the source of the recommendation (SICS-based or calculated by the application)
- The sequence of the primer for which this suggestion has been proposed. If the suggestion is proposed for some similar primer, mismatches between the original and similar sequences are underlined (e.g. "`TT`" on figure 4.3). Estimated similarity value is shown below.
- Suggested values of action's parameters. In terms of SICS, parameters are objects and their attributes. E.g. on figure 4.3 action `use reagent` has the only object "reagent" which has two attributes: reagent name and amount (concentration).
- "Use" button, which provides a simple and quick way to choose the recommended action. When the user clicks "Use" at the desired action, this action along with its parameters is copied into the input field, so that the user can either accept it "as is" or make some modifications before proceeding.

As one can see, SICS- and application-based recommendations are listed together with just a tag showing the origin of the suggested action.

#### 4.3.2.2 Action input controls

If the application does not provide any suggestions (e.g. at the start of the experiment) or the user is not satisfied by them, he or she can input the executed action directly by choosing the appropriate action name from combobox and filling the action's parameters in the field below. The format of parameters for each action is presented in table 3.1.

For the user's convenience, there is a link to a dynamically generated page which displays the list of all primers ever observed by the application.

When the executed action and its parameters has been entered or chosen from the list of suggestions, the user clicks "Proceed" in order to continue the experiment and get suggestions on the executed action.

If for some reason the experiment must be terminated, the user should click "Break experiment" button, which clears the buffer of observations made during the experiment and forwards the user to the start page. The same result can be achieved by clicking the logo on the top of the page, but field tests has shown that such a solution of breaking the experiment was unintuitive for the users. Thus, we added a special button to break the experiment without saving any observations.

#### 4.3.2.3 Experiment result controls

When the experiment is considered as finished, the user should inform the prototype about the outcomes. These can be:

- specific product;
- specific and unspecific products;
- unspecific product;
- no product.

Having selected the right product type, the user should send it to the application by clicking “Submit” button.

If the experiment has succeed, the observations accumulated during the experiment in a temporary buffer are sent to the SICS and saved in the internal observations store. Otherwise, the experiment is ignored (see section 3.3.4 for detailed explanation).

The rejection criteria should represent a balance between accepting “bad” experiments and rejecting “satisfactory” ones. Therefore, the prototype’s decision boundary is set on “No product” result. The experiments that finish with at least unspecific product are stored.

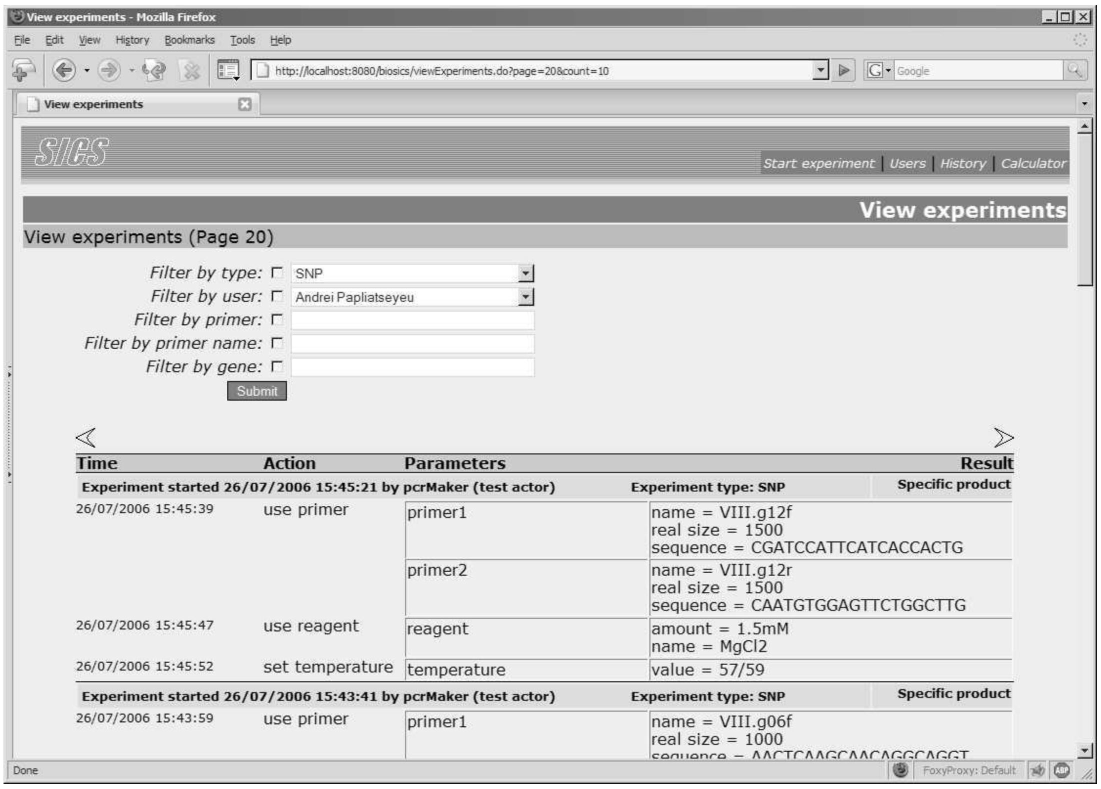
### 4.3.3 History of experiments

History of experiments can be accessed via main menu item “History”. Prototype’s interface in this mode is shown on figure 4.4.

The workspace of the experiment history viewing module is divided in two main parts.

At the top of the page, there is a set of filter settings fields, each one preceded by a checkbox which includes the corresponding field into the filtering criteria. Thus, filtering criteria may include an arbitrary set of independent fields, which provides a high flexibility and control over the process.

The history can be filtered by the following fields:


View experiments - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8080/biosics/viewExperiments.do?page=20&count=10
View experiments
SICS
Start experiment | Users | History | Calculator
Submit
Done
FoxyProxy: Default

**View experiments**

View experiments (Page 20)

| | |
|---|---|
| *Filter by type:* ☐ | SNP |
| *Filter by user:* ☐ | Andrei Papliatseyeu |
| *Filter by primer:* ☐ | |
| *Filter by primer name:* ☐ | |
| *Filter by gene:* ☐ | |

| Time | Action | Parameters | | Result |
|---|---|---|---|---|
| **Experiment started 26/07/2006 15:45:21 by pcrMaker (test actor)** | | | **Experiment type: SNP** | **Specific product** |
| 26/07/2006 15:45:39 | use primer | primer1 | name = VIII.g12f<br>real size = 1500<br>sequence = CGATCCATTCATCACCACTG | |
| | | primer2 | name = VIII.g12r<br>real size = 1500<br>sequence = CAATGTGGAGTTCTGGCTTG | |
| 26/07/2006 15:45:47 | use reagent | reagent | amount = 1.5mM<br>name = MgCl2 | |
| 26/07/2006 15:45:52 | set temperature | temperature | value = 57/59 | |
| **Experiment started 26/07/2006 15:43:41 by pcrMaker (test actor)** | | | **Experiment type: SNP** | **Specific product** |
| 26/07/2006 15:43:59 | use primer | primer1 | name = VIII.g06f<br>real size = 1000<br>sequence = AACTCAAGCAACAGGCAGGT | |

Figure 4.4: "View experiments" interface

- Experiment type (SNP or Microsatellites);
- Name of the biologist who conducted the experiment;
- Primer sequence;
- Primer name;
- Gene name.

Experiment type matching criterion is a strict equality of the compared types.

The same is valid for the biologists' names: these are matching only if the compared strings are equal.

The other fields are compared using substring matching algorithm. That is, if the filtering string appears anywhere in the string being checked, then the filter is passed. For example, for primer sequence filtering criterion "`GCCAC`", both "`CACATGT`<u>`GCCAC`</u>`TTCTGGAG`" and "`CTATACTCT`<u>`GCCAC`</u>`GGACTGC`" sequences would pass the filter.

It should also be noted, that when any of the primer-related filtering field is enabled, the filter is passed by all the experiments where either of the used primers match the criterion.

Under the filter settings, there is a list of the experiments which are stored by the prototype. For each experiment, the following data is displayed:

- Experiment date and time;
- Experiment description;
- Name of the biologist;
- Type of the experiment;
- Result of the experiment;
- Detailed timestamped log of actions executed during the experiment, along with their parameters.

### 4.3.4 User management

As it already has been noticed before (section 3.1), although the prototype does not require an access control system (as there is only one group of users considered), it still needs some kind of user management, in order to tag experiments with appropriate biologists' names in a consistent and efficient manner.

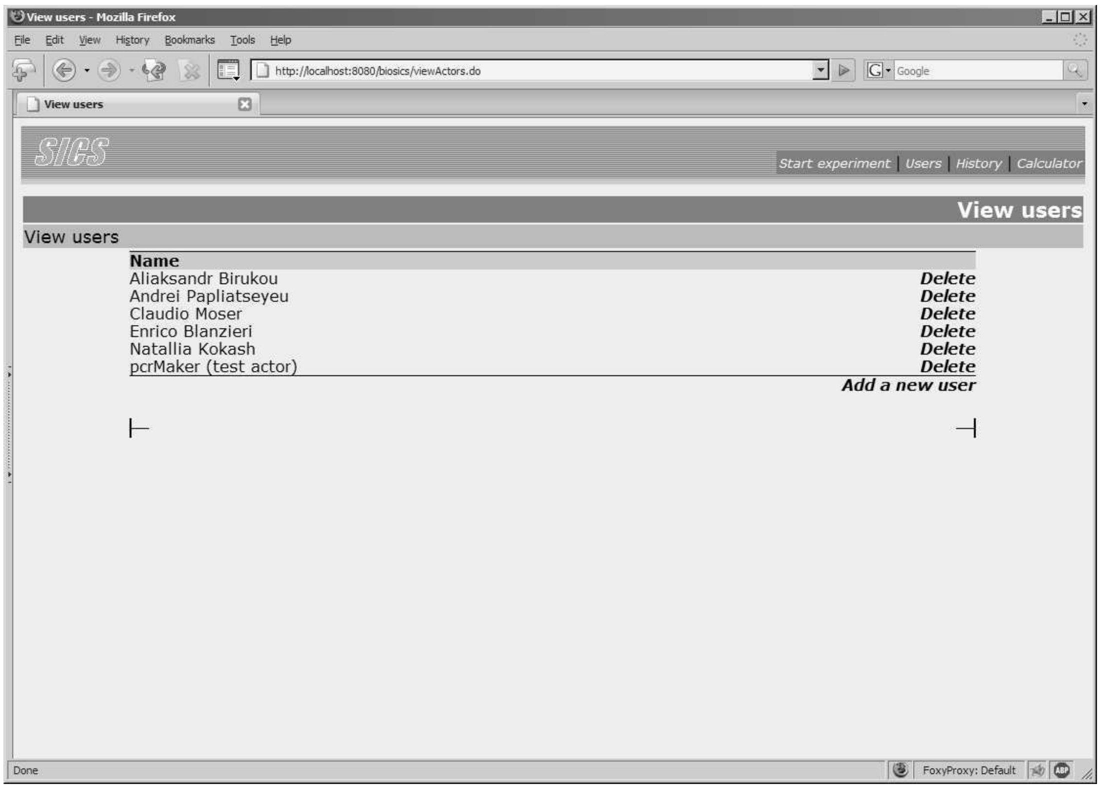
Figure 4.5 presents the interface of user management module of the prototype. Each user profile contains only name of the biologist who uses the application. Every line of the user list is followed by "Delete" button which allows one to remove the corresponding user profile from the application.

New user profiles can be added to the system by clicking "Add a new user". In this case, the next page prompts for a user name, and the new profile is saved in the SICS storage.

It should be noted that in terms of Implicit Culture new users are added both to G and G' groups which is due to the fact that it is generally impossible to divide users unambiguously into "experienced" and "unexperienced" groups (see section 2.3.2).

### 4.3.5 Temperature calculator

Temperature calculator is a simple utility for calculation of the appropriate melting temperature values for a given primer sequence, basing on the empirical formulas 1.1 and 1.2.


View users - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8080/biosics/viewActors.do
View users
SICS
Start experiment | Users | History | Calculator
Done
FoxyProxy: Default

View users

View users

| Name | |
|---|---|
| Aliaksandr Birukou | *Delete* |
| Andrei Papliatseyeu | *Delete* |
| Claudio Moser | *Delete* |
| Enrico Blanzieri | *Delete* |
| Natallia Kokash | *Delete* |
| pcrMaker (test actor) | *Delete* |
| | *Add a new user* |

Figure 4.5: “User management” interface

This module is just an independent shortcut to the temperature calculating functionality available as a part of in-experiment support. It has been added by biologists' request as they found it useful to have such a utility for quick checks and estimations.

## 4.4 Evaluation

Due to the very nature of this work, we have applied a qualitative evaluation approach with the elite perspective [44].

The prototype containing about thirty real experiment records has been presented to a small focus group of prospective users of the system for PCR support. The group included three experienced biologists from the laboratory of genomics of Instituto Agrario di San Michele all'Adige, Italy.

Experts' opinions have been collected by means of collective discussion and informal individual interviews.

The prototype has received a contradictory feedback.

The biologists generally liked the idea of having a complete shared searchable record of experimental results. However, the quality of suggestions has been assessed as “not useful”. The reasons for this seem to be the following.

- Insufficient attention has been paid to the positioning of the prototype, which caused over-realistic expectations among the biologists.

  The biologists expected the prototype to give original suggestions. However, the very idea of the system is to make the previous experiences available to a group of people. Thus, the biologists’ complain “It cannot suggest more than already known” perfectly depicts the limitation of the selected approach for recommendation inference.

- One-sided evaluation.

  The initial goal of the prototype was to assist both senior researchers and junior biologists and students who are not familiar enough with the PCR itself and local “rules of the thumb”. However, only senior researchers have participated in the evaluation of the prototype due to the seasonal absence of students in the laboratory.

  It is reasonable, that the experienced scientists do know the procedure, and they might only need information about some specific parameter’s value. Indeed, the interviewed biologists have been mostly concerned about the temperature values that depend on primer sequences. Therefore, they expected some exceptionally good suggestions on the temperature while the prototype was able to provide only the values the biologists themselves would try first.

- Premature conclusions, as a pitfall of the preliminary exploratory case study. Information needs of the biologists inferred from the on-site observations might require further refinements.

Summarizing the results of the prototype’s evaluation, we are in a position to conclude the following.

- The biologists have found the record keeping and filtering part of the prototype useful.

- The recommendation giving part requires further improvements in order to be useful for experienced biologists.

- The prototype “as is” might be useful for students and junior scientists for training and learning the details of PCR procedure.

## 4.5 Discussion

The prototype of the system for PCR support that has been described in this chapter, has helped us to identify a number of challenges concerning supporting of the biologists working in laboratory with PCR procedure.

In spite of having rather limited functionality, the prototype may already be used by students and junior biologists who want to get acquainted with PCR experiments. It also provides more advanced functionality than the biologists currently have available.

However, in order to make the system useful also for the experienced biologists, the recommendation producing part needs major improvements related to the actions' parameter estimation algorithms (especially, melting/annealing temperatures).

Although there are many empirical formulas for calculating primer's melting temperature, neither of them give better than $\pm 5 \ldots 10°\mathrm{C}$ ($8 \ldots 17\%$) accuracy [25].

One of the possible solutions is to adapt data mining techniques in order to infer a systematic relationship between primers' sequences and corresponding annealing temperature values. This approach requires a large amount of real experimental data which, in our best knowledge, is not easily available, probably because of security and copyright regulations of the laboratories.

Further improvements of the proposed system for PCR support may include:

- The system should use the information about failed experiments in order to improve the quality of suggested actions.
- It might be useful to distinguish experienced and inexperienced users whenever such a distinction is possible. This could also improve the quality of suggestions by rejection of the actions of definite novices.
- Another way of improvement is to go beyond text-only suggestions (for example, provide a short video of the sequence of required operations)
- Finally, one could apply the Implicit Culture approach to the system interface. For example, the system may dynamically modify its interface in order to make the most used interface elements more easily accessible for users.

# Conclusion

In this project, we addressed the information needs of the biologists working in a laboratory, with a special focus on Polymerase Chain Reaction (PCR) experiment.

The work outcomes have two aspects.

First, the paper presents an illustrative case study dedicated to the description of the biological laboratory work practices from the point of view of computer science and done with close interaction with biologists. This study facilitates better understanding of the interdisciplinary domain of bioinformatics by prospective researchers from the field of computer science.

The second is the practice-oriented aspect. Having analyzed the information needs of the biologists performing PCR, we elaborated a set of requirements to a system for PCR support and proposed an architecture of such system. The system adopts the Implicit Culture approach to make the personal knowledge of the biologists retrievable and to provide them with in-experiment support, basing on the history of actions performed by other biologists in similar situations. In order to test our expectations regarding the system, we have implemented a simplified prototype.

Evaluation of the prototype has discovered a new research challenge related to the accurate prediction of experimental parameters, specifically, annealing and melting temperatures of primers.

The prototype has proved itself relevant as a shared storage of experimental data with advanced functionality.

However, recommendation system of the prototype requires further improvements concerning the quality of suggestions, in order to make them useful for experienced biologists. Nevertheless, students and junior researchers may already benefit from the prototype's suggestions while learning the specifics of PCR practices in the laboratory.

The future work concerns improvement of the quality of suggestions provided by the system. The proposed approach is to apply data mining techniques to the real experimental data and attempt to infer implicit dependencies between experimental parameters. This approach may provide better prediction quality of the correct values than the current rules-of-thumb and empirical formulas.

# Bibliography

[1] URL http://scholar.google.com.

[2] Apache Struts framework. URL http://struts.apache.org/.

[3] Java2 enterprise edition. URL http://java.sun.com/j2ee/1.4/docs/.

[4] LightCycler one-step RT-PCR systems: Choose the right kit for your RT-PCR application. URL http://www.roche-applied-science.com/PROD_INF/BIOCHEMI/No1_01/PDF/b101oberlaender.pdf.

[5] Paper++. URL www.paperplusplus.net.

[6] Primer3 – PCR primer design tool. URL https://sourceforge.net/projects/primer3.

[7] Real-time compendum 7000 SDS ver 3.0. URL http://www.appliedbiosystems.com.au/7000\%20RT\%20Compendium\%20v3.0.pdf.

[8] Teranode corp. URL http://www.teranode.com.

[9] Tuffts University. School of Medicine. PCR fees. URL http://www.tufts.edu/med/teac/fees.html.

[10] Mark S. Ackerman and David W. McDonald. Answer garden 2: merging organizational memory with collaborative help. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 97–105, New York, NY, USA, 1996. ACM Press. ISBN 0-89791-765-0. doi: http://doi.acm.org/10.1145/240080.240203.

[11] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, 1994.

[12] H. Ailisto, P. Alahuhta, V. Haataja, V. Kyllönen, and M. Lindholm. Structuring Context Aware Applications: Five-Layer Model and Example Case. *Workshop in Ubicomp, Gothenburg, Sweden*, 2002.

[13] L. Arnstein, R. Grimm, C.Y. Hung, J.H. Kang, A. LaMarca, S.B. Sigurdsson, J. Su, and G. Borriello. Systems Support for Ubiquitous Computing: A Case Study of Two Implementations of Labscape. *Proceedings of the 2002 International Conference on Pervasive Computing*, pages 30–44, 2002.

[14] L. Arnstein, Chia-Yang Hung, R. Franza, Qing Hong Zhou, G. Borriello, S. Consolvo, and Jing Su. Labscape: a smart environment for the cell biology laboratory. *Pervasive Computing, IEEE*, 1(3):13–21, 2002.

[15] L. Arnstein, S. Sigurdsson, and R. Franza. Ubiquitous computing in the biology laboratory. *Journal of Lab Automation (JALA)*, 6(1), 2001.

[16] Shane Back. Do you have a license?: Products licensed for PCR in research applications. *The Scientist*, 12(12):21, 1998.

[17] P. Bahl and VN Padmanabhan. RADAR: an in-building RF-based user location and tracking system. *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2:775–784, 2000.

[18] XaoMing Bao, See-Kiong Ng, Eng-Huat Chua, and Wei-Khing For. Virtual lab dashboard: Ubiquitous monitoring and control in a smart bio-laboratory. *Lecture notes in computer science*, pages 1167–1176, 2005.

[19] R. Bentley, W. Appelt, U. Busbach, E. Hinrichs, D. Kerr, K. Sikkel, J. Trevor, and G. Woetzel. Basic support for cooperative work on the world wide web. *International Journal of Human Computer Studies*, 46:827–846, 1997. URL `citeseer.csail.mit.edu/bentley97basic.html`.

[20] Aliaksandr Birukou, Enrico Blanzieri, Vincenzo D'Andrea, Paolo Giorgini, Natallia Kokash, and Alessio Modena. ICService: A Service-Oriented Approach to the Development of Recommendation Systems. *The 22nd Annual ACM Symposium on Applied Computing*, 2007. (accepted for publication).

[21] A. Birukov, E. Blanzieri, and P. Giorgini. Implicit: an agent-based recommendation system for web search. *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 618–624, 2005.

[22] E. Blanzieri and P. Giorgini. Implicit culture for information agents. *Lecture Notes in ArtificialIntelligence*, 2586:152–164, 2003.

[23] E. Blanzieri, P. Giorgini, P. Massa, and S. Recla. Implicit Culture for Multi-agent Interaction Support. *CooplS01: Proceedings of the 9th International Conference on Cooperative Information Systems*, 2172:27–39, 2001.

[24] Douglas P. Bogia and Simon M. Kaplan. Flexibility and control for dynamic workflows in the worlds environment. In *COCS '95: Proceedings of conference on Organizational computing systems*, pages 148–159, New York, NY, USA, 1995. ACM Press. ISBN 0-89791-706-5. doi: http://doi.acm.org/10.1145/224019.224034.

[25] Stuart M. Brown. Using Computers for Molecular Biology. PCR & Sequencing Primer Design. NYU Medical Center. URL `http://www.med.nyu.edu/rcr/rcr/course/primer.html`.

[26] S. A. Bustin. Absolute quantification of mRNA using real-time reverse transcription polymerase chain reaction assays. *Journal of Molecular Endocrinology*, 25: 169–193, 2000.

[27] BioMath Calculator. $T_m$ Calculations for Oligos. URL `http://www.promega.com/biomath/calc11.htm`.

[28] Paul Dourish. *Open implementation and flexibility in CSCW toolkits*. PhD thesis, Department of Computer Science, University College London, June 1996.

[29] Willard M. Freeman, Stephen J. Walker, and Kent E. Vrana. Quantitative RT-PCR: Pitfalls and potential. *BioTechniques*, 26(1):112–125, 1999.

[30] H.T. Gao, JH Hayes, and H. Cai. Integrating Biological Research through Web Services. *Computer*, 38(3):26–31, 2005.

[31] C.S. Gillespie, C.J. Proctor, D.P. Shanley, D.J. Wilkinson, R.J. Boys, and T.B.L. Kirkwood. Web-services for the biology community: the BASIS project. In *UK eScience All Hands Meeting*, 2005.

[32] F. Guimbretière. Paper augmented digital documents. *Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 51–60, 2003.

[33] Saul Halfon. Collecting, testing and convincing: Forensic DNA experts in the courts. *Social Studies of Science*, 28(5/6):801–828, October-December 1998. Special Issue on Contested Identities: Science, Law and Forensic Practice.

[34] H. Hile, J. Kim, and G. Borriello. Microbiology tray and pipette tracking as a proactive tangible user interface. *Proc. of the 2nd Int. Conf. on Pervasive Computing*, pages 323–339, 2004.

[35] J. Indulska and P. Sutton. Location management in pervasive systems. *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003-Volume 21*, pages 143–151, 2003.

[36] K. Jordan and M. Lynch. The dissemination, standardization and routinization of a molecular biological technique. *Social Studies of Science*, 28:773–800, 1998.

[37] W. E. Mackay, G. Pothier, C. Letondal, K. Boegh, and H. E. Sorensen. The missing link: augmenting biology laboratory notebooks. In *Proceedings of the 15th Annual ACM Symposium on User interface Software and Technology*, pages 41–50, Paris, France, October 2002. ACM Press, New York.

[38] Xiu mei Wei and Zhi min Yang. The memory of historical data in CSCW. In *The 8th International Conference on Computer Supported Cooperative Work in Design Proceedings*, pages 18–21. IEEE, 2003.

[39] K. Mullis, F. Faloona, S. Scharf, R. Saiki, G. Horn, and H. Erlich. Specific enzymatic amplification of DNA in vitro: the polymerase chain reaction. *Cold Spring Harbor Symp. Quant. Biol.*, 51:263–273, 1986.

[40] Jeffrey M. Perkel. The trouble with kits. *The Scientist*, Volume 20(3):73, 2006.

[41] J.R. Quinlan and R.M. Cameron-Jones. FOIL: A midterm report. *Proceedings of the European Conference on Machine Learning*, pages 3–20, 1993.

[42] M. Sarini, E. Blanzieri, P. Giorgini, and C. Moser. From actions to suggestions: supporting the work of biologists through laboratory notebooks. In *Proceedings of 6th International Conference on theDesign of Cooperative Systems (COOP2004)*, pages 131–146, French Riviera, France, 2004. IOSPress.

[43] Jonathan Trevor, Tom Rodden, and Gordon Blair. COLA: a lightweight platform for CSCW. *Computer Supported Cooperative Work*, 3(2):197–224, 1995. ISSN 0925-9724. doi: http://dx.doi.org/10.1007/BF00773447.

[44] Wikipedia. Evaluation approaches, . URL `http://en.wikipedia.org/wiki/Evaluation_approaches`.

[45] Wikipedia. Naturalistic observation, . URL `http://en.wikipedia.org/wiki/Naturalistic_observation`.

[46] Wikipedia. Polymerase Chain Reaction, . URL `http://en.wikipedia.org/wiki/PCR`.

[47] R. Yeh, C. Liao, S. Klemmer, F. Guimbretière, B. Lee, B. Kakaradov, J. Stamberger, and A. Paepcke. ButterflyNet: a mobile capture and access system for field biology research. *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 571–580, 2006.