Mutational data loading routines for human genome databases: the BRCA1 case.

Matthijs van der Kroon, Ignacio Lereu Ramirez, Ana M. Levin, Oscar Pastor, Sjaak Brinkkemper

Technical Report 2009-020
September 2009

Department of Information and Computing Sciences
Utrecht University, Utrecht, The Netherlands
www.cs.uu.nl

ISSN: 0924-3275

Department of Information and Computing Sciences
Utrecht University
P.O. Box 80.089
3508 TB Utrecht
The Netherlands

MUTATIONAL DATA LOADING ROUTINES FOR HUMAN GENOME DATABASES: THE BRCA1 CASE.

Matthijs van der Kroon, Ignacio Lereu Ramirez, Ana M. Levin, Oscar Pastor, Sjaak Brinkkemper

Abstract

This technical report contributes a framework for genomic mutational data problems, as well as insights on the automation of genomic mutational data loading into an existing database. This report describes the process of loading genomic mutational data for the human BRCA1 gene into an existing database. The encountered problems affiliated with this procedure were documented and a standard notation for them has been devised in order to facilitate more efficient research in the future.

TABLE OF CONTENTS

PREFACE

This paper came to be as a result of the collaboration between the technical university of Valencia, la Universidad Politécnica de Valencia, and the University of Utrecht as part of the curriculum of Informational Science in Utrecht. The receiving department in Valencia was el departamento de Sistemas Informáticos y Computación (DSIC), and the involved department in Utrecht was the department of Information and Computing science. This temporary collaboration took place from the 19th of April 2009, until the 31st of July 2009.

The training provider in Valencia was:

Name	Prof. O. Pastor López
Address	Facultad de Informática, Universidad Politécnica de Valencia Camino de Vera, s/n 46022 Valencia, Valencia, Spain
Phone	+34 9638 77000 ext. 83508
Fax	+34 9638 77359
E-mail	opastor@dsic.upv.es

The training supervisor in Valencia was:

Name	Dr. A. Levin
Address	Facultad de Informática, Universidad Politécnica de Valencia Camino de Vera, s/n 46022 Valencia, Valencia, Spain
Phone	+34 9638 77000 ext. 83594
Fax	+34 9638 77359
E-mail	alevin@dsic.upv.es

The training provider in Utrecht was:

Name	Prof. S. Brinkkemper
Address	Department of Information and Computing science, Universiteit Utrecht P.O. Box 80.089 3508 TB Utrecht the Netherlands
Phone	+31 30 253 3175
Secr	+31 30 253 1454
E-mail	s.brinkkemper@cs.uu.nl

1. INTRODUCTION

The last few years, benefits of advances in the genetics field in general have become increasingly evident. The most obvious advantage of these being the improvement of human health in general by eliminating approximately 2.000 genetic diseases like Huntington's chorea and dozens of cancers. Examples in other fields stated by [*Coates, 1997*] include solving the global famine problem by increasing agricultural efficiency and solving many environmental problems.

1.1 GENETICS

A genome defines as the complete collection of genes that serves as a blueprint for that particular organism. Almost every human cell has a copy of this hereditary database contained in it's cell-nucleus, which probably consists of around 25.000 genes in total. About the exact number of genes exists a lot of debate and recent estimates vary from 20.000 to 25.000 [*Collins, 2004*] and [*Gerstein, 2007*].

1.2 DNA STRUCTURE

The chemical structure holding this hereditary information is called deoxyribonucleic acid, or DNA. The syntax in which the genetic code is written, consists of 4 elements; A, C, T and G denoting particular chemicals, referred to as nucleotides, or bases. Each of these nucleotides comprises of 3 components; a phosphate group, a five-carbon sugar and a nucleobase. The phosphate group and the sugar form a backbone structure while the nucleobase defines the nucleotide denotation, or meaning. 4 different nucleobases exist; Adenine, Cytosine, Thymine and Guanine, hence the nucleotide identifiers. In DNA the phosphate groups of each nucleotide bond with the sugar of the next nucleotide forming a sequence of nucleotides in that process.

At the same time, nucleobases adhere to each other in a specific way: A only bonds to T and C only bonds to G. Laid out in 2D as depicted in [*figure 1*], the structure

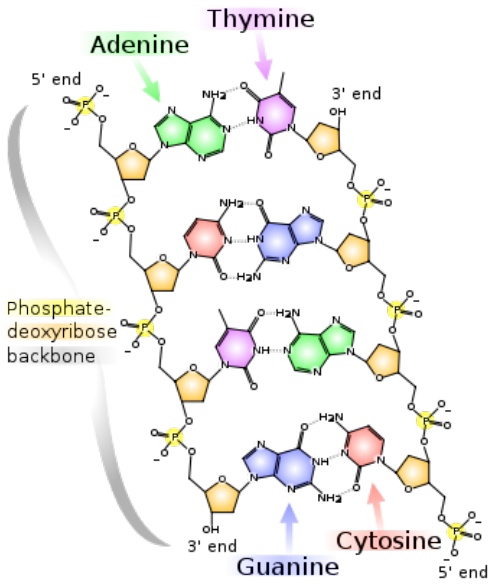


Figure 1: 2D representation of the DNA molecule

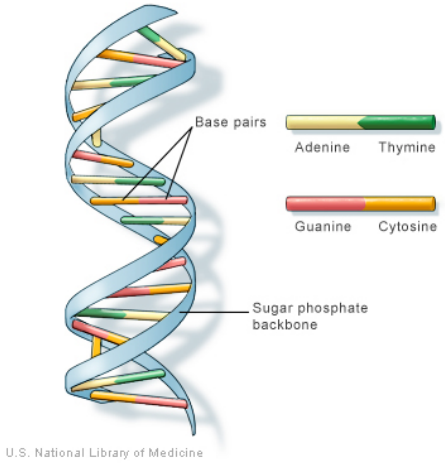


Figure 2: 3D representation of the DNA molecule.

formed by these bonds consists of two phosphate-deoxyribose backbones with pairs of nucleobases in between. In 3D the molecule looks like the well known helix, as it is shown in [*figure 2*]. The phosphate-deoxyribose backbones are often referred to as strands and since the nucleotides adhere to each other in only one manner, the strand sequences are complementary to each other. This phenomenon is often referred to as one strand sequence being sense and the other anti-sense. The sequence on the strand that is transcribed to mRNA is called the 'sense' sequence, while the sequence on the opposite strand is called the 'anti-sense' sequence. The sense strand then is denoted with a '–' symbol, while the '+' symbol indicates a non-sense strand. Both sense and anti-sense sequences can exist on different parts of the same strand of DNA. Since the nucleotides bond in an asymmetrical way, sugar to phosphate, DNA has a direction. In the helix, the direction of one strand is opposite to the direction of the other. The strand ends are referred to as the 5' and 3' ends, where the 5' corresponds to the end with a terminal phosphate group and the 3' to the end with a terminal sugar group. Nevertheless, genomes rarely exist as one uninterrupted DNA string, in *homo sapiens* for example the hereditary information is divided over 23 pairs of DNA strings, commonly referred to as chromosomes.

Contrary to what would be expected however, only a small part of the total genome sequence codes for genes. About 95% of the human genome has been designated 'junk' in the past and percentages representing coding parts of the genome ranged from 1.1% [*Venter, 2001*] to less then 5% [*Lander, 2001*]. The non-coding parts of the genome were a long time considered to be evolutionary artifacts, serving no present day function. However, recent research shows the non-coding parts of the genome might actually be fulfilling functions, not yet well understood [*Mattick, 2003*], [*Mattick, 2004*] and [*Ahnert, 2008*].

EXPRESSION

The genetic representation on the DNA of a particular trait, hence a specific nucleotide sequence, is referred to as the genotype, whilst the observable characteristics like hair or eye color are being referred to as the phenotype. Thus, for a gene moving from a genotype to

a phenotype it has to be involved in a process referred to as expression.

During expression the message undergoes various operations, eventually resulting in an observable trait. First, it has to be transcribed from DNA to a message carrier called RNA, in what is called transcription. The chemical structure of RNA is very similar to that of DNA. During this process, a DNA sequence denoted by a promotor and a terminator sequence is copied onto the RNA by a molecular copying machine called Polymerase II.

The RNA string, which is called at this stage the primary transcript, then undergoes a process called splicing [*Graveley, 2001*], during which these earlier mentioned non-coding sequences, introns, are discarded and the resulting coding pieces, exons, are rejoined. This splicing process eventually results in a continuous coding sequence, but which exons are in- or excluded in the final message may vary. Consequently, different ways of splicing the same gene, result in different proteins, en thus different phenotypes. This feature explains how it is possible for the ~25.000 genes to produce the ~100.000 proteins of which the human proteome consists. [*figure 3*] visualizes the splicing process just described.

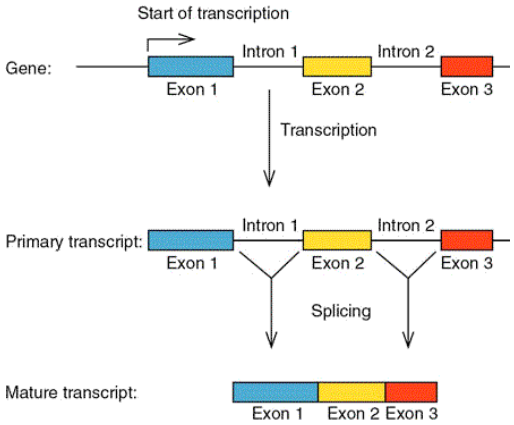


Figure 3: RNA splicing.

Eventually, some more processing is done to ensure the stability of the RNA molecule and ultimately, the message leaves the cell-nucleus as what is now called messenger RNA (mRNA) and is converted by the ribosome to the appropriate protein in a process called translation. The ribosome is essentially a protein factory on a molecular scale. Some proteins dictate phenotype on their own, but by far the most traits are a result of multiple proteins collaborating in what is known as Protein-Protein Interaction, or PPI [*Stelzl, 2005*].

This has been a rather limited introduction to the very complex field of genetics, for a more detailed account about genetics consult [*Alberts, 2003*].

1.3 THEORETICAL BACKGROUND

The last decades a large amount of research has been done in the genetics field, not the least by the Human Genome Project, [*Olson, 1993*], [*Venter 2001*] and [*Lander, 2001*], which has and is generating Terabytes, if not Exabyte's, of information that is stored globally in a very fragmented way. The NCBI and HGMD databases are examples of these data repositories and will be discussed shortly in the 'methods used' section. However, different databases use different ways of storing the same data, resulting in undesired redundancy and restrained information transfer. Adding to this, keeping the existing databases consistent is mainly left to human intervention, which in turn is very costly and error prone. This is where informational science enters the game with it's wide-ranging experience in structuring and modeling real-world data. This experience allows for a more effective and efficient digital representation, a process referred to as normalization [*Simsion, 2005*]. [*Bornberg-Bauer, 2002*] and [*Graves, 1996*] provide suggestions on how this knowledge could be put to good use in the Bioinformatics field.

An informational approach to this specific biological problem space isn't entirely new. [*Okayama, 1998*] describes the conceptual schema of a DNA database using an extended entity-relationship model. [*Chen, 1995b*] has indicated how an extended object data model can be used to capture the properties of scientific experiments, and [*Medigue, 1999*] includes models for representing genomic sequence data. [*Paton, 2000*] advanced on these efforts by presenting a first effort in conceptually modeling the *S. cerevisiae* genome, which is a type of yeast, by proposing a collection of conceptual data models for genomic data. Among these conceptual models are a basic schema diagram for genomic data, a protein-protein interaction model, a model for transcriptome data and a schema to modeling alleles. Whereas [*Paton, 2000*] provides a broader view by presenting conceptual models for describing both genome sequences and related functional data sets, [*Pastor, 2009a*] further elaborated on the basic schema diagram for genomic data thereby narrowing the focus and specializing it for the human genome and eventually produced a database (CSHGDB) corresponding to this model and following the standard rules of logical design.

This database is now in the prototype phase and the first gene (NF1) has been partially inserted. The insertion of this very first gene exposed many problems of the initial conceptual model and hence led to just as many modifications to it. [*Pastor, 2009c*] describes this process more specifically by providing an overview of how the CSHGDB and HGMD databases correspond to each other, at the same time uncovering the evolution of CSHGDB by matching HGMD to it. [*Pastor, 2009b*] describes the evolution of the model more in general and provides a descriptive overview of how the model came to be, and from where it evolved to what it is now. The present day model will be discussed in chapter 4: the CSHGDB conceptual model.

1.4 PROBLEM SPACE

Looking from an informational point of view, the human genome is an extremely complex system in which exists a lot of ambiguity. For example, basic concepts of what exactly defines a gene weren't explicitly described by biologists, leading to confusing situations. To illustrate the consequence of such a lack of definition, imagine a library. This library consists of numerous books, each one of those possessing various properties, like title, author, ISBN number and so on. Each book can then be identified uniquely by, for instance, a single ISBN number.

Retrieving this identifying property in the case of genes is much more challenging. Of course, a gene could simply be interpreted as a specific nucleotide sequence starting with a promoter sequence and ending with a terminator sequence. And this is basically how the concept was defined until the end of the 1970s. [*Gerstein, 2007*] describes the evolving definition of the gene as a concept in more detail. However the existence of introns and the process of splicing complicates this view as is put very well by [*Scherren, 2007*]: "in eukaryotes, the gene is, in most cases, not yet present at DNA-level. Rather, it is assembled by RNA processing". Thus, different ways of splicing the same coding DNA sequence, leads to different phenotypes [*Gerstein, 2007*].

Another way of looking at a gene, would be to identify it by it's function, or more accurately, how it is expressed. However, since the majority of traits is actually a result of multiple gene combinations, this view isn't adequate either. For example, the phenotype eye color, is actually a result of 16 different genes.

1.5 SCIENTIFIC USE

Obtaining a complete conceptual model of the human genome holds great potential. Not only will it provide a framework, enabling efficient and effective access to the genomic data, thereby offering ways of reusing previously researched data by pharmaceutic and medical industries as mentioned by [*Pastor, 2008*]. It might also completely change the way conventional Bioinformaticians look at genomic data.

To present day, most Bioinformatics research is located in the solution space, by attempting to interpret the data that comes 'out of the black box'. For instance by applying powerful search tools like BLAST [*Pertsemlidis, 2001*]. However the team led by professor Pastor mainly directs it's efforts at tracing the processes effectively leading to these data. Essentially, seen from an informatics point of view, professor Pastor and his team are trying to find the source-code, by analyzing the object-code, of what may very well be the most sophisticated software ever to be analyzed: life itself.

1.6 RESEARCH QUESTIONS

To populate CSHGDB with the roughly 25.000 genes, a manual approach would be too expensive, time-consuming and error prone. Hence an automated approach is required. Before this automated approach can be put to good use, one has to explore how the data behaves within the existing conceptual model. Thus a list of 20 genes has been composed, to be inserted in the database manually, so that the conceptual model can be adjusted according to the obtained insights. All genes on this list are related to common disorders, since it is in these kind of genes benefits of the Genoma project will directly be appreciable.

The first gene, NF1, has been inserted partially, and plays a significant role in the Neurofibromatosis hereditary disease. The BRCA1 gene, playing an important role in producing so-called tumor suppressor proteins, is next on the insertion list and the main subject of this paper. The gene will be discussed in more detail shortly in the 'the BRCA1 gene' section.

This paper will mainly focus on the process of inserting the genomic mutational data of this second gene into CSHGDB, thereby analyzing the degree to which mapping is possible to the current conceptual model. This leads to the main research question:

To what degree is it possible to map the BRCA1 gene to the existing conceptual model, compared to the NF1 gene?.

The deliverable for this research question will be a complete a possible loaded CSHGDB database with BRCA1 genomic mutational data. Also will the loading procedure of the BRCA1 gene be compared to the NF1 gene's loading procedure, [*chapter 6: BRCA1 and NF1 comparison*] will discuss this in more detail.

The BRCA1 loading procedure will then be related to the NF1 insertion, comparing the found problems and solutions to those found with the NF1 insertion. This leads to sub research question 1:

To what degree do the mutational data loading problems encountered during the BRCA1 gene insertion resemble the ones found with the NF1 gene?

and sub research question 2:

To what degree do the solutions found to the BRCA1 gene mutational data loading problems resemble the ones found with the NF1 gene?.

The deliverable to sub research question 1 and 2 will be an analysis of the BRCA1 gene mutational data loading encountered problems, the underlying causes and their solutions. [*chapter 5: findings*] will be covering this subject. By uncovering the answers to sub research questions 1 and 2 a more accurate conceptual model can be identified. Also will be analyzed whether the loading of genomic data in the database can be retraced to a common procedure, which in turn might eventually allow for an automated approach. This leads to the identification of sub research question 3:

Can the loading of the genome database from available gene information be based on a common procedure?

and sub research question 4:

Can the underlying loading procedure be automated?

Sub research question 3 and 4 will be discussed in detail in [*chapter 7: conclusions*].

2. METHODS USED

Various sources were used in the process of populating CSHGDB with the BRCA1 gene. First of all, the conceptual model [*appendix 1: conceptual model*] created by the team was used as a reference tool.

The data used to populate CSHGDB with BRCA1, is acquired mainly from two databases: NCBI [*NCBI, 2009*] and HGMD [*HGMD, 2009*]. Further more, the HGNC database [*HGNC, 2009*] is used for universal naming conventions.

2.1 USED GENOMIC DATA SOURCES

In the following section the genomic data sources used for populating the CSHGDB database will be discussed.

NCBI

The following section was taken from the NCBI website [*About NCBI 2004*] and describes it's goals:

"Established in 1988 as a national resource for molecular biology information, NCBI creates public databases, conducts research in computational biology, develops software tools for analyzing genome data, and disseminates biomedical information - all for the better understanding of molecular processes affecting human health and disease."

HGMD

The following section was taken from the HGMD website [*HGMD background 2007*] and describes the organization in general and it's goals:

"The Human Gene Mutation Database (HGMD) represents an attempt to collate known (published) gene lesions responsible for human inherited disease. This database, whilst originally established for the study of mutational mechanisms in human genes (Cooper and Krawczak 1993), has now acquired a much broader utility in that it embodies an up-to-date and comprehensive reference source to the spectrum of inherited human gene lesions. Thus, HGMD provides information of practical diagnostic importance to (i) researchers and diagnosticians in human molecular genetics, (ii) physicians interested in a particular inherited condition in a given patient or family, and (iii) genetic counselors."

HGNC

The following section, taken from the HGNC website [*About the HGNC 2007*], explains HGNC's reason for existence and it's main goals:

"Problems of nomenclature in human genetics were recognized as early as the 1960s and in 1979 full guidelines for human gene nomenclature were presented at the Edinburgh Human Genome Meeting (HGM). Since then we have continued to strike a compromise between the convenience and simplicity required for the everyday use of human gene nomenclature and the need for adequate definition of the concepts involved."

[*Pastor, 2009b*] and [*Pastor, 2009c*] were used as reference guides; the first describes the process of mapping the HGMD mutations data to CSHGDB while the latter provides a more general explanation about how CSHGDB should be populated by explaining the concepts in the model.

2.2 PROCESS DESCRIPTION

During this project a number of steps will be taken before achieving the stated goals. Since the main purpose of the project is to load the BRCA1 gene mutational data from HGMD to CSHGDB the process description is mainly directed at this effort. [*figure 4*] provides a process deliverable diagram on this process.

First of all, the sources will be analyzed to get a general idea of how the sources work and to get comfortable with their structure. Then CSHGDB itself will be investigated, for that very same purpose. When all data structures are clear, the manual loading of some entries

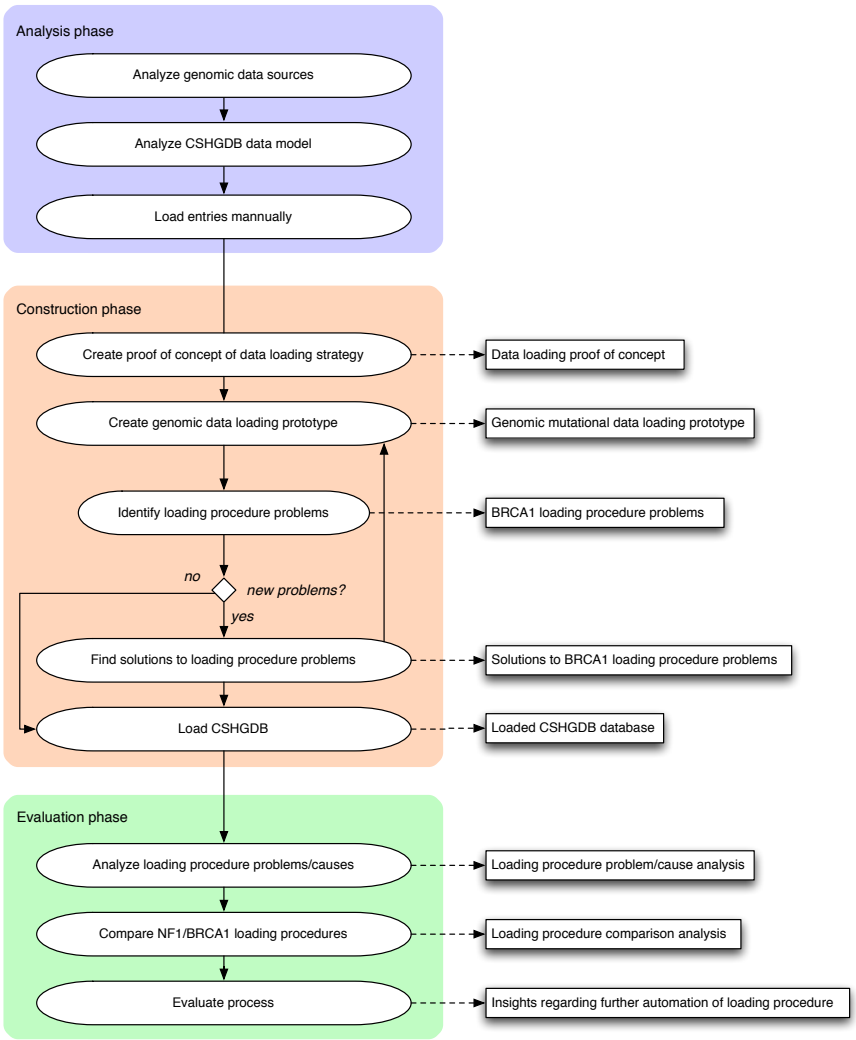


Figure 4: Process Deliverable Diagram of this project.

into CSHGDB can take place. This manual loading helps in getting a feeling for the data in order to understand the way it behaves in the database. After loading some entries into CSHGDB it is expected that some common actions can be identified and eventually integrated into a proof-of-concept. This proof of concept will be a PHP/MySQL script. The process then enters a systems development life cycle (SDLC), during which problems are encountered. These problems will need to be resolved, thereby leading to modifications to the existing code and eventually leading to an evolved final, working solution. Given these properties, it is expected an evolutionary prototyping approach will be most effective. Ultimately, the encountered problems, being documented during the process that uncovered them, will be analyzed together with uncovering what caused them. After the problem analysis a comparison will be performed on the NF1 and BRCA1 genes in order to identify common features providing insights on how to further automate the process. These insights, added to the general insights obtained during the rest of the process, will be discussed in the conclusions section [*chapter 7: conclusions*].

3. THE BRCA1 GENE

During a recent survey performed by [*Martí, 2007*] in the city of Valencia, Spain the frequency of various cancer forms amongst the Valencia population was investigated along with mortality rates . According to this research 70.7 out of 100.000 women were diagnosed with breast cancer in Valencia during the year 2004, which is significantly lower than the European mean of 110.3 for that same year. This accounts for 28.4% of the total amount of cancer cases in women [*figure 5*]. Since breast cancer is considered to be a well-treatable form of cancer providing it is diagnosed at an early stage, mortality rate was found to be relatively low at 16.7% [*figure* 6]. However, due to the relative high incidence of breast cancer, it accounts for the greatest part of all cancer mortality at 19.1%. Usually, breast cancer in humans is caused by genetic mutations in somatic breast cells. However breast cancer is not always a matter of coincidence, and sometimes susceptibility to the disease is inherited. As a matter of fact, according to [*Martí, 2007*], ~7% of all breast cancer cases are a result of hereditary susceptibility. To this day, what is known about this heredity is that it is dictated to great extend by two genes; BRCA1 and BRCA2 [*figure 7*]. The BRCA1 hereditary factor was first traced back to chromosome 17 locus q21 by [*Hall, 1990*] and later specifically to BRCA1 by [*Miki, 1994*] and [*Chen, 1995a*]. BRCA1 spans 22 coding exons, spanning 80 kb of genomic DNA, and has a 7.8 kb transcript coding for an 1863 amino acid protein. Both of the mentioned genes are involved in maintenance of genome stability, but given the fact the paper converges on BRCA1, a short explanation on this specific genes' working procedure will be given.

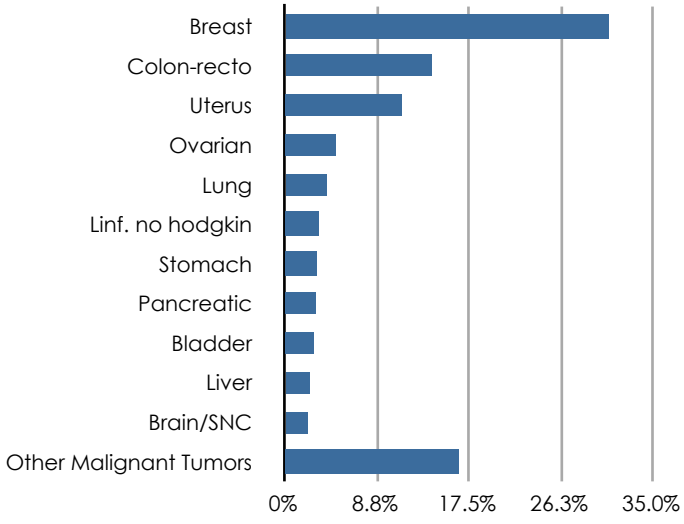


Figure 5: Cancer incidence in women in Valencia, Spain during the year 2004. Expressed in absolute percentages. [Martí, 2007]

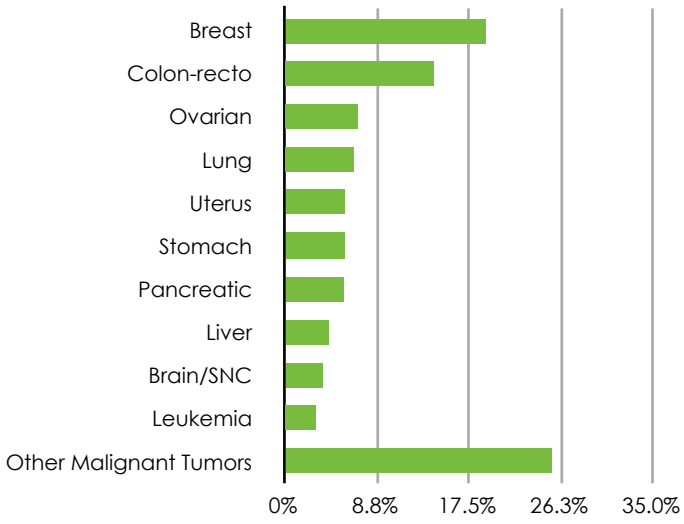


Figure 6: Breast cancer mortality rate as measured in Valencia, Spain during the year 2004. Expressed in absolute percentages. [Martí, 2007]

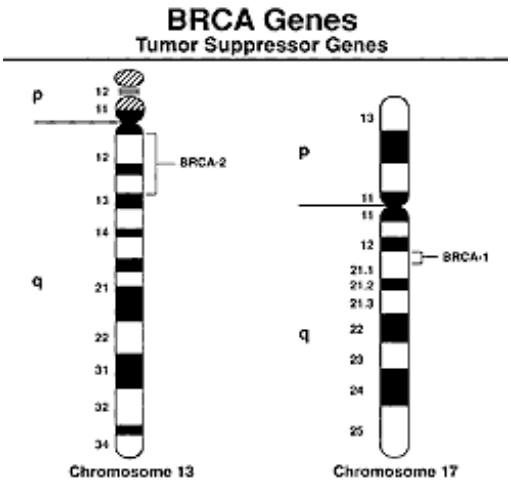


Figure 7: BRCA gene locations.

3.1 WHAT DOES IT DO?

BRCA1 actually is, contrary to what could be concluded from the name and the fact that 'it is involved in breast cancer development', not a gene somehow coding for breast cancer. It's modus operandi is actually more subtle, and explains why there is no 100% certainty anyone with a mutated BRCA1 gene will indeed develop a form of cancer. In reality BRCA1 encodes a tumor suppressing protein, which in turn binds with other tumor suppressors, DNA damage sensors and signal transducers to form a large protein complex. This large protein complex, known as BASC, associates with polymerase II and thereby affects transcription, DNA repair and recombination. Thus, a mutated BRCA1 gene accounts for greater susceptibility to cancer, more specifically breast and ovarian cancer, due to the fact that cells are less capable of repairing damaged DNA.

3.2 MUTATIONS

Due to the fact the HGMD database for academic use is not as exhaustive as the commercial one, at the time of writing (14/07/2009) only 894 mutations can be looked into. The commercial version grants access to a total of 1194 mutations. According to the distribution of the mutations shown in [*table 1*] the greater part of mutations in BRCA1 are missense/nonsense (35.79%) and Small deletions (32.66%), together accounting for 612 (68.45%) of the total 894 recorded mutations [*table 1*].

Mutation type	Amount	Relative
Missense/nonsense	320	35.79%
Splicing	80	8.95%
Regulatory	0	0.00%
Small Deletions	292	32.66%
Small Insertions	101	11.30%
Small Indels	11	1.23%
Gross Deletions	65	7.27%
Gross Insertions	16	1.79%
Complex Rearrangements	9	1.01%
Repeat Variations	0	0.00%
Total	**894**	100.00%

Table 1: BRCA1 mutations distribution according to HGMD as of 14/07/2009.

3.3 PHENOTYPES

Mutations of the BRCA1 gene can result in various phenotypes, including breast cancer [*Vallon-Christersson, 2001*], ovarian cancer [*Vallon-Christersson, 2001*] and pancreatic cancer [*Gallinger, 2008*]. The absolute risk of cancer by the age of 70 years attributable to a BRCA1 mutation is reported to be between 46% [*Satagopan, 2001*] and 87% [*Ford, 1994*] for breast cancer and between 40% [*Parmigiani, 2007*] and 44% [*Ford, 1994*] for ovarian cancer [*figure 8*]. The estimates of elevated risks for other cancers in BRCA1 carriers have been researched by [*Thompson, 2002*] and vary from a 0.05% elevation for bone cancer to a 4.06% elevation for liver cancer.

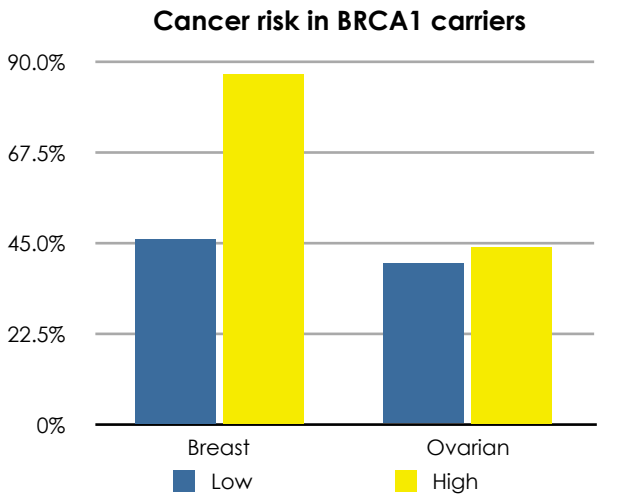


Figure 8: Absolute risk of cancer by the age of 70, attributable to a BRCA1 mutation.

4. THE CSHGDB CONCEPTUAL MODEL

In the beginning of the project, the team created an initial model, essentially describing how the concepts should be. However, as the team started matching data from various external sources to this ideal model (*appendix 1: conceptual model*), it soon became clear there existed a dichotomy between the ideal model and the way data was presented in real life. Thus a second model was created, logically named the 'real model' (*appendix 1: conceptual model*). It is anticipated the real model will eventually approximate the ideal model as much as possible. The conceptual model can be consulted in [*appendix 1: conceptual model*]. This section will describe the conceptual model as a whole, and therefore can best be related to as the ideal model.

The ideal model itself exists of three sub views; (i) the gene-mutation view, (ii) the genome view and (iii) the transcription view. At the time of writing the main focus of the research was pointed at the gene-mutation view, as a result the following section will primarily handle this part of the model.

As a result of the gene-mutation view dimensions, it will be discussed in 2 parts; (a), and (b). [*figure 9*] depicts part (a) and will cover mostly the main concepts of the model; hence the Gene, Allele and Allelic Reference Type entities. [*figure 10*] then illustrates part (b) thereby continuing with the mutational information including the Variation, Precise and Imprecise entities.

4.1 MAIN CONCEPTS

See for an overview of the main concepts [*figure 9*].

The Gene class models the generic concept of a gene: ID_symbol represents an alphanumeric code for the gene according to HGNC [*HGNC, 2009*], it also functions as the primary key; ID_HUGO, a numeric code assigned to the gene by HGNC; Official_name, the full name of the gene; Summary, a short description; Chromosome, the chromosome on which the gene is located and Locus, representing the location of the gene within the chromosome.

The Allele class then represents the various instances of the Gene concept. It stores information about the start and the end of the allele; the start_position and end_position attributes respectively, and which strand ('+' or '-') it is located: the strand attribute. The ord_num attribute functions as an internal identifier.

An allele can either be an Allelic Variant or an Allelic Reference Type. The two entities contain the same sequence attribute which represents a nucleotide string. Each gene has one reference allele associated with it, which is obtained from external sources. The Allelic Variant tuples then, are derived from this allele reference in combination with information contained in the Variation entity, referred to as 'Changes' in [*figure 9*]. The Variation entity will be discussed shortly in the 'mutational concepts' section.

The Data Bank entity holds information about the various external sources used to populate the database. It

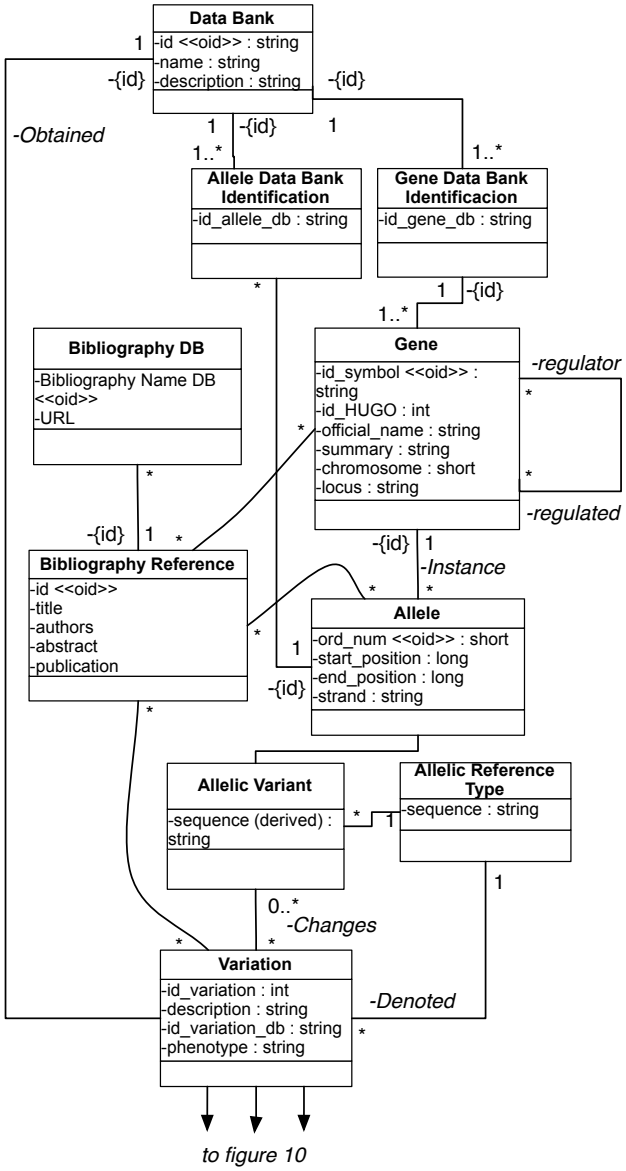


Figure 9: Conceptual model, part (a).

consists of Id, which is an internal identifier, Name and Description. The tuples in Data Bank are related to the other concepts through the use of two entities: Allele Data Bank Identification and Gene Data Bank Identification, whose main functions are to solve the many-to-many relations existing between those concepts. The first relates the appropriate Data Bank to a Gene, and the latter to an Allele.

Since CSHGDB is also meant to be storing information about various literature sources, a Bibliography DB concept has thus been created. In this class information is stored about the name of the literature source, Bibliography name DB, and where to find it; URL. The Bibliography Reference entity then functions as a link to the many-to-many relations existing between the Gene, Allele, Variation concepts and the corresponding literature. It holds information about the literature title (Title), authors (Authors), abstract (Abstract) and publication details (Publication). Along with this data, it contains an Id attribute for internal identifying purposes.

4.2 MUTATIONAL CONCEPTS

See for an overview of the mutational concepts [*figure 10*].

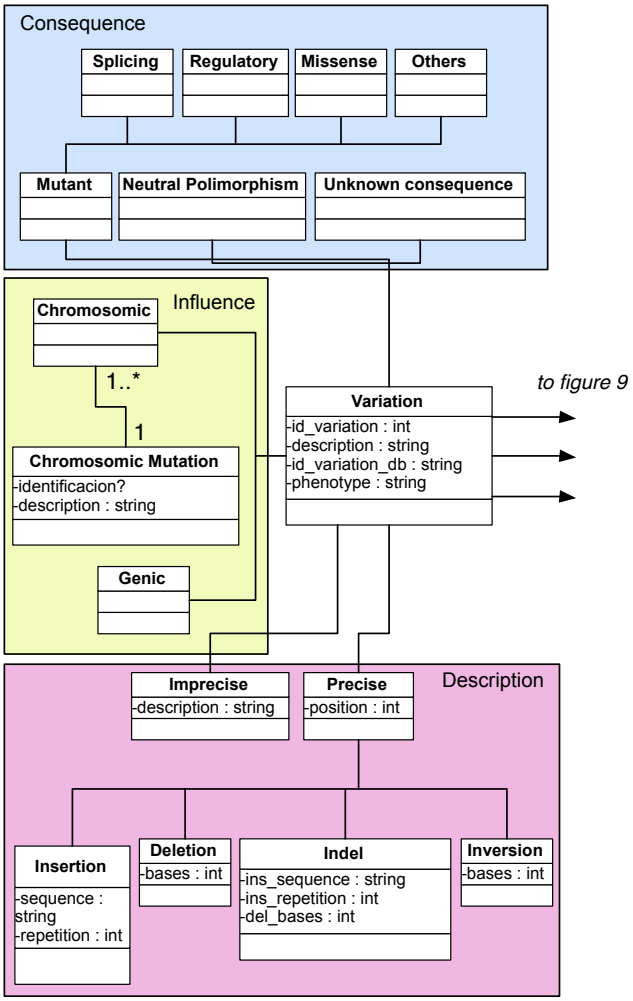


Figure 10: Conceptual model, part (b).

The Variation entity stores information about changes in a certain Allele in respect to it's reference, the Allelic Reference Type. It has an id_variation attribute for internal identifying purposes, and id_variation_db refers to the identification used in the external source. It further holds a description, meant to store a small description about the variation.

The other entities in part (b) can then be classified into three categories; Consequence, Influence and Description, each storing a distinct type of data on the variation at hand.

The Influence classes store information about whether the variation affects one or more genes. In the case of a genic influence, only one gene is affected, while in the case of chromosomal, multiple genes might be influenced.

The Consequence entities specify the variations' effect. This can either be Mutant, a Neutral Polymorphism or the effect might be unknown. The Splicing, Regulatory, Missense and Others concepts are considered to be mutations since they have a negative effect, hence they are a specialization of the Mutant concept.

Ultimately, the Description classes include descriptive information about the variation. Depending on the degree to which the data on the variation is precise, it falls into either the Precise or Imprecise class. When imprecise, the entity only stores a general description. In the case of precise data, it stores the position of the variation and further specifies the nature of the variation into four classes: Insertion, Deletion, Indel and Inversion. Each of these concepts stores information about this specific type of variation and the exact attributes vary from type to type.

4.3 THE BRCA1 MAPPING

This section covers the mapping of the data found in the various sources to the main concepts of CSHGDB. The main concepts are the Gene, Allele, Allelic Reference Type, Allelic Variant, Changes, Variation, Precise and Imprecise classes [*figure 11*]. The Changes, Allelic Variant and Allelic Reference Type deserve some extra explanation. Each gene contains one or more alleles,

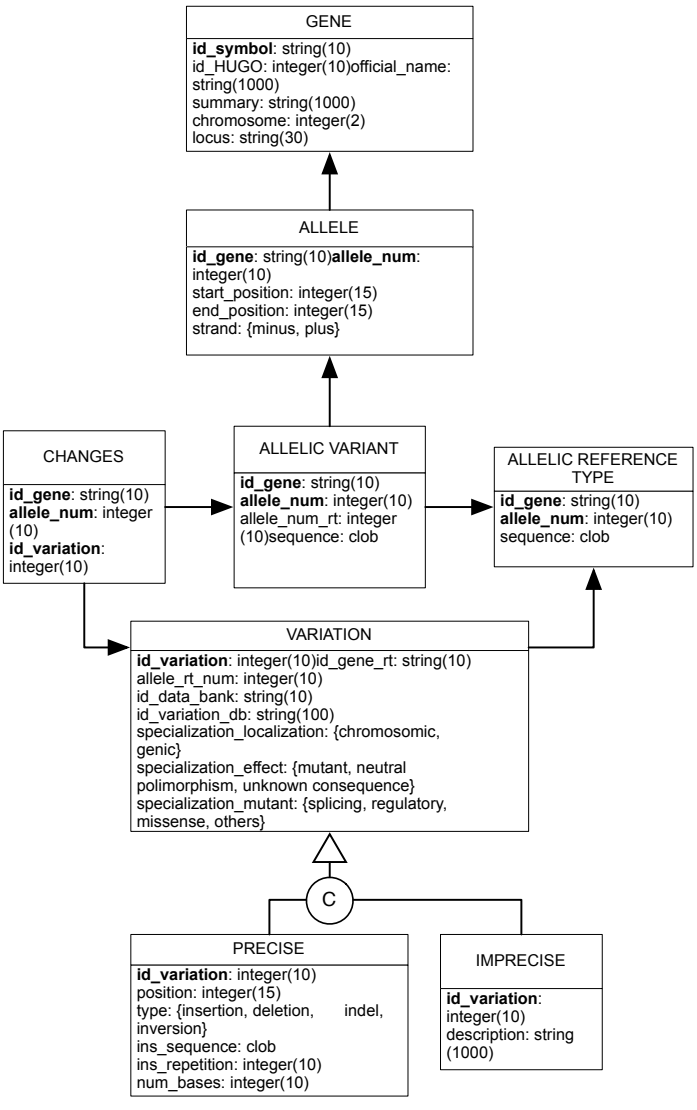


Figure 11: CSHGDB main concepts.

these alleles have genetic sequences, and one particular sequence is regarded to be the reference sequence, which is an instance of the Allelic Reference Type concept. In CSHGDB the NCBI refseq [*NCBI NG_005905.1*] fulfills this function. Encountered variations in genomes from different people are then considered to be instances of the Variation concept, and thus deviations of the reference sequence. From these deviations, or Changes, an Allelic Variant sequence is generated, thus forming an instance of the Allelic Variant concept. A Variation can either be Precise of Imprecise, depending on the degree to which it's characteristics are known, and thus a generalization relation exists between these concepts.

4.3.1 GENERAL

Most of the Gene entity attributes can be filled with data from the NCBI source: Official Symbol, Official full name, Primary source and Summary correspond respectively to the CSHGDB attributes: ID_symbol, Official_name, ID_HUGO and Summary. The other attributes are then extracted from HGMD, Location then corresponds to Chromosomal Location, and Chromosome to the first two digits of Chromosomal Location.

The Allele class is next to be populated. The ID_gene is a foreign key matching the ID_symbol of the corresponding gene. Allele_num functions as the internal identification number of the allele. Start_position and end_position refer to the start and end position in reference to the chromosome, which corresponds to the NCBI accession region, found at the NCBI Reference Sequence. The Strand attribute is extracted from the 'splice junctions overview' found at HGMD, a 5' start corresponding to 'plus' and a 3' to 'minus'.

In the Allelic_Reference_Type entity, the ID_gene and Allele_num attributes respectively match the corresponding gene's ID_symbol and allele's Allele_num and function as foreign keys. The Sequence corresponds to the 'NCBI reference sequence'.

	Missense/nonsense	Splicing	Regulatory	Other
Small Insertions	-	0	n.a.	99
Small Deletions	-	0	n.a.	290
Small Indels	320	79	n.a.	11
Total:	320	79	n.a.	400

Table 2: HGMD BRCA1 precise mutational data classified in a two dimensional way. Striped cells indicate some incompatibility between the two classifications.

4.3.2 VARIATIONS

Each variation found in the HGMD database is inserted as a distinct tuple in the Variation entity, which stores mainly common information on the variation. The ID_variation is locally assigned as are the ID_gene_RT and ID_allele_num_RT attributes, referring to the corresponding gene's ID_symbol and matching allele's Allele_num respectively. specialization_effect represents the effect of the variation, in case the effect is negative and therefore the variation accounts as a mutation, this attribute values as Mutant. However, in some cases variations' effects are neutral or simply unknown, in the first case the value Neutral Polymorphism is applied and in the latter Consequence Unknown.

Specialization_mutant is assigned the values 'splicing', 'regulatory', 'missense', or 'other' depending on how the variation affects expression, how to populate this attribute will be discussed more thoroughly shortly. The specialization_location depends on whether the mutation affects one or more genes, and is assigned Genic in case only one gene is involved and Chromosomal in case it affects multiple genes. The ID_data_bank is assigned the 'HGMD' value, since all the mutational data was acquired from this source. ID_variation_db is appointed the corresponding identifier HGMD uses to indicate the variation at hand internally and Description includes a summary description of the variation.

HGMD classifies it's mutational data in a two-dimensional way [*table* 2], first of all it distinguishes on the basis of it's physical nature, so a mutation can either be an insertion, a deletion or a combination of those, an indel. If the affected number of base pairs (bp's) is equal to or less then 20, the mutation classifies as a small variation. Since all small mutations only affect one gene, the specialization_location attribute is always set to Genic. If more than 20 bp's are influenced by the mutation, it classifies as a gross variation, in this case however, the specialization_location attribute might be set to Chromosomal, since they might affect more then one gene. Adding to this, each mutation can also by categorized by it's consequence on expression, hence the missense/nonsense, splicing, regulatory and other categories. The categories show some overlap, as both splicing and regulatory mutations can involve either a small insertion, a small deletion or a small indel. Nonetheless in the case of missense/nonsense, the physical nature of the mutation is restricted to small indels.

The Precise concept holds information about variations whose whereabouts are specifically known. ID_variation refers to that same attribute in the Variation entity, thus referring to the corresponding variation. Position holds the nucleotide location of the mutation in respect to the entire gene. Due to the fact each type of HGMD mutation undergoes a specific transformation to fit CSHGDB, each of them will be discussed separately.

Missense/Nonsense

Missense and Nonsense mutations are both single nucleotide changes, hence only affecting one base pair. Missense mutations cause substitution of a different amino acid, while nonsense mutations result in a premature stop codon, or a nonsense codon in the

transcribed mRNA, eventually leading to a truncated, incomplete and usually nonfunctional protein product.

Since Missense/nonsense mutations are always of the Small Indel type, a standard loading procedure can be given: the Type attribute is assigned the 'indel' value, 'ins_sequence' the inserted nucleotide, 'ins_repetition' is not applicable and therefore is assigned '1'. Since this type of mutations only involves single nucleotide substitutions, the 'num_bases' attribute is always assigned '1'. Also, the specialization_mutant attribute in the Variation entity is set to Missense. At the same time, since this type of mutations only affects one gene, the specialization_location attribute is set to the Genic value.

Splicing

Splicing mutations affect the splicing process, usually resulting in one or more introns remaining in mature mRNA. This may eventually lead to the production of anomalous proteins.

Since a splicing mutation can be either an insertion, deletion or an indel the attributes are variable and should be set to the appropriate values depending on the physical nature of the mutation at hand. Anyway, the specialization_mutant attribute in the Variation entity is set to Splicing. Here too, the specialization_location attribute is set to the Genic value.

Regulatory

Regulatory mutations affect the regulatory functions of genes, hereby affecting the frequency of expression.

Since a regulatory mutation can also be either an insertion, deletion or an indel the attributes are variable as well and should be set to the appropriate values depending on the physical nature of the mutation at hand. Here too, the specialization_mutant attribute in the Variation entity is set to Splicing and the specialization_location attribute to Genic.

Repeat variations mutations
There are no Repeat variations mutations in the BRCA1 gene, thus no problems were identified here.

Small deletions

Here the Type attribute is appointed the 'deletion' value. Since the 'ins_sequence' and 'ins_repetition' are not applicable, they are both assigned the '1' value. 'num_bases' corresponds to the number of deleted nucleotides. Further more, the specialization_mutant attribute in the Variation class is set to Other, since HGMD does not define this mutation as having a missense/nonsense, splicing or regulatory consequence on expression.

Small insertion

Obviously, the Type attribute should contain the 'insertion' value and 'ins_sequence' the inserted nucleotide sequence. 'ins_repetition' and 'num_bases' are not applicable, so the first is assigned '1' while the latter stays empty. Also, the specialization_mutant attribute in the Variation entity is set to Other.

Small indels

The Type attribute is assigned the 'indel' value, 'ins_sequence' refers to the inserted nucleotide sequence. The number of deleted nucleotides is then introduced in the 'num_bases' attribute. 'ins_repetition' is assigned '1'. And in this case as well, the specialization_mutant attribute in the Variation entity is set to Other.

The HGMD Gross insertions, Gross deletions, Complex rearrangements then are inserted in the Imprecise entity, since they are both either chromosomal or genic ánd have a specified effect.

5. FINDINGS

5.1 MUTATIONAL DATA LOADING PROBLEMS

Since the main effort of this project has been directed at introducing the mutational data HGMD provides, the following analysis will mainly be focussed on this process. HGMD provides 10 mutations types, which overlap to a considerable extend as has been discussed earlier. The types are: *Missense/nonsense, Splicing, Regulatory, Small Deletions, Small Insertions, Small Indels, Gross Deletions, Gross Insertions, Complex Rearrangements* and *Repeat Variations*. In theory, the *Missense/nonsense, Splicing and Regulatory* mutation-types indicate the mutational effect. Whereas the *Small Insertions, Small Indels, Gross Deletions, Gross Insertions, Complex Rearrangements* and *Repeat Variations* mutation-types typically indicate the nature of the operation of the mutation itself. However, in CSHGDB the highly variable mutations are considered to be imprecise, this includes the following types: *Gross Deletions, Gross Insertions, Complex Rearrangements and Repeat Variations*. The imprecise mutations then, are to be loaded in a very generic way during which no problems surfaced, so they will not be discussed in detail.

First the problem categories will be presented in the 'problem categories' section, then concrete examples of each category, in the form of encountered problems in the BRCA1 gene, will be discussed in the 'problem instances' section. Ultimately, a deeper analysis of the problems will be performed in the 'problem analysis' and 'causes' sections. Solutions will be discussed in their respective 'solutions' section, after which conclusions will be drawn in the 'conclusions' section.

Also, this section needs some terminology explanation. Up until this point HGMD has been considered to be a source from which data is extracted and then loaded into CSHGDB. However HGMD retrieves it's information elsewhere, mainly scientific papers and is thus not a source on it's own. Therefore, HGMD will be referred to as being a medium, functioning as a data transmitter between the scientific papers that present the results of biological research, and the rest of the world. The sources are thus considered to be the papers from which HGMD loads it's data.

5.2 MUTATIONAL DATA PROBLEM CATEGORIES

The encountered problems during the BRCA1 loading procedure lead to the identification of 6 generic categories. Each category is then subdivided into more concrete subcategories, or 'instances'. This approach allows for flexibility as it permits easy adding and modifying of concrete problems in the existing categories without the necessity of changing the architecture, thereby coping with unforeseen conditions. The categories are: *(p1) data extraction*, *(p2) data transformation*, *(p3) data ambiguity*, *(p4) incomplete data*, *(p5) inadequate data* and *(p6) inconsistent data*.

5.2.1 p1: data extraction

This category represents problems directly influencing the process of data extraction. 3 specific instances of this category were encountered during the BRCA1 loading procedure:

(a) *medium using HTML tables to convey information*
(b) *medium using natural language* and
(c) *medium using HTML mouse-over tags to convey information*.

5.2.2 p2: data transformation

The encountered problems affiliated with notational differences between the medium and CSHGDB, requiring a transformation, are to be categorized here. For BRCA1 they include the following:

(a) *medium using codon referenced notation*,
(b) *medium using cDNA referenced notation*,
(c) *medium using splice-junction referenced notation*.

5.2.3 p3: data ambiguity

In case the data provided is ambiguous for some reason, it should be reported in this category. In the case of BRCA1, only one concrete instance was identified:

(a) *medium indicating mutation phenotype ambiguously*.

5.2.4 p4: incomplete data

This category represents the occurrence of the medium lacking data entry, this can include lack of information about tuples in the medium's database but also entire tuples. During the BRCA1 loading procedure only one concrete instance was identified:

(a) *medium lacking data entry entry*

5.2.5 p5: inadequate data

Here the problems affiliated with medium data incorrectness are categorized. Due to the difficult nature of detecting this type of problems only one instance has been detected in the BRCA1 loading procedure:

(a) *medium providing erroneous data*.

Data incorrectness is interpreted very wide and includes errors due to data entry errors, but also methodological deficiencies in research papers used to populate HGMD.

5.2.6 p6: inconsistent data

In category p6 then, problems with data inconsistencies are to be included. Three concrete instances have been identified:

(a) *medium referring mutation to non existing intron*
(b) *medium referring to nucleotides, not appearing at that location*
(c) *medium using inconsistent way of locating mutations*.

Generally speaking, in case the medium mentions things to be X in one case, referring to that same thing as being Y in another, data is said to be inconsistent and should be categorized in this category.

5.3 MUTATIONAL DATA PROBLEM INSTANCES

A total of 2629 instances of the earlier mentioned problem categories was encountered during the investigation, this section provides an overview of the encountered problems, along with examples to clarify the terminology and also describing the problem in more detail. Not every encountered problem is mentioned in this oversight, for a complete list of encountered problems ordered by HGMD mutations type, consult [*table 7*] and [*appendix 2: BRCA1 problem occurrence*].

5.3.1 p1: data extraction

This type of problems describes the difficulties associated with extracting the data from the various external sources. The following instances of the this category are HGMD specific and might be different in other sources.

(A) (*medium using HTML tables to convey information*) In the case of HGMD, all mutational data is provided in HTML-tables, making a fully-automated approach difficult. 801 occurrences of this problem have been identified in the process, which means basically all mutational information is provided through HTML-tables [*figure 12*].

(B) (*medium using natural language*) Some data is provided in natural language. For instance the fact that the first two BRCA1 exons are alternative non-coding exons is only mentioned in the 'splice junctions description': '*The first 2 exons are alternative non-coding exons*' and '*The Translation Initiation Codon is located within exon 2*', this mainly affects locating Splicing mutations (1 instance). Adding to this, in Small Deletions (2 instances) and in Small Insertions (3 instances) some mutations are located through mouse-over tags (this problem is further described in p1c), the information communicated by these tags is highly unstructured to a degree that we might call it natural language as well. Also, in the case of imprecise mutations (Gross Deletions, Gross Insertions, Complex Rearrangements and Repeat Variations), the greater part of the information presented by HGMD is in natural language. As a result, in these cases the Genoma team has decided to simply insert them as plain text and thus no further attention has been directed at solving them in this project.

(C) (*medium using HTML mouse-over tags to convey information*) In some cases HGMD provides data through so-called mouse-over tags [*figure 13*]. Which means this data is only accessible by hovering the mouse cursor over specific parts of the table. This is a problem, since a semi-automated approach by copy-pasting the HTML-tables in the programming logic as will be discussed shortly, does not capture this kind of information. Also the provided data in these tags seem to be highly unstructured. Two instances of this problem have been identified, both of them in Small Deletion mutations.

5.3.2 p2: data transformation

Each external source has it's own format on how to present data. Since CSHGDB will be using it's own data model the data from the external sources will need to undergo various transformations to fit the database. The problems associated with these operations are to be discussed in this section.

This problem involves a CSHGDB design decision; it is desired for the various known alleles, locations of mutations are introduced on a DNA-referenced single nucleotide scale. [*figure 14*] explains the difference between DNA nucleotide referenced and cDNA codon referenced notations and it's implications on locating mutations. The first nucleotide of codon A in the cDNA can easily be located in the DNA by simply multiplying the codon number by 3. However, cDNA codons B and C are not located that easily due to introns. Therefore the first nucleotide of codon B is actually the 15th nucleotide in the DNA sequence, instead of the 12th. The first nucleotide of codon C then is located at the 27th nucleotide in the DNA sequence, instead of the 18th. Codon B also visualizes the possibility of cDNA

Accession Number	Codon change	Amino acid change	Codon number	Phenotype	Reference
CM021503	aATG-GTG	Met-Val	1	Breast and/or ovarian cancer	Meindl (2002) Int J Cancer **97,** 472
CM041678	ATG-ACG	Met-Thr	1	Breast and/or ovarian cancer ?	Abkevich (2004) J Med Genet **41,** 492
CM014520	ATG-AGG	Met-Arg	1	Ovarian cancer	Sekine (2001) Clin Cancer Res **7,** 3144
CM960163	ATGg-ATT	Met-Ile	1	Breast cancer	Couch (1996) Hum Mutat **8,** 8
CM940170	GTA-GCA	Val-Ala	11	Breast cancer	Castilla (1994) Nat Genet **8,** 387
CM031646	aCAA-TAA	Gln-Term	12	Breast cancer	Adem (2003) Cancer **97,** 1

Figure 12: Screenshot from HGMD, revealing the HTML-tables from which data is presented.

Accession Number	Deletion	Codon(^)	Phenotype	Reference
CD991644	CAATTGCTTGactgttctttACCATACTGT	Non-coding	Breast cancer	Li (1999) Hum Genet **104,** 201
CD994433	ACATGCGTGTgtGTGGTGTCCT	N… I7E8-24, aka IVS7 -15 del10. ?		Lallas (1999) Mol Genet Metab **67,** 357
CD022526	GCAGAAA^ATCttAGAGTGTCCC	21	Breast and/or ovarian cancer	Fries (2002) Mil Med **169,** 99

Figure 13: Screenshot from HGMD. The image shows the use of mouse-over tags to communicate certain information.

codons being cut in half by splice junctions, further complicating this view.

However, HGMD uses two different ways of locating mutations: the first, which is used most, locates mutations by indicating codons plus offset in a cDNA sequence. This way of locating mutations is used for all mutation-types except for Splicing mutations, in which case another way of locating is used: here the splice junctions (borders that separate the introns and exons in the DNA sequence) are used as a reference along with an offset [*figure 14*].

(A) (*medium using codon referenced notation*) As said, HGMD refers to codons in many cases, which are sets of three nucleotides. Since CSHGDB will be using a nucleotide referenced position to indicate mutations' whereabouts, a transformation of HGMD provided data is necessary. In theory, acquiring the correct nucleotide would be a matter of multiplying the codon number by three, however as will be discussed in the next problem (*p1b*) reality is slightly more complicated. HGMD uses this way of locating mutations in the case of Missense/nonsense (320 instances), most of the Small Deletion mutations (288 instances), most of the Small Insertion mutations (98 instances) and Small Indels (11 instances), leading to a total of 715 instances of this problem.

(B) (*medium using cDNA referenced notation*) Retrieving the corresponding nucleotide in DNA would simply be a question of multiplying the codon number by 3, if not for the existence of introns and exons. cDNA only comprises of the genes' exons, thereby excluding the introns, contained in the DNA. Due to this fact and given that CSHGDB will be incorporating a DNA referenced scale, a linear transformation, by multiplying the codon number by 3, simply is not possible [*figure 15*]. The fact that certain cDNA codons might be 'cut in two' by splice junctions further complicates this view [*figure 15*].

(C) (*medium using splice-junction referenced notation*) Also, in splicing mutations, HGMD uses a different way of locating mutations. Here, the mutations are located by referring them to the so-called splice-junctions. These splice junctions indicate the borders between exon en introns. HGMD thus indicates an intron border, by giving an intron number and specifying which border by providing either a '*donor*-' (ds) or '*acceptor*-' (as) side of the intron.

```
 5UTR E001a agcgcgggaattacagataaatt/AAAACTGCGACTGCGCGGCGTGA
E001a I001a TCACCCTCTGCTCTGGGTAAAGgtagtagagtcccgggaaaggg
I001a E001b cgttgtgaaccctggggagGGGGGCAGTTTGTAGGTCGCGAGGGA
E001b I001b GAGACTGTCT(CAAAA)6AACACCGGCTGgtatgtatgagaggatgggacct
I001b E0002 aa(at)6gtttttctaatgtgttaaagTTCATTGGAACAGAAAGAA[ATG]GATTTATCTGCTCTT
E0002 I0002 GAAAATCTTAGAGTGTCCCATC[TGgtaagtcagcacaagagtgtattaatttg
I0002 E0003 ttttcttttct(c)7taccctgctagT]CTGGAGTTGATCAAGGAACCTG
E0003 I0003 AAGTGTGACCACATATTTTGC[AAgtaagtttgaatgtgttatgtggctccat
I0003 E0005 attgtttctttctttctttataatttatagA]TTTTGCATGCTGAAACTTCTCA
E0005 I0005 TGTAAGAATGATATAACCAAA[AGgtatataatttggtaatgatgctaggttg
I0005 E0006 agtgtgttctcaaacaatttaatttcagG]AGCCTACAAGAAAGTACGAGATT
E0006 I0006 TCAGCTTGACACAGGTTTGGAG[Tgtaagtgttgaatatcccaagaatgcaac
I0006 E0007 acataatgttttcccttgtattttacagAT]GCAAACAGCTATAATTTTGCA
E0007 I0007 GTGAACCCGAAAATCCTTCCTTGgtaaaaccatttgtttt(ctt)7ttc(t)10c(t)12g
I0007 E0008 cttgactgttctttaccatactgtttagCAGGAAACCAGTCTCAGTGTCCA
E0008 I0008 GACGTCTGTCTACATTGAATTG[Ggtaagggtctcagg(t)6aagtatttaat
```

Figure 14: Screenshot taken from HGMD, showing the 'Splice junctions overview'.

Figure 15: The difference between DNA nucleotide referenced and cDNA codon referenced notations.

The donor side corresponds to the side closest to the 5' end of the DNA strand, while the acceptor side corresponds to the side closest to the 3' end of the DNA strand. Then an offset is given (+x, where 'x' stands for the amount of nucleotides the mutation is set off), to indicate the amount of nucleotides between the indicated splice junction and the actual mutation. In a so-called splicing mutations overview HGMD then provides a sample sequence for each intron/exon-junction contained in the gene. This method of locating mutations is used primarily in splicing mutations (80 instances), but in some exceptional cases HGMD also uses this notation to provide locational data for other types of mutations. For instance, In Small Deletions (2 instances) and in Small Insertions (3 instances).

5.3.3 p3: data ambiguity

(A) (*medium indicating mutation phenotype ambiguously*) In the HGMD data exists ambiguity; for instance, mutations may or may not result in a certain phenotype, this is indicated by a question mark following the supposed phenotype. However, no probability chances are stated and a mutation without a (noticeable) phenotype is considered to be a variation with neutral effect. Since variations and mutations are considered to be two different concepts in the CSHGDB data-model, this poses problems with loading the database correctly. 94 instances of this problem have been identified: missense/nonsense mutations account for the most instances (73), splicing mutations contains another 16, small deletion mutations 2 and small insertion mutations account for 3 instances.

5.3.4 p4: incomplete data

Since all the data HGMD contains has been extracted from scientific papers, and given the highly unstructured nature of scientific papers, it will be very difficult to obtain a 100% coverage of finding all problems that fall in this category. Examples are given though, to prove the problems are existing and should be accounted for in future

processing of the data.

(A) (*medium lacking data*) In some cases the HGMD mutational data simply lacks entries. For instance, in the splice mutations overview provided by HGMD mentions 5 mutations in intron 22, while [*Panguluri, 1999*] states at least 2 other mutations; IVS22+67(T>C) and IVS22+8(T>A). Three concrete examples of this problem were encountered, all three in Splicing mutations. However, this particular type of problem is very difficult to detect, since finding them involves rereading the articles HGMD provides which is hard to automate. Thus, although only three concrete occurrences of this problem have been encountered, it is very likely more exist.

5.3.5 **p5: inadequate data**

Since all the data HGMD contains has been extracted from scientific papers, and given the highly unstructured nature of scientific papers, it will be very difficult to acquire a 100% coverage of finding all problems that fit this category. However, due to the rather high amount of problems identified in the HGMD database, it is very probable it also contains plain incorrect data.

(A) (*medium providing erroneous data*) Splicing mutation CS961492 describes a C>T mutation, as a possible phenotype HGMD indicates Breast cancer. However, having the read the corresponding article [*Langston, 1996*], not once breast cancer is mentioned in combination with this mutation. The article does mention the mutation as being affiliated with men suffering from prostate cancer. Thus, deducing from the rather limited information made available by HGMD on this specific mutation, it is concluded HGMD made an error during data entry.

5.3.6 p6: inconsistent data

(A) (*medium referring mutation to non existing intron*) Splicing mutations CS063247 and CS011027 should be located near intron 4, according to the HGMD Splicing mutations overview. However according to the splice junctions overview HGMD provides, there exists no intron 4, nor an exon 4. Since indeed both of the papers [*van der Hout, 2006*] and [*Shattuck, 2009*] state the mentioned mutations near intron 4, it would be logical to presume the problem is on HGMD's side. So at first, this specific problem instance was considered to be either a major flaw in HGMD due to inconsistent reference sequences, or a result of human error during the HGMD loading procedure. However, deeper research revealed a more subtle situation. As mentioned in p2c, splice mutations are located in HGMD by using a splice junctions overview, which provides an overview of intron- and exonborders in the gene. HGMD constructs this overview by using a NCBI reference sequence, in this case '*L78833*'. This reference provides a comment in natural language that explains the absence of an exon 4 as: "Characterization of an aberrant BRCA1 cDNA clone in the original report [*Miki, 1994*] led to the misidentification of an inserted Alu element as exon 4. Not normally found in BRCA1 transcripts, insertion of this Alu would lead to introduction of a STOP codon. Hence, BRCA1 exons and introns are numbered 1a, 1b, 2, 3, 5, 6, etc.". The fact remains however, that the two earlier mentioned splicing mutations CS063247 and CS011027 do refer to an intron 4, indicating an inconsistency between HGMD and the mentioned papers by [*van der Hout, 2006*] and [*Shattuck, 2009*].

(B) (*medium referring to nucleotides, not appearing at that location*) In Splicing mutations, mutation CS012667 indicates a G>A mutation in nucleotide +3 from the start of intron 2 . However neither the HGMD splice junction overview, nor the NCBI reference gene sequence indicates a G-nucleotide at this location. A very similar event happens with Splicing mutations CS001825 and CS991331. They both involve a mutation located 7 nucleotides 'upstream' (+7) from the start of intron 22. However, the first mentions an A>G mutation, while the latter describes, for that exact same location, a T>C mutation. Since both the NCBI reference sequence and the HGMD 'splice junctions overview' indicate an 'A' nucleotide at the appropriate position, and the CS001825 reference article [*Khoo, 2000*] indeed mentions an A>G mutation at the specific location, one could easily conclude the CS001825 A>G mutation in 'the correct one'. However, the CS991331 reference article [*Panguluri, 1999*] does indeed point out a T>G mutation at the location, so the truth might be slightly more subtle. Since [*Panguluri, 1999*] involves an African-American population, while [*Khoo, 2000*] entails a Chinese population, a possible reason for the irregularity might be general genetic differences between those ethnic groups. A Phenomenon referred to as Single Nucleotide Polymorphisms, or SNPs [*Zhao, 2003*]. 5 Occurrences of this problem have been identified in Splicing mutations, 1 in Small Deletions and Small Insertions each, leading to a total of 7 occurrences.

(C) (*medium using inconsistent way of locating mutations*) In some cases, the HGMD database uses different ways of locating mutations, within the same type of mutations. For instance, Small Insertion mutations CI030168, CI962219 and CI022582 happen in non-coding areas of the gene, just like the Small Deletions mutations CD991644 and CD994433. Since HGMD generally uses a cDNA codon referenced way of locating these types of mutations, and given that non-coding sequences simply not exist in the cDNA, HGMD locates these earlier mentioned mutations in a different way. In the case of Small Insertions, HGMD provides a Splice Junction reference, very much like the method used to locate Splicing mutations. In this case the CI030168, CI962219 and CI022582 mutations are located at IVS20+21, IVS20+48 and IVS20+64 respectively. So 'IVS20' indicates the intron number, where '+21' indicates the offset, however since no acceptor/ donor information is provided, it is unclear from which side of the intron the offset should be referenced. In the case of Small Deletion mutations CD991644 and CD994433 at first sight, no indication of how to locate them is provided. However, as is described in p1c, this information is provided through mouse-over tags in the Splice Junctions referenced form, described earlier. CD991644 is thus located by '*I7E8-24, aka IVS7 -15 del10.*' and CD994433 is located by '*I12+34 / polymorphism ?*'. This problem was thus encountered 3 times in Small Insertions and 2 times in Small Deletions, making a total of 5 occurrences.

5.4 CATEGORY ANALYSIS

This section covers the analysis of encountered problems. The previous section described the various instances that were found during the BRCA1 gene loading process, which eventually resulted in the earlier mentioned categorization. Now the separate instances will be partly uncoupled from their concrete instances and be viewed on a more general scale, discussing their background and in some cases superficially their resolutions. The problem resolutions will be discussed in more detail shortly in the 'Problem solutions' section. Also, each problem is assigned a severity rating, that has to be considered on two dimensions; first the so-called acute severity and secondly the general severity. The acute severity handles basically the solvability of the concrete problem at hand, so whether it is possible to solve the particular occurrence of that problem within this gene. The general severity then indicates the degree to which the problem is considered to be solvable on a larger scale, considering there are many more genes to be inserted. In case a problem receives a low score, it is considered to be easily resolved and often simply refers to a difference in notational styles between the medium or source and CSHGDB. Medium indicates there exists a certain difficulty to solving the problem, however the problem can usually be solved but needs attention and a more time-consuming resolution. This typically is the case when an automated approach appears ineffective and a manual approach needs to be used. Problems that are considered to be solvable only by putting in unreasonable amounts of effort are classified a high score.

For example, the problem that HGMD provides some inconsistent information about mutations happening around intron 4 (p6a), was resolved for the BRCA1 gene by artificially completing the splice junctions overview, which lacked intron 4, by using NCBI provided information and is thus considered to have a low/ medium acute score. However, the problem actually indicates an underlying cause; that some papers used by HGMD to extract mutational information might use different reference sequences or might even be referring to non-existing introns/exons as will be discussed later. Since it is very difficult to check in an automated way which reference the papers use this problem is assigned a high score on general severity.

5.4.1 p1: data extraction

(A) (*acute=low, general=medium*) HGMD presents it's mutational information in HTML tables, which complicates a fully automated loading procedure. However, by copy-pasting the table in the programming logic, declaring it as a variable and 'cutting' the data in bite-size chunks using the PHP 'explode()' function, the data can be extracted semi-automatically. The problem itself for the BRCA1 gene was relatively easily resolved, hence the acute severity was rated low. However, due to the shear amount of genes to be inserted (around 2500 provided by HGMD, 25.000 however in total presumably contained in the human genome) the problem presents more difficulties when viewed from a larger perspective, hence the general severity was classified medium.

(B) (*acute=low, general=medium*) Some data is presented by HGMD in natural language, which is highly unstructured and therefore complicates an automated loading procedure to a high degree. However, for this particular gene, the data provided in natural language did not prove to be essential and was thus avoidable, hence a low acute score was assigned. However, the fact HGMD presents some information in this format is worrying, indeed this time the information provided was non-essential, or a work-around could be devised, but this might not always be the case. Considering natural language is inherently a weak point of computers in general and to this day no perfect method of handling this type of information has been developed, the general severity of the occurrence of natural language in the HGMD database is considered to be medium. Also, the main way of verifying HGMD data integrity would be to read all the scientific papers HGMD uses to extract it's information, which by nature are also highly unstructured and therefore extremely difficult to automate.

(C) (*acute=low, general=low*) Due to the chosen approach of solving the HTML-table problem (p1a), by copy-pasting the table in the programming logic, the information provided through HTML mouse-over tags at this point is not captured. Indeed, during this process the actual HTML data is not transmitted to our own programming logic, only the visual representation provided by the browser, and therefore some part is lost. However, there exist various work-arounds to solving this particular problem, as well acute as in general, therefore a low score was assigned to both.

5.4.2 p2: data transformation

HGMD uses two main ways of locating mutations, the first is used most frequently and includes indicating codons within a cDNA sequence of the gene, the second locates the mutations in reference to splice junctions. In this category exist three problems, the first deals with the fact CSHGDB uses a DNA reference, as opposed to HGMD's cDNA (*p2b, acute=low, general=low*), and the second deals with the fact CSHGDB will reference to nucleotides, as opposed to HGMD's codon notation (*p2a, acute=low, general=low*). The third problem then handles the splice-junctions referenced locating of mutations, used mainly in the Splicing mutations overview provided by HGMD (*p3b, acute=low, general=low*) and in some exceptional cases in other mutation types. Since all three described problems are a matter of differences between CSHGDB and HGMD representation of information, these problems intrinsically are easily resolved and thus receive a low score on both the acute and general scale.

5.4.3 p3: data ambiguity

(A) (*acute=low, general=low/medium*) One example of data ambiguity, with many occurrences (94), was identified while loading the BRCA1 gene into CSHGDB. For 94 mutations there exists no absolute certainty whether it leads to developing a disease or not, indicated by a question mark following the phenotype. Since a mutation is only a mutation when it affects health negatively, otherwise being a variation with neutral effect, this poses problems on loading CSHGDB properly. However, since this

ambiguity is indicated, through the question mark, it cán be dealt with, both on an acute and a general scale, thus this problem classifies as low on both. However, depending on the degree to which certainty about phenotype will be required in the future this problem might escalate on a general scale to medium. A little bit more explanation on this specific instance is given by HGMD in the background pages [*HGMD, Polymorphisms*], basically confirming ~55% of the polymorphic variants recorded in HGMD are 'disease associated', meaning ~45% is nót. However, HGMD urges to express they necessity of recording the not disease associated variants anyway as their effect on phenotype might simply not be clear yet.

5.4.4 p4: incomplete data

(A) (*acute=medium, general=medium*) In some cases HGMD simply lacks data entries. Since correcting this anomaly would include rereading the large amount of papers HGMD used to extract the data, a simple solution to this problem does not exist. However, the professional version of HGMD might be more complete and at this time incompleteness is considered to be inherently a property of the youth of the field this project is situated in. Due to the fact this problem might resolve itself (partly) in the future, and the possibility that the commercial version of HGMD might be more complete, it has been classified medium on both the acute and general scale.

5.4.5 p5: inadequate data

(A) (*acute=high, general=high*) One concrete example of this problem was identified, where HGMD indicates a certain phenotype, not mentioned by the paper HGMD used to extract the information. Even more, the paper related to the specific mutation actually indicates a relationship to a completely different phenotype, thus revealing a possible error in the HGMD data-set.

However, indications exist there might be more occurrences of this particular problem instance. Due to the difficulties associated with detecting them, since that would involve rereading all of the papers HGMD used to populate it's database, many more occurrences might 'hide' in the data-set. Especially problem category 6: data inconsistencies suggest the existence of inadequate data. For instance problem p6b describes the occurrence of HGMD referencing nucleotide change 'X' > 'Y', while in the DNA reference sequence (both provided by HGMD and NCBI) at X's location happens to be another nucleotide, 7 occurrences of this problem have been identified, 5 in splicing mutations and 1 in both Small Deletions and Small Insertions. This indicates that either the papers providing this specific mutation used another reference sequence, which in itself is rather troubling, ór there has been a data typing error during the HGMD loading procedure.

Only one concrete problem of this type has been identified, however there is reason to believe more exists in the HGMD data set. Since the only way to check whether certain data is correct or not would involve reading all the papers used by HGMD the solution to this type of problems is going to be very difficult, therefore this problem has been labeled high both acute and in general.

5.4.6 p6: inconsistent data

(A) (*acute=medium, general=high*) In Splicing mutations HGMD refers to intron 4 to locate 2 mutations, however according to the splice junctions overview this intron does not exist, the reason for this irregularity is given in the '*Problem instances*' section in p6a. A workaround to this problem exists, and will be discussed shortly in the 'Solutions' section, however the reliability of this solution is questionable and therefore this problem is assigned medium on an acute scale. The real problem though is the underlying cause, indicating the eventuality that HGMD constructed it's Splice Junctions Overview from a different reference sequence, then the source papers use to reference their mutations. For this reason the problem has been classified as high on a general scale.

(B) (*acute=low/medium, general=high*) Also, in some cases HGMD indicates nucleotides at positions in the reference DNA, at which point actually another nucleotide sits. These inconsistencies can be detected, however reliance of this detection is variable. This variable reliance is explained by the fact HGMD provides relatively large sample sequences for all mutational information except Splicing mutations. These sequences, that surround the mutation as it were, provide a handle from which can be verified whether the calculated location indeed is the correct location by matching the given sequence with the subsequence within the DNA at the calculated location. So a relatively reliable indication can be given whether the nucleotide at the calculated position is actually different from what the mutational information indicates. However in the case of Splicing mutations, the sample size consists of only one nucleotide, effectively increasing the likelihood of finding that nucleotide at any random position in the DNA sequence to 25%. Since in most cases, this kind of problem can be detected, and therefore be dealt with it classifies as low on an acute scale. However due to the relative high unreliability factor in splicing mutations, the problem also classifies acutely as medium. On a general scale, things appear different though. The very existence of this type of inconsistencies is an indication of structural data deficiency in the HGMD database, either indicating that the various papers used by HGMD use different reference sequences or data entry errors. In case the various papers are using different reference sequences, which has been suggested earlier in p6a and HGMD does not facilitate this, every reason exists to doubt HGMD's splicing mutations overview reliability and therefore this problem classifies as high on a general scale.

(C) (*acute=low, general=medium*) In Small Deletion mutations, HGMD provides an irregular way of locating 2 mutations thereby disrupting the structural integrity of how it's information is presented. Coping with these irregularities does not pose many problems as will be discussed shortly in the Solutions section. However, the fact that HGMD cán use various ways of locating a mutation, within one set of

mutations, indicates a possibility that other ways of locating, not yet found nor investigated might be existing. The existence of this type of structural deficiency in how HGMD presents it's information poses mainly problems when an automated approach is considered. A more generic way of locating this mutations is thus suggested, and will be discussed shortly in the 'Solutions' section. Since this more generic solution exists and is probable of solving this problem, the acute severity of the problem is low. The generic solution is probably effective at solving future, devious instances of this problem as well but since this can not be stated with certainty, and more research is required the general severity of the problem classifies as medium.

5.5 PROCESS ANALYSIS

[*figure 16*] presents a graphical overview of BRCA1 mutation loading problem frequency from two perspectives. The *mutational %* indicates the relative amount of mutations affected by a specific problem instance and the *occurrence %* provides information on the relative amount a specific problem instance contributes to the total amount of encountered instances. Problem instances p1a, p2a and p2b immediately stand out due to both their high mutational % and occurrence % values, together accounting for 88.55% (accumulated occurrence % values) of all encountered problems. Problem instance p1a represents the fact that all HGMD mutations are presented in HTML tables, explaining the 100% mutational % score. Problem instances p2a and p2b describe the codon and cDNA referenced notation style respectively that HGMD uses to locate mutations in most cases, thus explaining their high mutational % scores.

[*table 3*] shows that the greater part of the found instances of problems are easily resolved: p1c, p2a, p2b, p2c and p3a all have a low severity rating assigned to them, meaning they can be resolved easily and thus the mutations affected by them should be automatically loaded in CSHGDB.

[*table 4*] indicates the accumulated occurrence percentages, indicating the distribution of problem instances by their severity. The "*Best*" column represents the situation if problem p1a is to be resolved, the "*Best, p1a unresolved*" column then represents the situation is p1a is not resolved. The "*Worst*" column indicates the situation if p3a turns out to have a medium severity rating, instead of low ánd p1a is not resolved.
As is shown in [*table 4*] and [*figure 17*], 61.54% of the encountered problems are assigned a low severity score. In the worst case when p3a turns out to have a medium severity this number could be 57.96%, however

	Acute severity	*General severity*	*Mutational %*	*Occurrence %*
p1a	low	medium	*100.00*	*34.01*
p1b	low	medium	*10.74*	*3.65*
p1c	low	low	*0.56*	*0.19*
p2a	low	low	*80.20*	*27.27*
p2b	low	low	*80.20*	*27.27*
p2c	low	low	*9.51*	*3.23*
p3a	low	low/medium	*10.51*	*3.58*
p4a	medium	medium	*0.34*	*0.11*
p5a	high	high	*0.11*	*0.04*
p6a	medium	high	*0.22*	*0.08*
p6b	low/ medium	high	*0.78*	*0.27*
p6c	low	medium	*0.89*	*0.30*
				100.00

Table 3: Severity ratings of the encountered problems during the BRCA1 variational data entry.

	Best	*Best, p1a unresolved*	*Worst, p1a unresolved*
general low	95.55%	61.54%	57.96%
general medium	4.06%	38.07%	41.65%
general high	0.39%	0.39%	0.39%

Table 4: Accumulated occurrence percentages, indicating the distribution of problem instances by their severity.

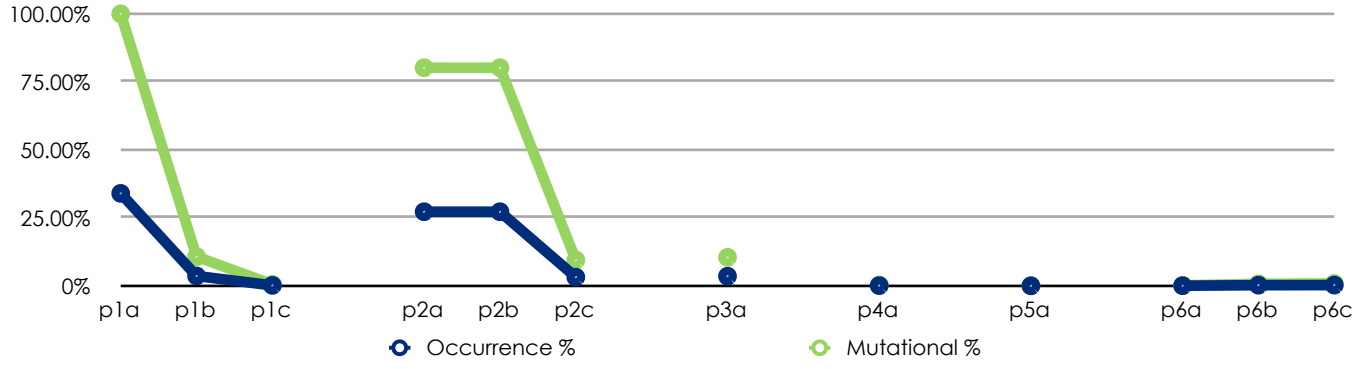


Figure 16: BRCA1 mutation loading problem frequency.

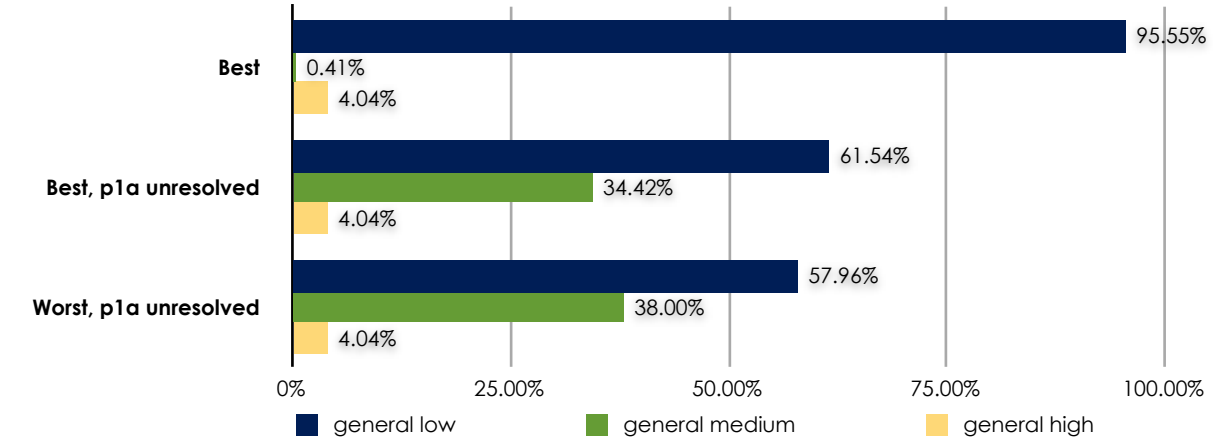


Figure 17: The distribution of problem instance severity.

due to the nature of problem p3a it is reasonable to expect it's severity to be low on a general scale. 38.07% of the problems then is relatively easily resolved by putting some effort in it, however it must be noted by far the greater deal of this percentage comes from problem p1a (*34.01%*). Problem p1a handles the data extraction difficulties induced by the fact HGMD has chosen HTML tables as a way to present it's data. The only reason this problem was classified a medium severity is due to the shear amount of genes to be inserted (around 2.500 contained in the HGMD database, ~25.000 exist in the human genome), leading to considerable effort in case all those HTML-tables have to be copy-pasted manually. However, in case the data extraction can be automated at this point, a large gain can thus be acquired. P1a is not a new problem in the sense that Bioinformaticians have been struggling with this specific type of problem for a long time. [*Stein, 2002*] handles the problem in more detail and also presents a vision on solving it. However, in the case p1a can ben solved, 95.55% of the encountered problems in the BRCA1 gene should be regarded easily solvable and the mutations associated to them automatically loaded into CSHGDB. Only 4.06% of the problems would require additional effort in solving them. In any case, 4.04 % of the encountered problems seem unsolvable at this point by reasonable means. These should be detected by the loading software, so they can be checked and solved manually.

The mutation % then is slightly more interesting, it uncovers the relative amount of problems affected by a type of problem, or problem instance. Thereby establishing a relation between problem severity and relative frequency in the BRCA1 gene data-set. However, since these individual frequencies can be overlapping, the individual percentages can not be summed. Indeed, the problems mentioned in *p6b: source referring to nucleotides, not appearing at that location* for instance can also be mentioned in *p5a: source providing erroneous data*, thus the percentages should be viewed individually. Looking at the mutational % values in [*table 4*] it is apparent problem *p1a: source using HTML tables to convey information* is going to be problematic. This problem appears in 100% of the mutations ánd requires manual effort in dealing with it, hence the medium severity rating. However, if each instance has to be dealt with manually, p1a is going to be quite time consuming and therefore problematic.

The three most difficult problems to solve; *p5a*, *p6a* and *p6b* involve respectively 0.11%, 0.22% and 0.78% of the mutations [*figure 18*], meaning in the worst case 1.11% of the BRCA1 mutations is affected by them. Here, 'the worst case' means the case where the individual problems don't overlap and are thus only captured in one problem category each. This in turn means ~98.98%

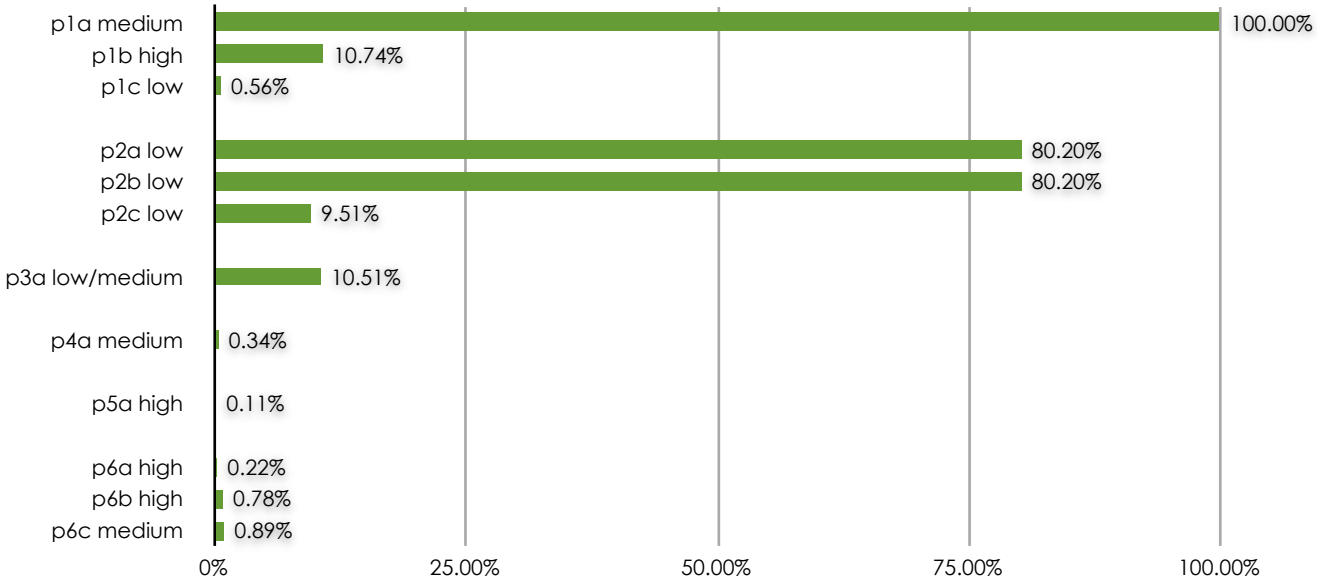


Figure 18: The encountered problems, their severity rating and the amount of mutational entries they affect.

of the BRCA1 mutational entries in HGMD should be loadable into CSHGDB, either easily or with some effort.

5.6 CAUSES

The main research topic of this project was not to investigate the underlying causes of the encountered problems. However, it is possible to some extend to identify possible causes and link them to the problems they are most likely to be involved with [*table 5*]. Also in some cases, for instance *p6b: source referring to nucleotides, not appearing at that location*, the underlying cause is clear and the explanation adds significantly to the comprehension of the problem at hand. For this reason it was felt this section had to be included; although be it not exhaustive, it adds to the general comprehension of the encountered problems.

	g1		*m1*	*m2*	*m3*		*s1*	*s2*	*s3*	*s4*
p1a				x						
p1b				x						x
p1c				x						
p2a				x						
p2b				x						
p2c				x						
p3a	x			x						x
p4a	x		x				x		x	
p5a			x		x		x	x	x	
p6a					x		x	x	x	
p6b			x		x		x	x	x	
p6c				x						x

Table 5: Causes underlying the encountered problems during the BRCA1 variational data insertion.

General causes:

Causes attributable to neither the medium, nor the source are considered to be of general nature, and thus fall into this category. In this case only one cause has been identified here and involves the effects of the fact the field is still relatively young and therefore not exhaustive.

g1 *immatureness of the field.*

Causes attributable to medium:

The causes clearly attributable to the medium (HGMD in this case) are classified into this category. They include human data entry error, which is inevitable to some extend, but also more serious issues related to the medium's design decisions and structural flaws.

m1 *human data entry error in medium.*
m2 *medium design decision.*
m3 *medium not facilitating source using different reference sequences.*

Causes attributable to source:

Here the causes attributable to the source (mostly scientific papers) are categorized. Sometimes the papers simply present erroneous data as a result of flawed research methods, and although it is impossible to check this from our point of view, the eventuality needs to be kept in mind. Here also human data entry error might be playing part as well as design decisions.

s1 *results of source being incorrect.*
s2 *source not indicating reference sequence.*
s3 *human data entry error in source.*
s4 *source design decision.*

P6b: source referring to nucleotides, not appearing at that location is the problem instance that indicates the eventuality of HGMD indicating inconsistently nucleotides throughout it's data-set. The raw fact HGMD indicates nucleotides inconsistently is disturbing and suggests a major flaw in the HGMD data-set. To understand this, some knowledge about how HGMD locates it's mutations is required, the past sections ('problem instances' and 'problem analysis') have discussed this earlier. Some causes underlying p6b are possible, it might be due to human error on either the medium (m1) or the source side (s3) or the source might simply be providing erroneous data (s1). However, the most probable cause is the use or incapability of coping with different, or inadequate reference sequences by either the medium (m3) or the source (s2). This claim is supported by findings in splicing mutations in which five mutation entries refer to nucleotides, not appearing in the splice junctions overview, see [*table 6*] for an overview of these mutational entries.

		Source paper	**Reference Sequence**
1	CS012667	[*Gao, 2000*]	*Unknown*
2	CS045210	[*Hedau, 2004*]	U14680
3	CS991331	[*Panguluri, 1999*]	U14680
4	CS951360	[*Matsushima, 1995*]	*Source paper unaccessible*
5	CS991330	[*Panguluri, 1999*]	U14680

Table 6: The five inconsistently referenced mutations in the HGMD splicing mutations data-set.

HGMD uses a splice junctions overview to locate it's splicing mutations, this method has been described and explained in the 'problem instances' section, part p2c. In short, HGMD constructs this so called splice junctions overview by extracting nucleotide sequences up- and downstream from splice junctions, the borders between exons and introns where the spliceosome splices the RNA string. For a more complete explanation on the splicing process please consult the 'introduction' section of this paper and [*Graveley, 2001*]. However, HGMD uses a non-standard reference sequence (Genebank accession number L78833), instead of NCBI RefSeq NG_005905.1. As can be seen from [*table 6*], 3 of 5 inconsistent mutations are described by papers using a different reference sequence, namely U14680.

5.7 SOLUTIONS

Inserting the 804 precise and 90 imprecise variations provided by the non-commercial version of HGMD manually seemed like a cumbersome and more importantly, an error prone operation. For this reason a series of PHP-MySQL scripts [*appendix 3: script source*] was devised to automate this procedure, while at the same time providing useful experience and knowledge about how to further process the variational data automatically. The basic function of the software is to transform the HGMD provided mutational data into the format used by CSHGDB and detecting inconsistencies in the data. It's main tools are the cDNA-sequences provided by HGMD [*Genebank, U14680*] [*Genebank, L78833*], the NCBI DNA reference sequence [*NCBI NG_005905.1*] and the NCBI 'Coding Sequences' (CDS, located in NCBI RefSeq NG_005905.1). The HGMD cDNA sequence is the aggregation of all coding sequences of the BRCA1 gene. The NCBI DNA sequence then is the complete sequence of the BRCA1 gene, including introns. The CDS information then specifies what parts of the NCBI DNA sequence are coding and which are not. The scripts extract, and in some cases, calculate the variables which are to be inserted into the CSHGDB Variation, Precise and Imprecise tables. At the same time detecting inconsistencies in the HGMD database that can not be resolved in an automated way, indicating a manual approach is needed in those cases. Since dealing with the above mentioned problems was necessary to devise those PHP scripts, they are considered to be the crystallized solutions to the earlier mentioned problems.

The main disadvantage of this 'data-mining' approach, using PHP-MySQL scripts is it's high vulnerability to changes in HGMD data structure, thereby rendering the scripts unusable. This means the scripts, and any data-mining script, will require regular updating. Also, by depending on a medium like HGMD, trust is invested in the integrity of the source. However, at this point the exact reliability of HGMD has not been identified and some of the encountered problems during this project clearly suggest reasons to doubt this reliability.

Each of the encountered problems was solved in a different way, and they will be discussed separately in the upcoming sections.

5.7.1 p1: data extraction

(a) medium using HTML tables to convey information
(b) medium using natural language
(c) medium using HTML mouse-over tags to convey information

Solving the data extraction problems involved copy-pasting the HTML-tables with mutational data, provided by HGMD, into the programming logic of the scripts. By using the PHP explode-function the data is then cut into bite-size chunks and stored in an array, ready for further processing. Due to choosing this approach, information contained in HTML mouse-over tags is not captured. A simple solution to this problem would be to copy-paste the HTML source-code instead of the browsers' rendering, effectively including the HTML tags and thus the mouse-over contained information. Then use a more elaborate algorithm to extract the pieces of information from the source. Also, since many biological information sources present their information in HTML tables as discussed by [*Stein, 2002*], many solutions to this specific problem have been devised, although be it no silver bullet solution exists today. [*www.open-bio.org*] presents some re-usable code libraries directed at solving this particular problem, thus indicating a lot is possible. However, the information contained in these tags seems to be highly unstructured to a degree that is might be considered natural language and thus be dealt with difficulty in an automated way. Also, the information structure of the different occurrences of the problem differs highly, as is discussed in the 'problem analysis' section, part p6c. So a more generic solution has been chosen to solving this problem: instead of using the information HGMD provides to locate the mutation, the PHP script simply takes the sample string provided by HGMD and matches this to the entire BRCA1 gene, given that the string is unique, a location will be found and presented.

Solving the natural language problem in this instance was possible by using a work-around consisting of a string matching strategy, as will be discussed in category p2: splicing. Although solving this particular instance proved possible due to the fact the natural language contained non-vital information, future occurrences might be more difficult and no satisfying resolution exists, due to the fact coping with natural language is inherently is a weakness in computer technology.

5.7.2 p2: data transformation

(a) medium using codon referenced notation
(b) medium using cDNA referenced notation
(c) medium using splice-junction referenced notation

The way HGMD locates it's various variations, is very distinct from the format in which CSHGDB stores the information. Transformation of this data is thus required. Each of the HGMD variation types, undergoes a distinct transformation operation, depending on how HGMD indicates it's location within the cDNA and will be discussed separately.

Missense/nonsense

HGMD indicates a missense/nonsense location by providing a codon (p2a) number, referencing cDNA (p2b) plus an offset. So, since CSHGDB requires a DNA referenced nucleotide position, transformation of this data is required. First, the software composes it's own cDNA sequence by extracting and merging the coding sequences from the NCBI DNA sequence using the NCBI

CDS information. It then matches the composed cDNA sequence to the HGMD cDNA to detect errors, which might lead to incorrect output later on. Because a missense/nonsense mutation is considered to be a single nucleotide change, HGMD's codon based reference style is somewhat imprecise since the variation may happen in any of the three nucleotides comprising the codon. The software counters this by determining the exact position of the mutation within the codon, it uses the original and substituted codon (ATG > GTG) provided by HGMD and character-matching for this. Subsequently, it multiplies the codon number by 3, in order to acquire a nucleotide referenced scale. It adds to this number the exact location of the mutation within the codon hereby obtaining the exact location of the mutation, referenced on a nucleotide cDNA scale. Ultimately, to acquire the correct location within the DNA, a calculation involving the NCBI CDS and nucleotide location of the mutation in the cDNA has to happen. The software calculates the length of each coding sequence and each non-coding sequence in the DNA, according to the NCBI CDS. It then identifies the amount of non-coding nucleotides between the start of the NCBI DNA sequence and the mutation position, then adds this number to the cDNA referenced nucleotide location, resulting in the DNA referenced nucleotide position of the mutation.

Splicing

Solving the problems affiliated with the manner in which HGMD locates Splicing mutations, seems rather straightforward at first: by simply using the splice junctions overview provided by HGMD, every mutation should be located (p2c). However due to the discovery that HGMD's Splicing mutations overview poor correspondence to it's own Splice Junctions overview as is discussed in section 5.4.6, an alternative solution had to be devised. In this case the PHP/MySQL script uses a 'matching strategy' in which it grabs the given nucleotide sequence from the Splice Junctions overview and matches this to the reference sequence provided by NCBI, thereby locating the location of the mutation more reliably.

Regulatory

The HGMD non-commercial database does not indicate any regulatory problems in the BRCA1 gene. However, according to the HGMD website, the HGMD commercial database does hold 3 mutations of this type so in order to gain experience with this type of mutation one has to acquire access to the commercial version of HGMD.

Small Deletions / Small Insertions / Small Indels

These three types of variations are located by HGMD in exactly the same manner, therefore they are discussed together. HGMD uses a codon referenced cDNA scale (p2a and p2b) to locate these variations, just like with missense/nonsense mutations. However, in this case, the mutation can involve up to 20 bp's and therefore often happens 'outside' the referenced codon, either to it's 3' or the 5' side. In this case, the transformation software calculates the given codon location in very much the same way as with the missense/nonsense mutations, by matching the NCBI CDS data with the codon number, resulting in a nucleotide position on a DNA scale for the first codon base. The software then calculates the offset in nucleotides between the referenced codon, and the actual start of the mutation, adding (or subtracting, depending on where the mutation starts in relation to the reference codon, 3' or 5') this quantity to the nucleotide position of the codon in the DNA, resulting in the DNA referenced nucleotide start position of the mutation.

Gross Deletions / Insertions / Complex rearrangements / Repeat Variations

Since these type of mutations are considered to be imprecise, no data about them needs processing. In this case, the script simply appends an identifier and applies the values to the corresponding cells in the CSHGDB Variation and Imprecise tables.

5.7.3 p3: data ambiguity

(a) medium indicating mutation phenotype ambiguously

Absolutely solving this problem would include rereading all the papers HGMD used to populate her database, indicating probability scores to mutations on their phenotype. However, this approach would be impossible to automate due to the highly unstructured nature of scientific papers. Hence another solution could be suggested, simply modifying the CSHGDB to account for the uncertainty by adding a certainty attribute to the variational table. HGMD presents on it's background pages an indication of how to interpret this uncertainty by providing the inclusion criteria for Disease-Associated/Functional Polymorphism's [*HGMD, Polymorphisms*].

5.7.4 p4: incomplete data

(a) medium lacking data entry

As mentioned, HGMD can be lacking data entries for two reasons. HGMD might simply not be up-to-date with the latest information provided by scientific research on the subject, or HGMD might have 'missed' entries during the manual loading of the database. A partial solution to this problem might be to use the professional version of HGMD, which includes more and more up-to-date entries. However, it is also a characteristic of the immatureness of the field that no 100% coverage exists, simply not enough research has been performed to identify all existing mutations. Therefore the database will inherently be incomplete, and thus resolving it falls outside the scope of this project.

5.7.5 p5: inadequate data

(a) medium providing erroneous data

HGMD can provide erroneous data for a variety of reasons, human error on either HGMD or the source paper side might be an issue or inconsistencies in the HGMD and source paper notation might play a role. In either case, this is a very difficult problem category to detect since detection should take place by rereading the papers, which as has been mentioned a couple of times earlier already is very difficult to automate. A possible, but unsatisfactory to some degree, solution would be to perform a deep investigation on a limited

amount of mutational data provided by HGMD, thereby uncovering error frequency. This error frequency, provided it is investigated according to scientific measures, can then be extrapolated to the rest of the database, hence providing a handle from which to calculate reliability of the data-set.

5.7.6 p6: inconsistent data

(a) medium referring mutation to non existing intron
(b) medium referring to nucleotides, not appearing at that location
(c) medium using inconsistent way of locating mutations

Since inconsistencies can be detected, a manual check of the inconsistent data is possible. Indeed, the scripts indicate only a few inconsistencies and so the shear amount of papers to be read manually can be reduced drastically, making the manual approach possible in most of these cases.

In this case the source is locating a mutation within a non existing intron (p6a), the solution might be to manually complete the 'splice junctions overview', combining the information from the NCBI reference sequence and coding sequences information (CDS), however as has been explained in the '*problem analysis*' section part p6a, contradiction exists about whether an intron 4 actually exists. The implications of this, as has been discussed earlier that reference papers are using different reference sequences, are far more serious and indicate a structural flaw in the HGMD splicing mutations data-set since no facilities to indicate such differences exist within HGMD, neither indicate the involved reference papers exactly what reference sequence they are using. Therefore reasons exist to seriously doubt the Splicing mutations integrity. Solving the problem would include detecting the irregularities handling them manually, and in the worst case simply considering them to be unreliable. However, detection depends on whether a given sample corresponds to the actual nucleotide occurrence at that position in the DNA sequence. Since HGMD only provides a single nucleotide sample for splicing mutations, the odds of a nucleotide at any position in the DNA corresponding to it is 25%, reducing error detection reliability greatly.

As has been mentioned, in some cases the HGMD mentioned mutation involves a nucleotide change where the to-be changed nucleotide is actually different in the reference DNA gene sequence (p6b), thereby indirectly suggesting an error on either the source paper side or the HGMD data entry side. Except for splicing mutations, as has been discussed earlier, most of these inconsistencies can be detected with relatively high reliability. This is possible because HGMD gives a sample sequence surrounding the mutations, that can be used as a handle to see whether the mentioned nucleotide actually is the one on the calculated position in the DNA sequence. In case of an inconsistency between the reference, and the given sample sequence a manual approach is possible to investigate further.

HGMD has a tendency to disrupt it's own structure, by providing different ways of locating mutations within the same mutation type (p6c). This complicates an automated approach, however a generic solution to this problem has been devised. Since the main difficulty here is the uncertainty about whether HGMD has more tricks up it's sleeves and might present it's locational data in yet other ways, it was decided not to use HGMD's locational data at all in these exceptional cases. Instead, the earlier mentioned sample sequence given by HGMD is extracted from the HTML table by using the technique described in p1a. The location of the mutation within this sample sequence is then found by detecting a character case change, going from high case to small case. The result of this calculation is then considered to be the offset, very much alike the offset given by HGMD in for instance the missense/nonsense mutations. The entire sample sequence is then matched against the entire BRCA1 reference gene sequence, the offset is then added to the found location resulting in the mutation location. The major flaw in this approach however, is the fact that the given sample sequence might be happening more then once in the BRCA1 gene, thereby compromising this methods reliability to some extend. For this reason, this approach is only considered usable in the case the regular method, as described in section '*solutions*' category p2, which is considered to be more reliable due to it's more rigid structure, fails.

6 BRCA1 AND NF1 COMPARISON

At the same time this paper was written, a similar project took place with the NF1 gene as subject. The raw data this project returned has been used in a comparison between the two genes in order to gain some insights on to what degree the loading of the genes was accompanied by the same problems. [*figure 19*] and [*figure 20*] visualize the occurrence % and mutational % respectively for each gene. [*table 7*] provides an overview of the encountered mutational data loading problems in the BRCA1 gene while [*table 8*] provides an overview of the encountered problems in the NF1 gene. In both tables certain values are of the format x..*, in which the x represents a number, meaning in these cases the problem was encountered an x number of times, but certainty is high more instances exist. Due to detection difficulties however, instances occurrences for these problems could not be reliably identified.

[*figure 19*] shows the occurrence % for both the NF1 and BRCA1 encountered problems. Thus uncovering the percentage to which an encountered problem accounts for in the total amount of encountered problems. Small differences are present and manifest themselves in problem categories p2a, p2b, p2c and p3a. In the case of problem category p2a/b/c it simply means the NF1 contains relatively more mutations of the type 'splicing mutations' that are located using a splice junctions method describes in p2c (BRCA1=9.51%, NF1=19.12%). Since this means there are relatively less mutations in the other types, Missense/nonsense, Small Insertion, Small Deletion, Small Indel, that use locating mechanisms described in p2a (BRCA1=80.20%, NF1=69.30%) and p2b (BRCA1=80.20%, NF1=69.41%), the difference in occurrence % is thus explained. The difference in *p3a: source indicating phenotype ambiguously* (BRCA1=10.51%, NF1=0.00%) illustrates the fact 94 mutational entries in the BRCA1 gene don't have a clear phenotype, while NF1 presents none.

It is clear that *p1a: source using HTML tables to convey information* is also a major problem in the NF1 gene data-set. Exactly the same mutational % is recorded (BRCA1=100.00%, NF1=100.00%), indicating that all the mutation entries of these two genes are presented in HTML tables. Thus showing NF1 suffers from the same difficulties associated to this problem as the BRCA1

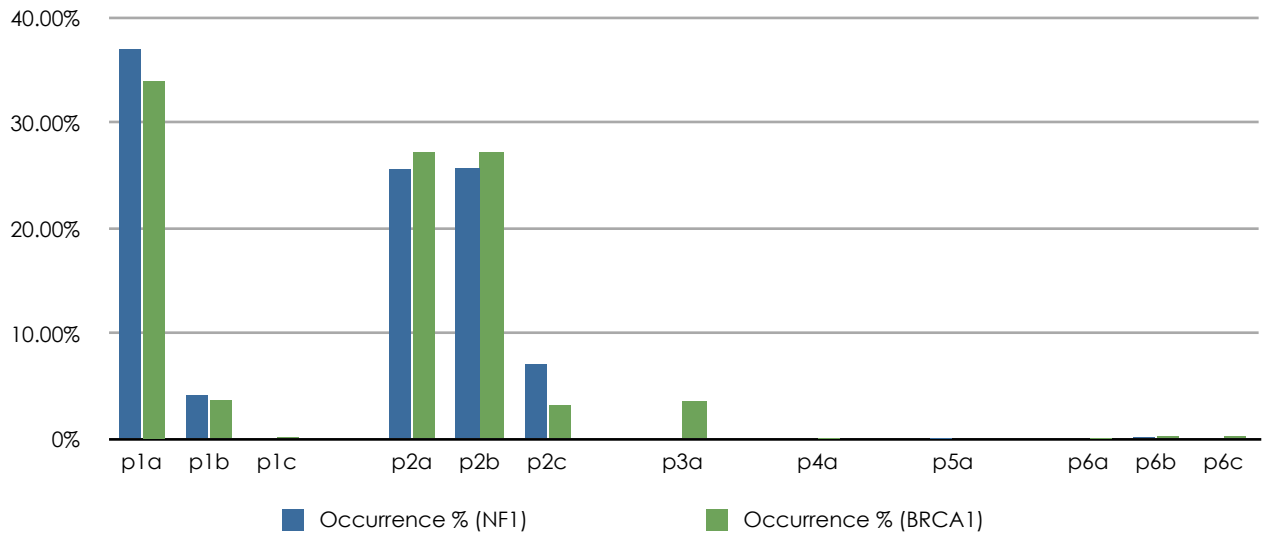


Figure 19: Occurrence % for both the NF1 and BRCA1 encountered problems.

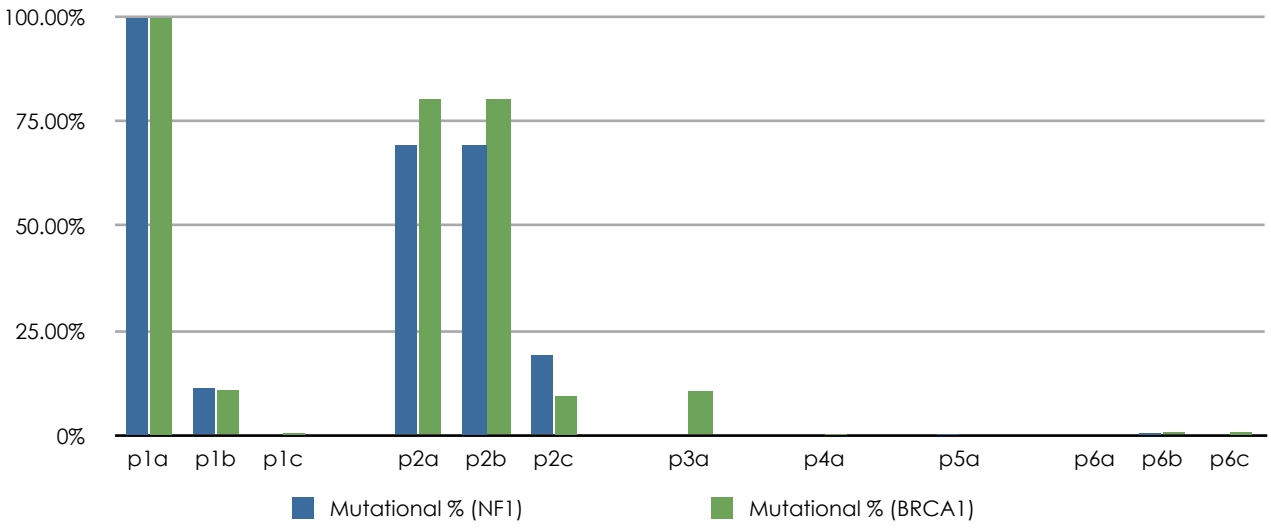


Figure 20: Mutational % for both the NF1 and BRCA1 encountered problems.

gene, further emphasizing the importance of a resolution to this type of problem.

[*figure 21*] and [*figure 22*] provide an overview of the occurrence and mutational percentages as absolute values. Thereby revealing in which problem categories exist most differences between NF1 and BRCA1. The charts are BRCA1 based, meaning a positive value in the chart corresponds to a higher score for BRCA1 on that particular problem category then NF1. [*figure 21*] roughly acknowledges the findings extracted from [*figure 19*]. The main differences in occurrence % exist in the problem categories p1, p2 and p3. Namely p1a, p2a, p2b, p2c and p3a display slight differences between the two genes. This means, although NF1 and BRCA1 encountered problem patterns are very alike, a slightly different distribution can be observed. The p2a, p2b and p3a seem to be making up a larger part of the total of encountered problems in the BRCA1 gene then in the NF1 gene, while p1a and p2c appear to be making up a smaller part of the encountered problem distribution.

[*figure 22*] roughly confirms the findings extracted from [*figure 20*]. The main differences observed here between the BRCA1 and NF1 gene are located in categories p2 an p3. According to the data, problems p2a, p2b and p3a are affecting more mutations in the BRCA1 gene then in the NF1. At the same time, p2c affects less mutations in BRCA1 then in NF1.

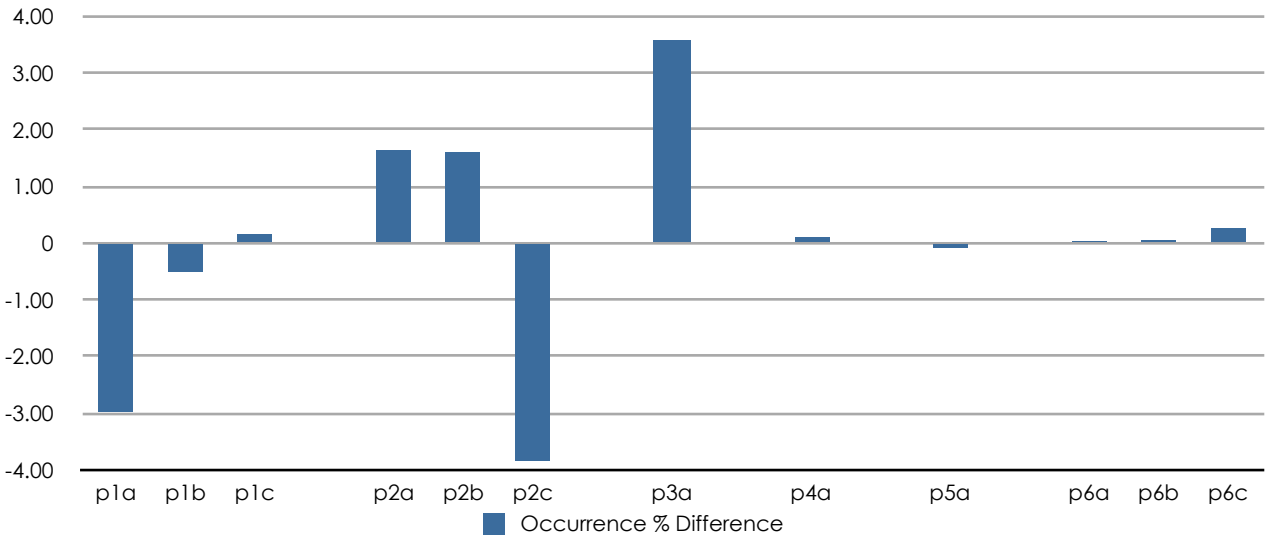


Figure 21: Occurrence % Differences expressed as absolute values, BRCA1 based.

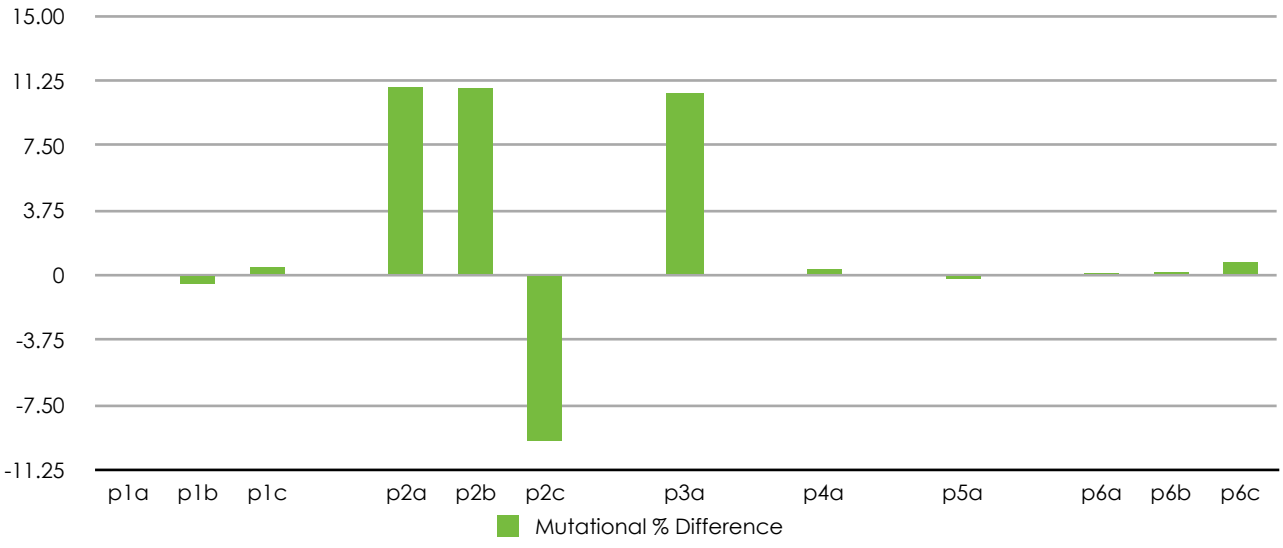


Figure 22: Mutational % Differences expressed as absolute values, BRCA1 based.

	Missense/nonsense	Regulatory	Splicing	Small deletions	Small insertions	Small indels	Imprecise	Mutation total:	Mutational %	Occurrence %
Mutations:	320	0	80	292	101	11	90	**894**	*100.00*	
p1a: HTML extraction	320	0	80	292	101	11	90	**894**	*100.00*	*34.01*
p1b: Natural Language extraction	0	0	1	2	3	0	90	**96**	*10.74*	*3.65*
p1c: Tags extraction	0	0	0	2	3	0	0	**5**	*0.56*	*0.19*
p2a: Codon transformation	320	0	0	288	98	11	0	**717**	*80.20*	*27.27*
p2b: cDNA transformation	320	0	0	288	98	11	0	**717**	*80.20*	*27.27*
p2c: Splice-junction transformation	0	0	80	2	3	0	0	**85**	*9.51*	*3.23*
p3a: Phenotype ambiguity	73	0	16	2	3	0	0	**94**	*10.51*	*3.58*
p4a: Data entry lacking	0..*	0	3..*	0..*	0..*	0..*	0..*	**3**	*0.34*	*0.11*
p5a: Inadequate data	0..*	0	1..*	0..*	0..*	0..*	0	**1**	*0.11*	*0.04*
p6a: Non-existing intron	0	0	2	0	0	0	0	**2**	*0.22*	*0.08*
p6b: Inconsistent nucleotide references	0	0	5	1	1	0	0	**7**	*0.78*	*0.27*
p6c: Inconsistently locating nucleotides	0	0	0	2	6	0	0	**8**	*0.89*	*0.30*
Problem Instances Total	*1033*	*0*	*184*	*879*	*316*	*33*	*180*	*2629*		
Relative Total	*39.3*	*0.0*	*7.0*	*33.4*	*12.0*	*1.3*	*6.8*	*100.0*		

Table 7: BRCA1 gene mutational loading data statistics. x..* means x or more.

	Missense/nonsense	Regulatory	Splicing	Small deletions	Small insertions	Small indels	Imprecise	Mutation total:	Mutational %	Occurrence %
Mutations:	220	0	160	237	113	14	93	**837**	*100.00*	
p1a: HTML extraction	220	0	160	237	113	14	93	**837**	*100.00*	*36.99*
p1b: Natural Language extraction	0	0	0	0	1	0	93	**94**	*11.23*	*4.15*
p1c: Tags extraction	0	0	0	0	1	0	0	**1**	*0.12*	*0.04*
p2a: Codon transformation	220	0	0	237	112	11	0	**580**	*69.30*	*25.63*
p2b: cDNA transformation	220	0	0	237	113	11	0	**581**	*69.41*	*25.67*
p2c: Splice-junction transformation	0	0	160	0	0	0	0	**160**	*19.12*	*7.07*
p3a: Phenotype ambiguity	0	0	0	0	0	0	0..*	**0**	*0.00*	*0.00*
p4a: Data entry lacking	0..*	0	0..*	0..*	0..*	0..*	0..*	**0**	*0.00*	*0.00*
p5a: Inadequate data	0..*	0	3..*	0..*	0..*	0..*	0	**3**	*0.36*	*0.13*
p6a: Non-existing intron	0	0	1	0	0	0	0..*	**1**	*0.12*	*0.04*
p6b: Inconsistent nucleotide references	0	0	1	0	4	0	0..*	**5**	*0.60*	*0.22*
p6c: Inconsistently locating nucleotides	0	0	0	0	1	0	0	**1**	*0.12*	*0.04*
Problem Instances Total	*660*	*0*	*322*	*711*	*345*	*36*	*186*	*2263*		
Relative Total	*29.2*	*0.0*	*14.2*	*31.4*	*15.2*	*1.6*	*8.2*	*100.0*		

Table 8: NF1 gene mutational loading data statistics. x..* means x or more.

7 CONCLUSIONS

The value of a full-scale conceptual model of the human genome is unmistakable and could be considered the holy grail of genomics. The potential of modifying the human source-code, as an analogy to the modification of software source-code is huge and presents unimaginable possibilities in the medical field as well as many other fields. In the quest to finding this blueprint of the human being, practical issues have to be tackled. In this paper the practical issue of loading a prototype genomic database with genetic data has been handled, problems have been reported, and a standard notation for this problem reporting is presented. Most of the problems were resolved and advise on future research has been devised, which will be discussed shortly in the upcoming paragraphs.

The main goal of this project was populating CSHGDB with the BRCA1 gene. The main focus of the project was directed at loading CSHGDB with mutational data from the HGMD database. During this process problems were encountered, analyzed and most of them resolved. A standard notation for these problems has been suggested, allowing for easier and more efficient comparisons in future gene loading project. The project shed light on how to further automate the loading procedure of CSHGDB by presenting various PHP-MySQL scripts [*appendix 3: script source*]. These scripts are at the time of writing not fully functional and further investigation is thus advised. However, they serve as a proof of concept and present functional algorithms to solving many of the encountered problems. A comparison with the NF1 gene was performed, in order to uncover similarities and thus present a vision on how resolutions to encountered problems in the BRCA1 gene are applicable to the ones found in the NF1 loading procedure. The report provides 3 contributions to the field:

- Genomic mutational data loading scripts
- A notational framework for genomic mutational data loading problems
- A database loaded with BRCA1 mutational data

The main conclusion of this report is that loading CSHGDB from HGMD can be automated to a large extend, probably more than 98%. The comparison between NF1 and BRCA1 clearly show high similarity in encountered problems, thus indicating the same resolutions can be applied. The only detected difference of significance, BRCA1 presents some mutational entries with ambiguous phenotype and NF1 does not, does not appear to affect CSHGDB to great extend. The main problem here is the way HGMD presents genomic mutational data, notably the use of HTML tables, which significantly hinders data-mining.

More worrying are the indications that HGMD might be structurally flawed, mainly by not facilitating a proper way of dealing with different reference sequences. Lessons should be learned from this and taken into account in the further development of CSHGDB by facilitating different reference sequences. The inability of HGMD dealing with this type of irregularity also indicates a flaw in the genetics field in general. Apparently scientific papers presenting their results are free to choose what reference sequence they refer their results to. Some structure does seem to exist as all papers refer to one or more of three reference sequences, but the apparent absence of a standard on how to indicate the location of found mutations by scientific papers unnecessarily complicates affairs. Although outside the scope of this project, presenting a common accepted standard for mutational descriptions would be desirable and could thus be considered a topic of further research.

A standard way of describing mutations, not only their nature and phenotype but also location within the genome, would eventually contribute largely to CSHGDB integrity and reliability. If scientific papers widely accept this standard, thus presenting their outcomes in a standard format as opposed to natural language and/or arbitrary formats, automatic data extraction from them will be possible. This effectively renders the in-between solutions of media like HGMD defunct. In turn meaning the imperfect HGMD data-set can be bypassed, ultimately leading to a more reliable CSHGDB data-set, allowing effective and efficient research on genomic mutational data. At the same time being more future-proof and solid by not depending on the capricious character of a third party.

Coming back to the main research question: "*Compared to the NF1 gene, to what degree is it possible to map the BRCA1 gene to the existing conceptual model?*", answering it is not as straightforward. First of all, this paper's focal point lies in the variational part of the conceptual model, thereby omitting the other parts of the model largely. However at the point of writing these parts were not fully developed yet. When looking at the question from a narrower perspective and interpreting it as "*Compared to the NF1 gene, to what degree is it possible to map the BRCA1 gene variational data to the existing conceptual model?*", the answer becomes clearer. Many similarities exist between the NF1 and BRCA1 gene at this point although more genes will have to be researched in order to determine the degree to which this is applicable to other genes. The framework used in this paper to categorize problems, can be used for future research to come to a standard notation, hence facilitating and simplifying these future comparisons. In any case, all data presented by HGMD can be mapped to the conceptual model, meaning every piece of HGMD data has a corresponding entity in CSHGDB, thereby suggesting a very high degree to which the variational data can be mapped to CSHGDB. However in this view, HGMD serves as a determinant in identifying the degree to which it is possible to map the BRCA1 gene to the existing conceptual model and desirability for this is questionable for obvious reasons. So, a better question would be: "*Does the current conceptual model fully capture the concept of genetic variability*", indicating a more generic point of view by removing HGMD from the equation and allowing for a fresh look.

8 ACKNOWLEDGEMENT

Hereby I would to like to thank the University of Utrecht and the Technical University of Valencia in providing the possibilities for this project. Thanks are due to prof. Sjaak Brinkkemper and prof. Óscar Pastor López in particular for their specific role in making the project possible, guiding it and providing valuable feedback when needed. Special thanks go out to dr. Ana Levin, guiding the project on a daily basis and always providing essential feedback in keeping it on track. Of course, thanks go out to the entire Genoma team, creating a productive working climate and hospitality in receiving a foreign researcher.

9 REFERENCES

1. [Alberts, 2003] B. Alberts, D. Bray, K. Hopkin, A. Johnson, J. Lewis, M. Raff, K. Roberts, P. Walter. *Essential Cell Biology* 2nd *edition,* (2003), **Editors:** E. Zayatz, E. Lawrence. Garland Science USA.

2. [Ahnert, 2008] S.E. Ahnert, T.M.A. Fink, A. Zinovyev (2008). *How much non-coding DNA do eukaryotes require?* **Journal of Theoretical Biology**, volume 252, issue 4, pp: 587-592.

3. [Bornberg-Bauer, 2002] E.W. Bornberg-Bauer, N.W. Paton (2002). *Conceptual data modeling for bioinformatics*. **Briefings in bioinformatics**, volume 3, issue 2, pp: 166-188.

4. [Chen, 1995a] Y. Chen, C. Chen, D.J. Riley, D.C. Allred, P. Chen, D. Von Hoff, C.K. Osborne and W. Lee (1995). *Aberrant Subcellular Localization of BRCA1 in Breast Cancer.* **Science**, volume 270, issue 5237, pp: 789-791.

5. [Chen, 1995b] I.-M.A. Chen, V. Markowitz (1995). *Modeling scientific experiments with an object data modeling*. In **Proceedings of the SSDBM**. IEEE Press, pp: 391-400.

6. [Coates, 1997] J.F. Coates, J.B. Mahaffie, A. Hines (1997). *The Promise of Genetics.* **Futurist**, volume 31, issue 5, pp: 18-23

7. [Collins, 2004] F.S. Collins, E.S. Lander, J. Rogers, R.H. Waterston (2004). *Finishing the euchromatic sequence of the human genome*. **Nature**, volume 431, pp: 931-945.

8. [Panguluri, 1999] R. Panguluri, G. Dunston, L. Brody, R. Modali, K. Utley, L. Adams-Campbell, A. Day, C. Whitfield-Broome (1999). *BRCA1 mutations in African Americans*. **Human Genetics**, volume 105, issue 1-2, pp: 28-31.

9. [Ford 1994] D. Ford, D.F. Easton, D.T. Bishop, S.A. Narod, D.E. Goldgar (1994). *Risks of cancer in BRCA1-mutation carriers*. **Lancet (North American edition)**, volume 343, issue 8899, pp: 692-696.

10. [Gallinger, 2008] S. Gallinger, W. Al-Sukhni, H. Rothenmund, A. Eppel Borgida, G. Zogopoulos, A. O'Shea, A. Pollet (2008). *Germline BRCA1 mutations predispose to pancreatic adenocarcinoma*. **Human Genetics**, volume 124, issue 3, pp: 271-278.

11. [Gao, 2000] Q. Gao, G. Tomlinson, D. Das, S. Cummings, L. Sveen, J. Fackenthal, P. Schumm, O.I. Olopade (2000). *Prevalence of BRCA1 and BRCA2 mutations among clinic-based African American families with breast cancer*. **British journal of cancer**, volume 83, issue 10, pp: 1301-1309.

12. [Gerstein, 2007] M.B. Gerstein, C. Bruce, J. Rozowosky, D. Zheng, J. Du, J. Korbel, O. Emanuelson, Z. Zhang, S. Weissman, M. Snyder (2007). *What is a gene, post-ENCODE?* **Genome Research**, volume 17, issue 6, pp: 669-681.

13. [Graveley, 2001] B.R. Graveley (2001). *Alternative splicing: increasing diversity in the proteomic world*. **Trends in genetics**, volume 17, issue 2, pp: 100-107.

14. [Graves, 1996] M. Graves, E.R. Bergeman, C.B.A. Lawrence (1996). *Graph conceptual model for developing human genome center databases*. **Computers in biology and medicine**, volume 26, Issue 3, pp: 183-197.

15. [Hedau, 2004] S. Hedau, B.C. Das, N. Jain, S.A. Husain, A.K. Mandal, G. Ray, M. Shahid, R. Kant, V. Gupta, N.K. Shukla, S.S.V. Deo (2004). *Novel germline mutations in breast cancer susceptibility genes BRCA1, BRCA2 and p53 gene in breast cancer patients from India*. **Breast cancer research and treatment**, volume 88, issue 2, pp: 177-186.

16. [Olson, 1993] M.V. Olson (1993). *The human genome project*. **Proceedings of the National Academy of Sciences of the United States of America (PNAS)**, volume 90, issue 10, pp: 4338-4344.

17. [Pastor, 2008] O. Pastor (2008). *Conceptual modeling meets the human genome*. **Conceptual Modeling – ER 2008**. *LNCS*, volume 5231, p.1. Springer-Verlag. Berlin-Heidelberg (2008).

18. [Pastor, 2009a] O. Pastor, A.M. Levin, J.C. Casamayor, M. Celma, A. Virrueta, L.E. Eraso, M. Perez-Alonso (2009). *Enforcing Conceptual Modeling to Improve the Understanding of Human Genome*. **Submitted to ER-2009**.

19. [Pastor, 2009b] O. Pastor , A.M. Levin, M. Celma, J.C. Casamayor, A. Virrueta, L.E. Eraso (2009). *Model driven-based engineering applied to the Interpretation of the Human Genome. In The Evolution of Conceptual Modeling*, R. Kaschek, L. Delcambre. Springer-Verlag (2009), editor: H. Mayr. (in the editing phase).

20. [Pastor, 2009c] O. Pastor, M.A. Pastor, V. Burriel (2009). *Conceptual Modeling of Human Genome Mutations: a Dichotomy between What We Have and What We Should Have*. **Submitted to ER-2009**.

21. [Paton, 2000] N.W. Paton, S.A. Khan, A. Hayes, F. Moussouni, A. Brass, K. Eilbeck, C.A. Goble, S.J. Hubbard, S.G. Oliver (2000). *Conceptual modeling of genomic information.* **Bioinformatics**, volume 16, issue 6, pp: 548-557.

22. [Pertsemlidis, 2001] A. Pertsemlidis, J.W. Fondon (2001). *Having a BLAST with bioinformatics (and avoiding BLASTphemy)*. **Genome Biology**, volume 2, issue 10, pp: 1-10.

23. [Hall, 1990] J.M. Hall, M.K. Lee, B. Newman, J.E. Morrow, L.A. Anderson, B. Huey, M. King (1990). *Linkage of Early-Onset Familial Breast Cancer to Chromosome 17q21*. **Science**, volume 250, issue 4988, pp: 1684-1689.

24. [Khoo, 2000] U. Khoo, H.Y.S. Ngan, A.N.Y. Cheung, K.Y.K. Chan, J. Lu, V.W.Y. Chan, S. Lau, I.L. Andrulis, H. Ozcelik (2000). *Mutational analysis of BRCA1 and BRCA2 genes in Chinese ovarian cancer identifies 6 novel germline mutations.* **Human Mutation**, volume 16, issue 1, pp: 88-89.

25. [Lander, 2001] E.S. Lander, et al. (2001). *Initial sequencing and analysis of the human genome.* **Nature**, volume 409, pp: 860 – 921.

26. [Langston, 1996] A. Langston, J.L. Stanford, K.G. Wicklund, J.D. Thompson, R.G. Blazej, E.A. Ostrander (1996). *Germ-Line BRCA1 mutations in selected men with prostate cancer*. **American Journal of Human Genetics**, volume 58, pp: 881-885.

27. [Martí, 2007] C.A. Martí, F.B. Quijal, C.G. Laserma, E.G. Ballester, J.M.G. De Cos González, A.T. Ramos, M.V. Raneda, Ó.Z. Lloréns (2007). *Análisis de los principales grupos tumorales - Mama*. In **Informes de Salud, Situación del Cáncer en la Comunitat Valenciana**, Generalitat, Conselleria de Sanitat, 1ª edición.

28. [Mattick, 2003] J.S. Mattick (2003). *Challenging the dogma: the hidden layer of non-protein-coding RNAs in complex organisms*. **BioEssays**, volume 25, issue 10, pp: 930-940.

29. [Mattick, 2004] J.S. Mattick (2004). *The Hidden genetic program of complex organisms*. **Scientific American**, volume 291, issue 4, pp: 60-68.

30. [Medigue, 1999] C. Medigue, F. Rechenmann, A. Danchin, A. Viari (1999). *Imagene: an integrated computer environment for sequence annotation and analysis.* **Bioinformatics**, volume 15, pp: 2-15.

31. [Miki, 1994] Y. Miki et al (1994). *BRCA1 Mutations in Primary Breast and Ovarian Carcinomas*. **Science**, volume 266, issue 5182, pp: 120-122.

32. [Okayama, 1998] T. Okayama, T. Tamura, T. Gojobori, Y. Tateno, K. Ikeo, S. Miyazaki, K. Fukami-Kobayashi, H. Sugawara (1998). *Formal design and implementation of an improved DDBJ DNA database with a new schema and object-oriented library*. **Bioinformatics**, volume 14, issue 6, pp: 472.

33. [Parmigiani, 2007] G. Parmigiani, S. Chen (2007). *Meta-Analysis of BRCA1 and BRCA2 Penetrance*. **Journal of Clinical Oncology**, volume 25, issue 11, pp: 1329-1333.

34. [Satagopan, 2001] J.M. Satagopan, K. Offit, W. Foulkes, M.E. Robson, S. Wacholder, C.M. Eng, S.E. Karp, C.B. Begg (2001). *The Lifetime Risks of Breast Cancer in Ashkenazi Jewish Carriers of BRCA1 and BRCA2 Mutations*. **Cancer Epidemiology Biomarkers & Prevention**, volume 10, pp: 467-473.

35. [Scherren, 2007] K. Scherrer, J. Jost (2007). *Gene and genon concept: coding versus regulation.* **Theory in Biosciences**, volume 126, issue 2-3, pp: 65-113.

36. [Shattuck, 2009] D. Shattuck-Eidens et al. (2009). *BRCA1 Sequence Analysis in Women at High Risk for Susceptibility Mutations.* **The Journal of the American Medical Association**, volume 278, issue 15, pp: 1242-1250.

37. [Simsion, 2005] G.C. Simsion, G.C. Witt, Editor: Morgan Kaufmann Pub (2005). *Data Modeling Essentials*. Morgan Kaufmann Series, 3rd edition USA.

38. [Stein, 2002] L. Stein (2002). *Creating a bioinformatics nation*. **Nature**, volume 417, issue 6885, pp: 119-121.

39. [Stelzl, 2005] U. Stelzl et al (2005). *A Human Protein-Protein Interaction Network: A Resource for Annotating the Proteome*. **Cell**, volume 122, issue 6, pp: 957-968.

40. [Thompson, 2002] D. Thompson, D.F. Easton (2002). *Cancer Incidence in BRCA1 Mutation Carriers*. **Journal of the National Cancer Institute**, volume 94, issue 18, pp: 1358-1365.

41. [Vallon-Christersson, 2001] J. Vallon-Christersson et al (2001). *Functional analysis of BRCA1 C-terminal missense mutations identified in breast and ovarian cancer families*. **Human Molecular Genetics**, volume 10, issue 4, pp: 353-360.

42. [Van der Hout, 2006] A. van der Hout et al. (2006). *A DGGE system for comprehensive mutation screening of BRCA1 and BRCA2: application in a Dutch cancer clinic*. **Human Mutation**, volume 27, issue 7, pp: 654-666.

43. [Venter, 2001] J.C. Venter, M.D. Adams, E.W. Myers, P.W. Li, R.J. Mural, G.G. Sutton, H.O. Smith, M. Yandell, C.A. Evans, R.A. Bolt, et al. (2001). *The sequence of the human genome.* **Science**, volume 291, pp: 1304-1351.

44. [Zhao, 2003] Z. Zhao, Y.X. Fu, D. Hewet-Emmett, E. Boerwinkle (2003). *Investigating single nucleotide polymorphism (SNP) density in the human genome and its implications for molecular evolution*. **Gene**, volume 312, pp: 207-213.

10 REFERENCED WEBSITES

1. [*HGNC, 2009*], HUGO Gene Nomenclature Commitee, European Bioinformatics Institute, Hinxton, Cambridge. Viewed 31/08/2009. <http://www.genenames.org/

2. [*HGMD, 2009*], The Human Gene Mutation Database, Institute of Medical Genetics, Cardiff. Viewed 31/08/2009. <http://www.hgmd.cf.ac.uk/ac/index.php>

3. [*NCBI, 2009*], National Center for Biotechnology Information, Bethesda. Viewed 31/08/2009. <http://www.ncbi.nlm.nih.gov/>

4. [*About the HGNC 2007]*, HUGO Gene Nomenclature Commitee, European Bioinformatics Institute, Hinxton, Cambridge. Viewed 31/08/2009. <http://www.genenames.org/aboutHGNC.html>

5. *[HGMD background 2007]*, The Human Gene Mutation Database, Institute of Medical Genetics, Cardiff. Viewed 31/08/2009. <http://www.hgmd.cf.ac.uk/docs/new_back.html>

6. [*About NCBI 2004]*, National Center for Biotechnology Information, Bethesda. Viewed 31/08/2009. <http://www.ncbi.nlm.nih.gov/About/index.html>

7. [*Genebank, U14680*], National Center for Biotechnology Information, Bethesda. Viewed 01/09/2009. <http://www.ncbi.nlm.nih.gov/nuccore/555931>

8. [*Genebank, L78833*], National Center for Biotechnology Information, Bethesda. Viewed 01/09/2009. <http://www.ncbi.nlm.nih.gov/nuccore/1698398>

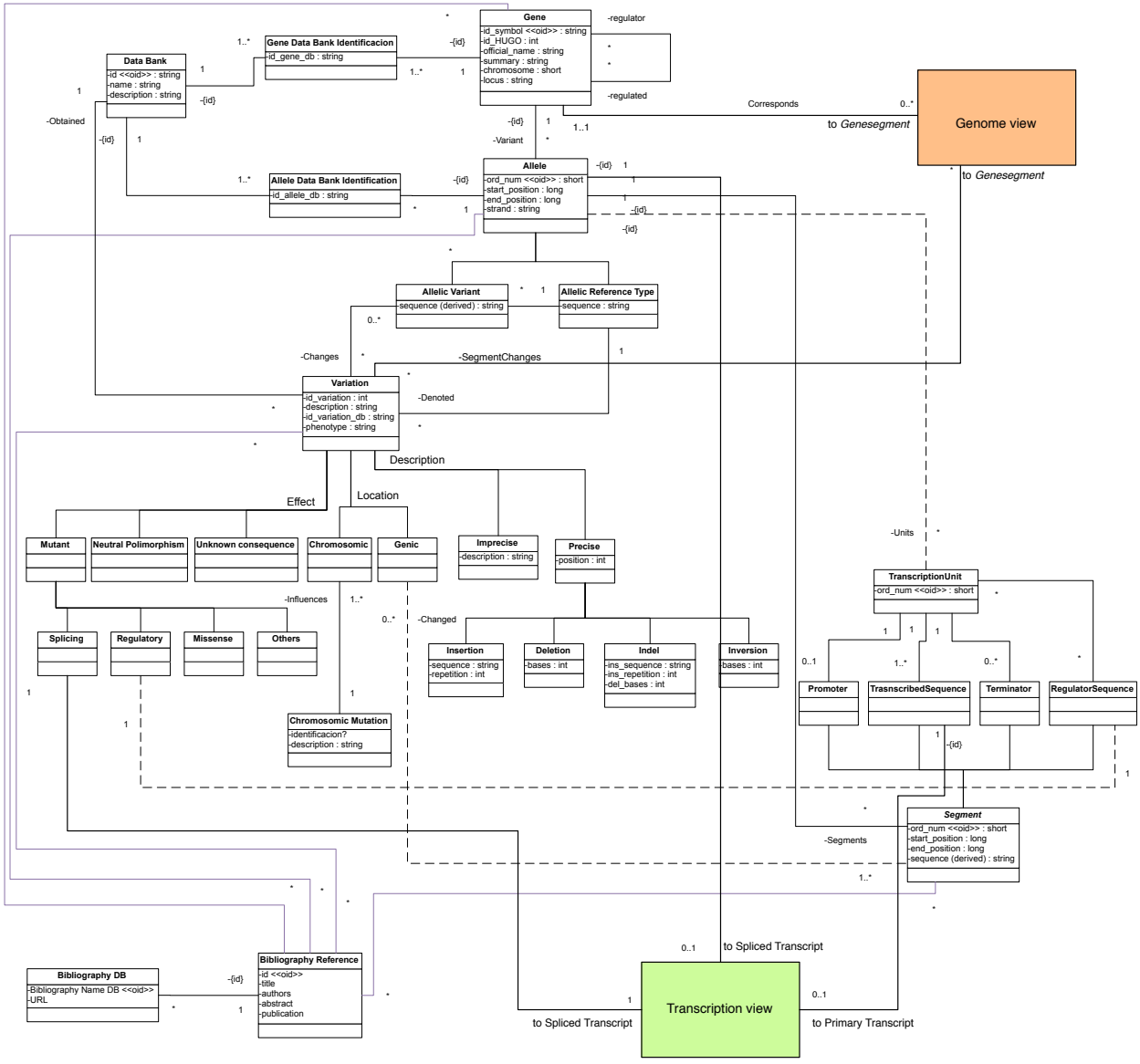
9. [*NCBI NG_005905.1*], National Center for Biotechnology Information, Bethesda. Viewed 01/09/2009. <http://www.ncbi.nlm.nih.gov/nuccore/126015854>

10. [*HGMD, Polymorphisms*], The Human Gene Mutation Database, Institute of Medical Genetics, Cardiff. Viewed 01/09/2009. <http://www.hgmd.cf.ac.uk/docs/poly.html>

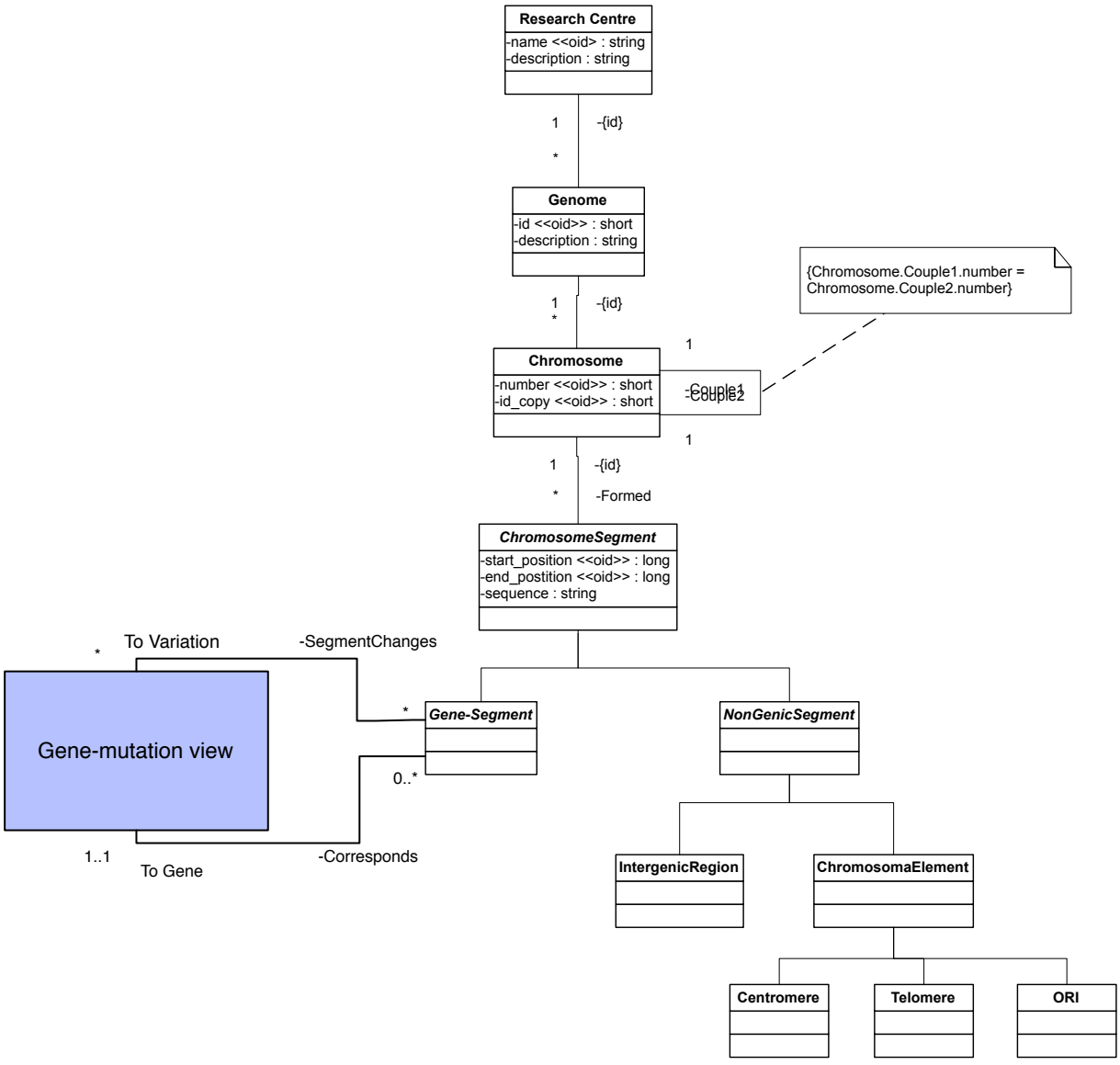
11 APPENDIX 1: CONCEPTUAL MODEL

11.1 IDEAL MODEL

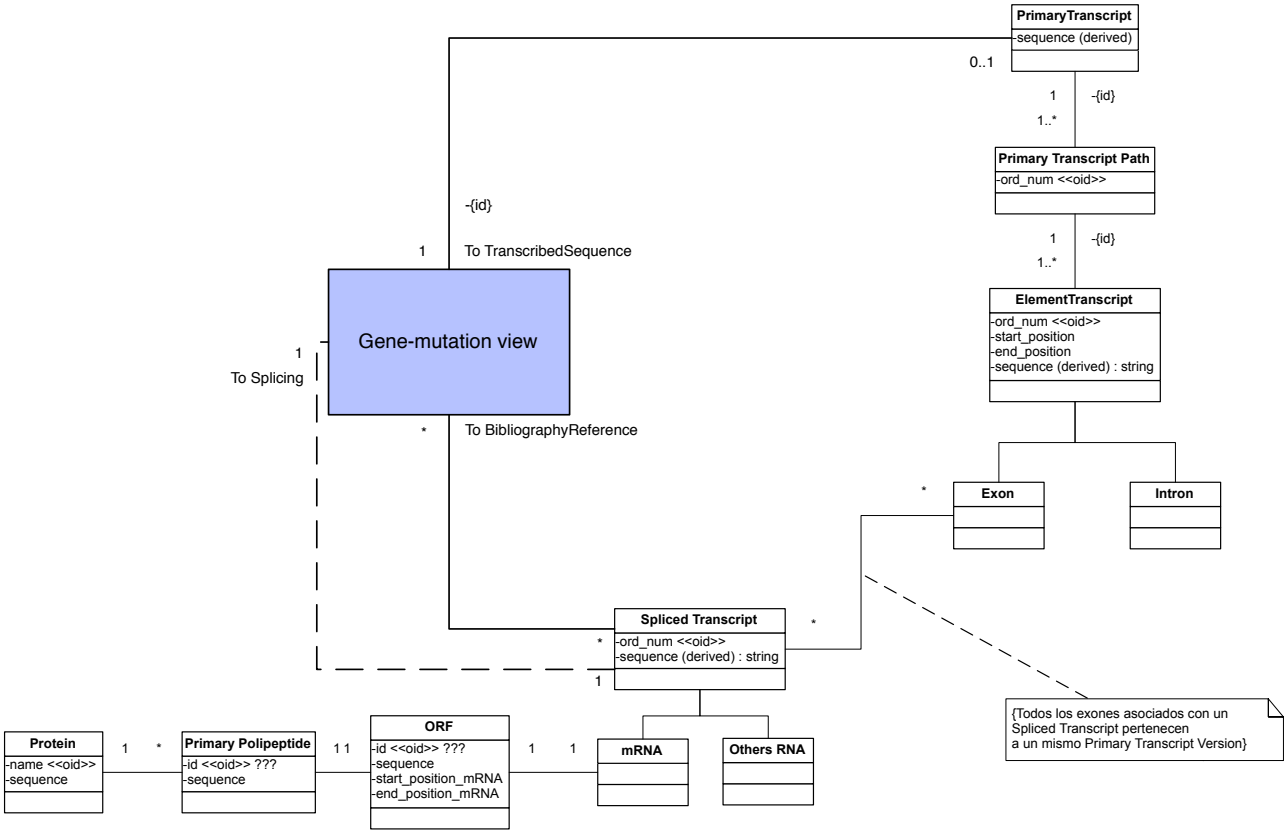
11.1.1 GENE-MUTATION VIEW



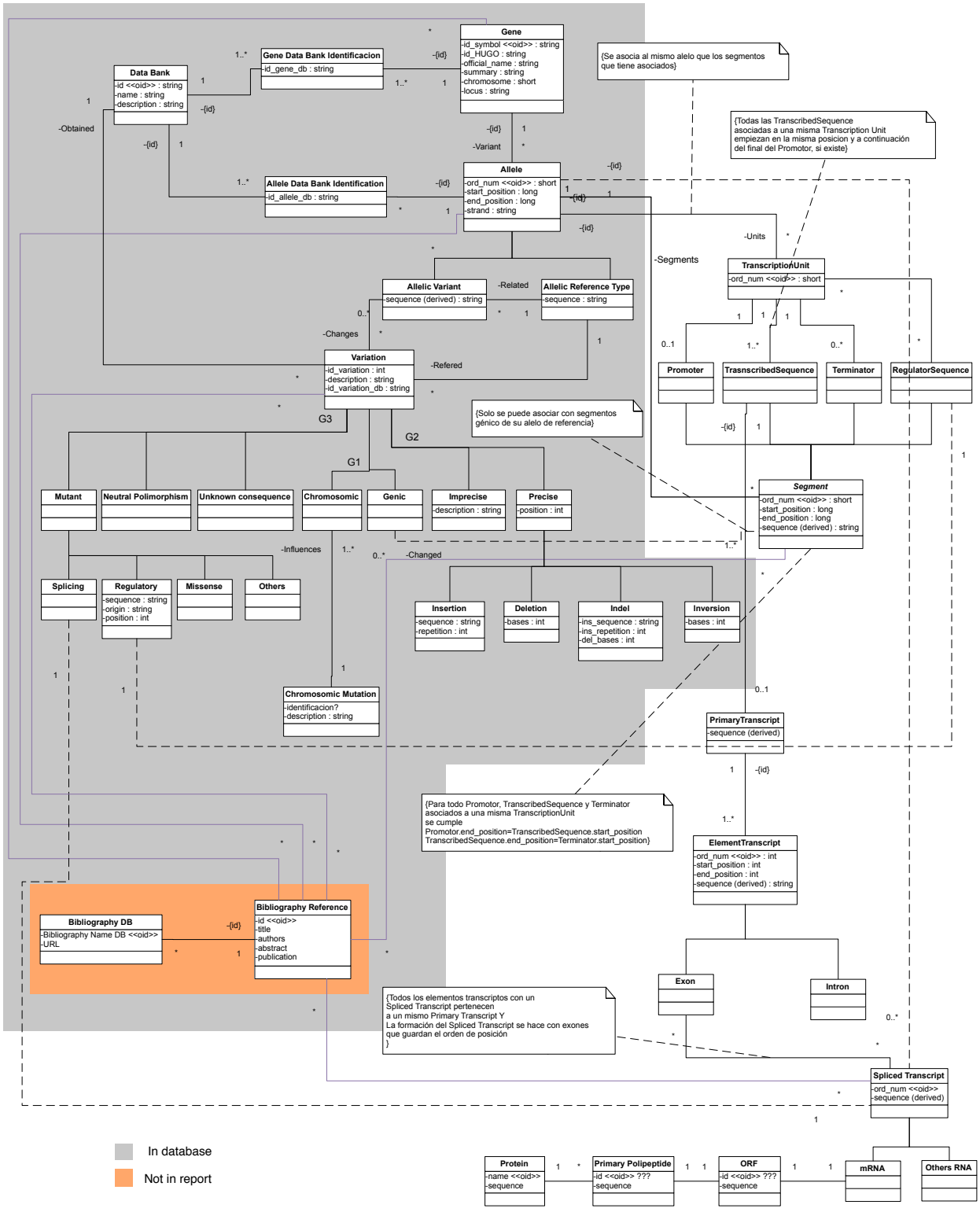
11.1.2 GENOME VIEW



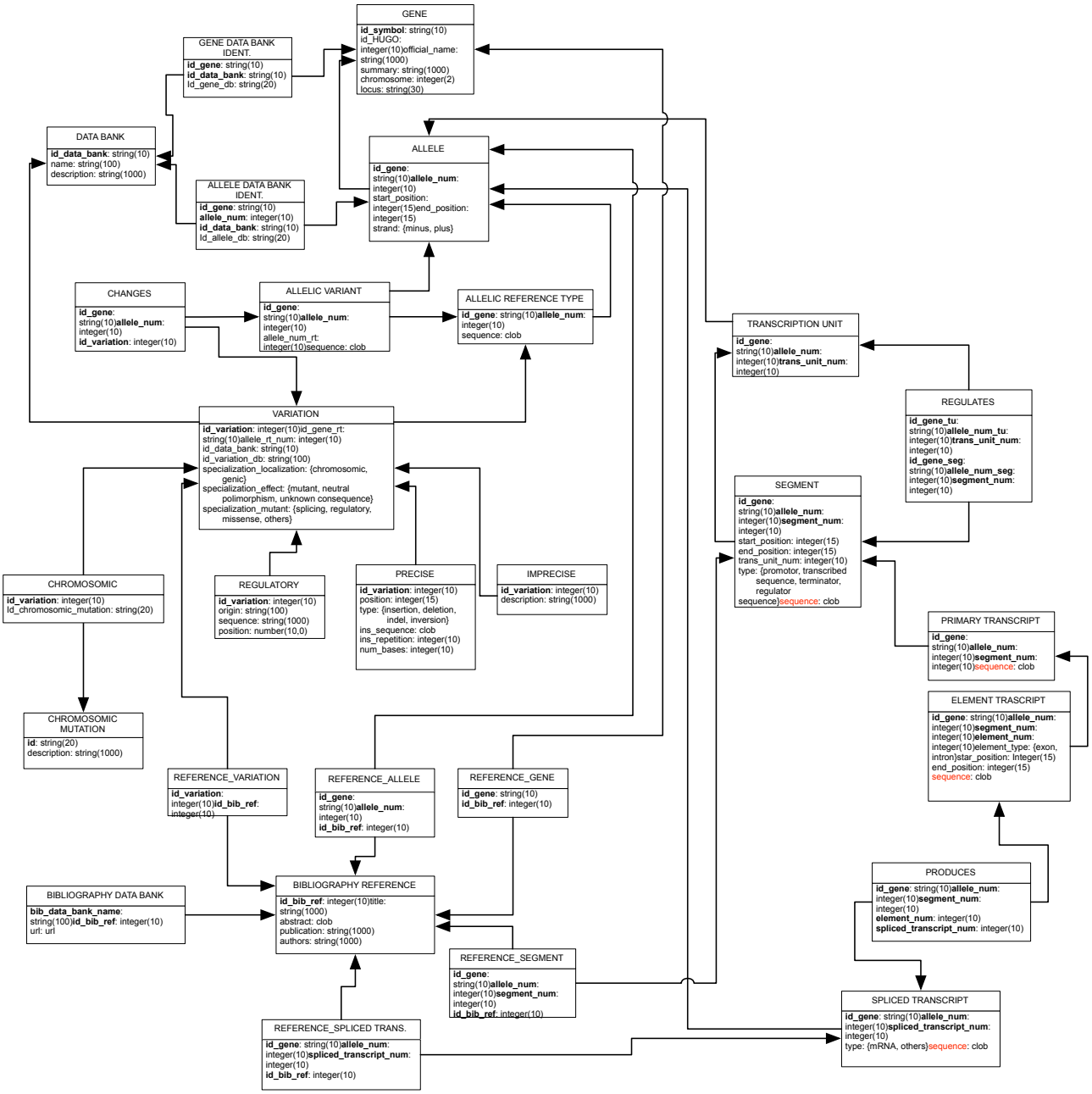
11.1.3 TRANSCRIPTION VIEW



11.2 REAL MODEL



11.3 ENTITY RELATIONSHIP DIAGRAM



12 APPENDIX 2: BRCA1 ENCOUNTERED PROBLEMS

12.1 P1: DATA EXTRACTION

12.1.1 P1A: HTML EXTRACTION

Information on every mutation held by HGMD is presented in HTML tables.

12.1.2 P1B: NATURAL LANGUAGE EXTRACTION

	Splicing	
1	-	In Splice Junctions Overview information about the exonic structure of the BRCA1 gene is provided in natural language.
	Small Deletions	
2	CD991644	The locational information about this mutation is provided through mouse-over tags. However, the provided information is very unstructured to a degree that it is considered to be natural language.
3	CD994433	The locational information about this mutation is provided through mouse-over tags. However, the provided information is very unstructured to a degree that it is considered to be natural language.
	Small Insertions	
4	CI030168	The locational information about this mutation is provided through mouse-over tags. However, the provided information is very unstructured to a degree that it is considered to be natural language.
5	CI962219	The locational information about this mutation is provided through mouse-over tags. However, the provided information is very unstructured to a degree that it is considered to be natural language.
6	CI022582	The locational information about this mutation is provided through mouse-over tags. However, the provided information is very unstructured to a degree that it is considered to be natural language.

12.1.3 P1C: TAGS EXTRACTION

	Small Deletions	
1	CD991644	Locational information on this mutation is given by HGMD only HTML in mouse-over tags
2	CD994433	Locational information on this mutation is given by HGMD only HTML in mouse-over tags
	Small Insertions	
1	CI030168	Locational information on this mutation is given by HGMD in HTML mouse-over tags, however an alternative indication of locating this mutation is provided in a conventional way as well.
2	CI962219	Locational information on this mutation is given by HGMD in HTML mouse-over tags, however an alternative indication of locating this mutation is provided in a conventional way as well.
3	CI022582	Locational information on this mutation is given by HGMD in HTML mouse-over tags, however an alternative indication of locating this mutation is provided in a conventional way as well.

12.2 P2: DATA TRANSFORMATION

12.2.1 P2A: CODON TRANSFORMATION

Almost all mutations are located using a codon referenced method. Only Splicing mutations (80 occurrences) and some exceptional cases use a different method. The exceptional cases are Small Deletion mutations (2 occurrences) CD991644, CD994433 and Small Insertion mutations (3 occurrences) CI030168, CI962219, CI022582.

12.2.2 P2B: CDNA TRANSFORMATION

Almost all mutations are located using a cDNA referenced method. Only Splicing mutations (80 occurrences) and some exceptional cases use a different method. The exceptional cases are Small Deletion mutations (2 occurrences) CD991644, CD994433 and Small Insertion mutations (3 occurrences) CI030168, CI962219, CI022582.

12.2.3 P2C: SPLICE-JUNCTION TRANSFORMATION

	Splicing	
1	CS961492	Mutations uses a Splice Junction reference location method.
2	CS012667	Mutations uses a Splice Junction reference location method.
3	CS055565	Mutations uses a Splice Junction reference location method.
4	CS993671	Mutations uses a Splice Junction reference location method.
5	CS023199	Mutations uses a Splice Junction reference location method.
6	CS021160	Mutations uses a Splice Junction reference location method.
7	CS063247	Mutations uses a Splice Junction reference location method.
8	CS011027	Mutations uses a Splice Junction reference location method.
9	CS011975	Mutations uses a Splice Junction reference location method.
10	CS021724	Mutations uses a Splice Junction reference location method.
11	CS951355	Mutations uses a Splice Junction reference location method.
12	CS032400	Mutations uses a Splice Junction reference location method.
13	CS951356	Mutations uses a Splice Junction reference location method.
14	CS971622	Mutations uses a Splice Junction reference location method.
15	CS982088	Mutations uses a Splice Junction reference location method.
16	CS941428	Mutations uses a Splice Junction reference location method.
17	CS063248	Mutations uses a Splice Junction reference location method.
18	CS023593	Mutations uses a Splice Junction reference location method.
19	CS062038	Mutations uses a Splice Junction reference location method.
20	CS045210	Mutations uses a Splice Junction reference location method.
21	CS032401	Mutations uses a Splice Junction reference location method.
22	CS030978	Mutations uses a Splice Junction reference location method.
23	CS063249	Mutations uses a Splice Junction reference location method.
24	CS034305	Mutations uses a Splice Junction reference location method.

25	CS032402	Mutations uses a Splice Junction reference location method.
26	CS042532	Mutations uses a Splice Junction reference location method.
27	CS064371	Mutations uses a Splice Junction reference location method.
28	CS971623	Mutations uses a Splice Junction reference location method.
29	CS951357	Mutations uses a Splice Junction reference location method.
30	CS045208	Mutations uses a Splice Junction reference location method.
31	CS023468	Mutations uses a Splice Junction reference location method.
32	CS032685	Mutations uses a Splice Junction reference location method.
33	CS951358	Mutations uses a Splice Junction reference location method.
34	CS022865	Mutations uses a Splice Junction reference location method.
35	CS992995	Mutations uses a Splice Junction reference location method.
36	CS030979	Mutations uses a Splice Junction reference location method.
37	CS033831	Mutations uses a Splice Junction reference location method.
38	CS014607	Mutations uses a Splice Junction reference location method.
39	CS041876	Mutations uses a Splice Junction reference location method.
40	CS031769	Mutations uses a Splice Junction reference location method.
41	CS032403	Mutations uses a Splice Junction reference location method.
42	CS031770	Mutations uses a Splice Junction reference location method.
43	CS991329	Mutations uses a Splice Junction reference location method.
44	CS011028	Mutations uses a Splice Junction reference location method.
45	CS012668	Mutations uses a Splice Junction reference location method.
46	CS032404	Mutations uses a Splice Junction reference location method.
47	CS982090	Mutations uses a Splice Junction reference location method.
48	CS041877	Mutations uses a Splice Junction reference location method.
49	CS043934	Mutations uses a Splice Junction reference location method.
50	CS982089	Mutations uses a Splice Junction reference location method.
51	CS023200	Mutations uses a Splice Junction reference location method.
52	CS982091	Mutations uses a Splice Junction reference location method.
53	CS951359	Mutations uses a Splice Junction reference location method.
54	CS032405	Mutations uses a Splice Junction reference location method.
55	CS032686	Mutations uses a Splice Junction reference location method.
56	CS045209	Mutations uses a Splice Junction reference location method.
57	CS961493	Mutations uses a Splice Junction reference location method.

58	CS993672	Mutations uses a Splice Junction reference location method.
59	CS000586	Mutations uses a Splice Junction reference location method.
60	CS043785	Mutations uses a Splice Junction reference location method.
61	CS013998	Mutations uses a Splice Junction reference location method.
62	CS021727	Mutations uses a Splice Junction reference location method.
63	CS030109	Mutations uses a Splice Junction reference location method.
64	CS030110	Mutations uses a Splice Junction reference location method.
65	CS030111	Mutations uses a Splice Junction reference location method.
66	CS030112	Mutations uses a Splice Junction reference location method.
67	CS021726	Mutations uses a Splice Junction reference location method.
68	CS971624	Mutations uses a Splice Junction reference location method.
69	CS014608	Mutations uses a Splice Junction reference location method.
70	CS004527	Mutations uses a Splice Junction reference location method.
71	CS021728	Mutations uses a Splice Junction reference location method.
72	CS971625	Mutations uses a Splice Junction reference location method.
73	CS001825	Mutations uses a Splice Junction reference location method.
74	CS991331	Mutations uses a Splice Junction reference location method.
75	CS951360	Mutations uses a Splice Junction reference location method.
76	CS991330	Mutations uses a Splice Junction reference location method.
77	CS032687	Mutations uses a Splice Junction reference location method.
78	CS023469	Mutations uses a Splice Junction reference location method.
79	CS014125	Mutations uses a Splice Junction reference location method.
80	CS033832	Mutations uses a Splice Junction reference location method.
	Small Deletions	
81	CD991644	Mutations uses a Splice Junction reference location method.
82	CD994433	Mutations uses a Splice Junction reference location method.
	Small Insertions	
83	CI030168	Mutations uses a Splice Junction reference location method.
84	CI962219	Mutations uses a Splice Junction reference location method.
85	CI022582	Mutations uses a Splice Junction reference location method.

12.3 P3: DATA AMBIGUITY

12.3.1 P3A: PHENOTYPE AMBIGUITY

	Missense / nonsense	
1	CM041678	Breast and/or ovarian cancer ?
2	CM041679	Breast and/or ovarian cancer ?
3	CM041680	Breast and/or ovarian cancer ?
4	CM041681	Breast and/or ovarian cancer ?
5	CM041682	Breast and/or ovarian cancer ?
6	CM041683	Breast and/or ovarian cancer ?
7	CM041684	Breast and/or ovarian cancer ?
8	CM041685	Breast and/or ovarian cancer ?
9	CM041687	Breast and/or ovarian cancer ?
10	CM041686	Breast and/or ovarian cancer ?
11	CM041688	Breast and/or ovarian cancer ?
12	CM984014	Breast cancer ?
13	CM041689	Breast and/or ovarian cancer ?
14	CM041690	Breast and/or ovarian cancer ?
15	CM041691	Breast and/or ovarian cancer ?
16	CM051394	Breast and/or ovarian cancer ?
17	CM014322	Breast and/or ovarian cancer ?
18	CM984015	Breast cancer ?
19	CM062466	Breast and/or ovarian cancer ?
20	CM960172	Breast cancer ?
21	CM004236	Ovarian cancer ?
22	CM041692	Breast and/or ovarian cancer ?
23	CM041693	Breast and/or ovarian cancer ?
24	CM032861	Endometriosis, association with ?
25	CM940175	Breast cancer ?
26	CM034004	Breast cancer ?
27	CM051395	Breast and/or ovarian cancer ?
28	CM022767	Breast cancer ?
29	CM022768	Breast cancer ?
30	CM041695	Breast and/or ovarian cancer ?
31	CM041696	Breast and/or ovarian cancer ?

32	CM045041	Breast cancer ?
33	CM041697	Breast and/or ovarian cancer ?
34	CM960181	Breast cancer ?
35	CM041698	Breast and/or ovarian cancer ?
36	CM940179	Breast cancer ?
37	CM960183	Breast cancer ?
38	CM045533	Breast and/or ovarian cancer ?
39	CM053798	Ovarian cancer ?
40	CM014325	Breast and/or ovarian cancer ?
41	CM041700	Breast and/or ovarian cancer ?
42	CM041702	Breast and/or ovarian cancer ?
43	CM041701	Breast and/or ovarian cancer ?
44	CM041703	Breast and/or ovarian cancer ?
45	CM042678	Breast cancer ?
46	CM034006	Breast cancer ?
47	CM041704	Breast and/or ovarian cancer ?
48	CM041705	Breast and/or ovarian cancer ?
49	CM041707	Breast and/or ovarian cancer ?
50	CM950153	Breast cancer ?
51	CM041708	Breast and/or ovarian cancer ?
52	CM041709	Breast and/or ovarian cancer ?
53	CM041710	Breast and/or ovarian cancer ?
54	CM041711	Breast and/or ovarian cancer ?
55	CM984051	Breast cancer ?
56	CM041712	Breast and/or ovarian cancer ?
57	CM041714	Breast and/or ovarian cancer ?
58	CM034008	Breast and/or ovarian cancer ?
59	CM041715	Breast and/or ovarian cancer ?
60	CM041716	Breast and/or ovarian cancer ?
61	CM041717	Breast and/or ovarian cancer ?
62	CM041718	Breast and/or ovarian cancer ?
63	CM041719	Breast and/or ovarian cancer ?
64	CM984016	Breast cancer ?

65	CM041720	Breast and/or ovarian cancer ?
66	CM041721	Breast and/or ovarian cancer ?
67	CM041723	Breast and/or ovarian cancer ?
68	CM062465	Breast and/or ovarian cancer ?
69	CM041725	Breast and/or ovarian cancer ?
70	CM032862	Breast and/or ovarian cancer ?
71	CM041726	Breast and/or ovarian cancer ?
72	CM041727	Breast and/or ovarian cancer ?
73	CM041728	Breast and/or ovarian cancer ?
	Splicing	
74	CS961492	Breast cancer ?
75	CS012667	Breast and/or ovarian cancer ?
76	CS045210	Breast cancer ?
77	CS033831	Breast cancer ?
78	CS041876	Breast and/or ovarian cancer ?
79	CS012668	Breast and/or ovarian cancer ?
80	CS041877	Breast and/or ovarian cancer ?
81	CS045209	Breast cancer ?
82	CS030109	Breast cancer ?
83	CS030110	Breast cancer ?
84	CS030111	Breast cancer ?
85	CS030112	Breast cancer ?
86	CS001825	Ovarian cancer ?
87	CS991331	Breast cancer ?
88	CS991330	Breast cancer ?
89	CS033832	Breast cancer ?
	Small Deletions	
90	CD994433	Ovarian cancer ?
91	CD041903	Breast and/or ovarian cancer ?
	Small Insertions	
92	CI030168	Breast cancer ?
93	CI962219	Breast cancer ?
94	CI022582	Breast and/or ovarian cancer ?

12.4 P4: INCOMPLETE DATA

12.4.1 P4A: DATA ENTRY LACKING

	Splicing	
1	-	At position IVS22+67 a T>C mutation happens according to [*Panguluri, 1999*]. This mutation is absent in the HGMD database.
2	-	At position IVS22+8 a T>A mutation happens according to [*Panguluri, 1999*]. This mutation is absent in the HGMD database.
3	-	[*Langston,1996*] mentions a mutation at position IVS4+49, however not indicating whether this is a donor, or acceptor referenced location. Also, the paper does not specify a (possible) phenotype. This mutation is absent in the HGMD database.

12.5 INADEQUATE DATA

12.5.1 P5A: INADEQUATE DATA

	Splicing	
1	CS961492	HGMD indicates a breast cancer phenotype, however this exact result of the mutation is not mentioned in the reference paper. Instead the paper mentions a relation to prostate cancer in men.

12.6 P6: INCONSISTENT DATA

12.6.1 P6A: NON-EXISTING INTRON

	Splicing	
1	CS063247	Mutation location referenced to intron 4, which is not indicated in the 'Splice Junctions Overview'.
2	CS011027	Mutation location referenced to intron 4, which is not indicated in the 'Splice Junctions Overview'.

12.6.2 P6B: INCONSISTENT NUCLEOTIDE REFERENCES

	Splicing	
1	CS012667	HGMD indicates a G>A mutation here, however at this location (ds,IVS2+3) in the reference DNA sequence appears an 'A'.
2	CS045210	HGMD indicates a T>C mutation here, however at this location (as,IVS7-34) in the reference DNA sequence appears a 'C'.
3	CS991331	HGMD indicates a T>C mutation here, however at this location (ds,IVS22+7) in the reference DNA sequence appears an 'A'.
4	CS951360	HGMD indicates a A>T mutation here, however at this location (ds,IVS22+32) in the reference DNA sequence appears a 'G'.
5	CS991330	HGMD indicates a C>A mutation here, however at this location (ds,IVS22+78) in the reference DNA sequence appears a 'T'.
6	CD063448	The sample sequence given by HGMD (`GCTTCCCTGCTTCCAACACTTGTT`), does not match the sequence at that location in the DNA sequence (`AGATTTCTCTCCATATCTGATTTC`).

7	CI063643	The sample sequence given by HGMD (AATGCTGAAGGACCCCAAAGT), does not match the sequence at that location in the DNA sequence (AATGCTGAAGACCCCAAAGAT).

12.6.3 P6C: INCONSISTENTLY LOCATING NUCLEOTIDES

	Small Insertions	
1	CI030168	Mutation location is not indicated by giving a codon number + offset, as is usual for the rest of this type of mutations. In this case the mutation has a different way of locating, by referring to a splice junctions + offset (IVS20+21).
2	CI962219	Mutation location is not indicated by giving a codon number + offset, as is usual for the rest of this type of mutations. In this case the mutation has a different way of locating, by referring to a splice junctions + offset (IVS20+48).
3	CI022582	Mutation location is not indicated by giving a codon number + offset, as is usual for the rest of this type of mutations. In this case the mutation has a different way of locating, by referring to a splice junctions + offset (IVS20+64).
4	CI055737	Instead of giving you a fixed nucleotide position within the cDNA, as is done for the rest of the mutations within this type, in this case '199+2' is given. However, due to the fact the position in this type of mutations is provided redundantly, both by using a codon+offset notation ánd a fixed cDNA nucleotide this does not pose a severe difficulty.
5	CI004793	In this case, instead of a fixed nucleotide position within the cDNA, only the codon+offset location is given.
6	CI023719	In this case, instead of a fixed nucleotide position within the cDNA, only the codon+offset location is given.
7	CD991644	Mutation location indicated as being 'non-coding', an indication of the mutations location is given in a mouse-over tag though.
8	CD994433	Mutation location indicated as being 'non-coding', an indication of the mutations location is given in a mouse-over tag though.

13 APPENDIX 3: SCRIPT SOURCE

13.1 MISSENSE / NONSENSE

```
<?php
// FUNCTIONS
include("lib/functions.php");

// MYSQL DATABASE CONNECTION
include("includes/mysql_config.php");
$db = mysql_pconnect($dbhost,$dbuser,$dbpass);
mysql_select_db("$dbname",$db) or die('Cannot select database');

$gen="BRCA1";
$allele='1';
$specialization_effect ='M';
$specialization_mutant = 'M';
$specialization_localization = 'G';
$ID_databank = 'HGMD';
$description = '';
$tipo="ID";
$ins_repetition = '';
$db = "(DESCRIPTION=(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = 158.42.186.17)(PORT = 1521)))
(CONNECT_DATA=(SID=genoma)))";
$num = 10999;
#$db= "";

# start Main program
echo "<h3> Proceso de Carga de ".$gen." MISSENSE/NOSENSE </h3><p><hr><p>";
```

```
$join=""; // NCBI CDS data

$misense = ""; // HGMD mutational entries

$query = "SELECT sequence FROM sequence WHERE name='cdna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$cdna = remove_space($row->sequence);

$query = "SELECT sequence FROM sequence WHERE name='dna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$dna = remove_space($row->sequence);

//echo "<pre>".substr($cdna,0,60)."</pre>".strlen($cdna);
//echo  "<pre>".substr($dna,132543,6)."</pre><br>".strlen($dna);
echo "<pre>".$cdna."</pre>";//.strlen($cdna);
//echo  "<pre>".$dna."</pre><br>".strlen($dna);

$micDNA="";
$rangos=explode(",",$join);

for($i=1;$i<count($rangos);$i++) {

        $limites_exon[$i]=explode("..",$rangos[$i]);
        $longitud[$i]=$limites_exon[$i][1]-$limites_exon[$i][0]+1;
        $micDNA.=substr($dna,$limites_exon[$i][0]-1,$longitud[$i]);
}

//echo "<pre>".print_r($micDNA)."</pre>";
echo "<pre>".$micDNA."</pre>";//.strlen($micDNA);
echo strcmp($micDNA,$cdna);
for($i=0;$i<strlen($micDNA);$i++) {
        if($micDNA[$i]!=$cdna[$i]) echo "<br>".$i;
}

//busqueda de los codones
$lineas2=explode("\r",$misense);
for($i=0;$i<count($lineas2);$i++) {
        $r_db=explode("\t",$lineas2[$i]);
        $ref_db[$i]=$r_db[0];
}

$lineas=explode("\n",$misense);
/*echo "<pre>";
print_r($lineas) ;
echo "</pre>";*/
$totales_mutaciones=count($lineas);
$mut_correct=0;
$mut_incorrect=0;
for($i=0;$i<count($lineas);$i++) {

        $num += 1; // mutation ID counter

        $campos=explode("\t",$lineas[$i]);//en las coordenadas 1 y 3 tenemos los cambios y el numero
de codon
        echo("<h3>REF: $ref_db[$i] Busqueda del codon $campos[3] reemplazo $campos[1]</h3>");
        //echo "<pre>";
        $long=2;
        $b=busqueda($campos[3],$longitud,$limites_exon,$dna,$cdna);
        $datos=explode("-",$campos[1]);
        echo "<br>".$datos[0]."<br>";
        echo substr($dna,$b[0]-1,1).substr($dna,$b[1]-1,1).substr($dna,$b[2]-1,1)."<br>";
        //echo substr($cdna,$c-1,3)."<br>";
        if(strlen($datos[0])==3) {
                $pp=$b[1];
                $reemplazo=$datos[1][1];
                $codon_valido=$datos[0];

        }
        else {
                if ($datos[0][0]=='a' || $datos[0][0]=='g' ||$datos[0][0]=='t' ||$datos[0][0]=='c') {
                        $v="coincide";
```

```
                        $reemplazo=$datos[1][0];
                        $pp=$b[0];
                        $codon_valido=$datos[0][1].$datos[0][2].$datos[0][3];
                }
                else {
                        $v="NO";
                        $pp=$b[2];
                        $reemplazo=$datos[1][2];
                        $codon_valido=$datos[0][0].$datos[0][1].$datos[0][2];
                }
        //echo "<p> $v <p>";
        }
        if($codon_valido==substr($dna,$b[0]-1,1).substr($dna,$b[1]-1,1).substr($dna,$b[2]-1,1)) {
                $mut_correct++;
        }
        else {
                $incorrect[$mut_incorrect]=$campos;
                $incorrect[$mut_incorrect][0]=$ref_db[$i];
                $mut_incorrect++;
        }

        $num_bases = count($reemplazo);
    $phenotype=$campos[4];
        echo "<br><b>Variation Table</b><br>";
        echo "ID_variation: $num<br>";
        echo "ID_gene_RT: $gen<br>";
        echo "ID_allele_num_RT: $allele<br>";
        echo "Specialization_effect: $specialization_effect<br>";
        echo "Specialization_mutant: $specialization_mutant<br>";
        echo "Specialization_localization: $specialization_localization<br>";
        echo "ID_databank: $ID_databank<br>";
        echo "ID_variation_DB: $ref_db[$i]<br>";
        echo "Description: $description<br><br>";

        echo "<b>Precise Table</b><br>";
        echo "ID_variation: $num<br>";
        echo "Position: $pp<br>";
        echo "Type: $tipo<br>";
        echo "Ins_sequence: $reemplazo<br>";
        echo "Ins_repetition: $ins_repetition<br>";
        echo "Num_bases: $num_bases<br>";
        echo "Pheotipo: $phenotype<br><br>";

        //Aqui se debe hacer la insercion en la base de datos
        //insert_data_variation($c1,$num,$gen,$allele ,$specialization_effect ,
$specialization_mutant ,$specialization_localization ,$ID_databank ," $ref_db[$i]",
$description );   // Insert a row using c1
        //commit($c1);
        //insert_data_precise($c1,$num,$pp,$tipo ,$reemplazo ,$ins_repetition ,$num_bases,
$phenotype);   // Insert a row using c1

}
//commit($c1);
?>
<TABLE style="border:1px solid green;"><tr><td>Mutaciones totales:</td><td><? echo
$totales_mutaciones; ?></td></tr><tr><td>Correctas</td><td><? echo $mut_correct; ?></td></
tr><tr><td>Incorrectas</td><td><? echo $mut_incorrect; ?></td></tr></TABLE>
<TABLE style="border:1px solid green;">
<?
for($i=0;$i<count($incorrect);$i++){
?>
<tr>
<td><? echo $incorrect[$i][0]; ?></td>
<td><? echo $incorrect[$i][1]; ?></td>
<td><? echo $incorrect[$i][2]; ?></td>
<td><? echo $incorrect[$i][3]; ?></td>
<td><? echo $incorrect[$i][4]; ?></td>
<td> <? echo $incorrect[$i][5]; ?></td></tr>
<?
}
echo "</TABLE>";
?>
```

13.2 SPLICING

```
<?php
```

```
// MYSQL DATABASE CONNECTION
include("includes/mysql_config.php");
$db = mysql_pconnect($dbhost,$dbuser,$dbpass);
mysql_select_db("$dbname",$db) or die('Cannot select database');

// FUNCTIONS
include("lib/functions.php");
$gen="BRCA1";
$allele='1';
$specialization_effect ='M';
$specialization_mutant = 'S';
$specialization_localization = 'G';
$ID_databank = 'HGMD';
$description = '';
$tipo="ID";
$ins_sequence = '';
$ins_repetition = '';
$db = "(DESCRIPTION=(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = 158.42.186.17)(PORT = 1521)))
(CONNECT_DATA=(SID=genoma)))";
$num = 14999;
#$db= "";

function elimina_ceros($c) {

        while($c[0]=="0") {
                $c=substr($c,1,strlen($c)-1);
        }

        return $c;
}

$splice_junctions=""; // HGMD splice junctions overview

function convertir_exon() {
        $exones=explode(" E",$splice_junctions);

        //echo "<pre>".print_r($exones)."</pre>";
        $j=0;
        for($i=1;$i<count($exones);$i++){
                $n_exon=explode(" ",$exones[$i]);
                $n_exon[0]=elimina_ceros($n_exon[0]);

                //echo $n_exon[0].'<br>';

                $j++;

                $n_exones[$n_exon[0]]=$j;

                if($n_exon[0] == '1a' || $n_exon[0] == '1b' ) {
                        $j = 1;
                        $n_exones[$n_exon[0]]=1;
                }

                //echo "EN la coordenada $n_exon[0] hay ".$n_exones[$n_exon[0]]."<br>";
        }
        return $n_exones;
}

function remove_zeros($input) {

        $new = "";
        for($i=0;$i<strlen($input);$i++) {

                //echo $input[$i];

                if($input[$i]=="0" && $i<4) {
                        $new .= "";
                }
                else {
                        $new .= $input[$i];
                }
        }

        return $new;
}

function strip_sequence($sequence) {
```

```
        //echo $sequence."<br>";

        $flipper = 0;
        $fullNotation = "";
        for($i=0;$i<strlen($sequence);$i++) {

                // replace short notation for repeating sequences with complete sequence
                if($sequence[$i]=="(") {
                        $bracket_start = $i+1;
                        $flipper = 1;
                }

                if($sequence[$i]==")" && $flipper == 1) {
                        $bracket_end = $i+1;

                        $num_digits = 0;
                        for($z=$i+1;$z<strlen($sequence);$z++) {

                                if(is_numeric($sequence[$z])) {
                                        $num_digits++;
                                }
                        }

                        $frequencyRepeat = substr($sequence, $bracket_end, $num_digits);
                        $repeatedSequence = substr($sequence, $bracket_start, ($bracket_end-
$bracket_start)-1);

                        $constructedSequence = str_repeat($repeatedSequence, $frequencyRepeat-1);

                        $flipper = 0;

                        $fullNotation .= $constructedSequence;
                }
                else {
                        $fullNotation .= $sequence[$i];
                }
        }

        $validChars = array("1" => "A", "2" => "C", "3" => "T", "4" => "G", "5" => "a", "6" => "c",
"7" => "t", "8" => "g");
        $output = "";
        for($i=0;$i<strlen($fullNotation);$i++) {

                if(in_array($fullNotation[$i], $validChars)) {
                        $output .= $fullNotation[$i];
                }
                else {
                        $output .= "";
                }

        }

        //echo $output."<br>";
        return $output;
}

# start Main program
echo "<h3> Proceso de Carga de ".$gen." - SPLICING</h3><p><hr><p>";

$splicing=""; // HGMD mutational entries

$query = "SELECT sequence FROM sequence WHERE name='cdna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$cdna = remove_space($row->sequence);

$query = "SELECT sequence FROM sequence WHERE name='dna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$dna = remove_space($row->sequence);

$basesUpperCase = array(1 => "65", 2 => "67", 3 => "71", 4 => "84");
$basesLowerCase = array(1 => "97", 2 => "99", 3 => "103", 4 => "116");;
```

```
$lineas2=explode("\r",$splicing);
for($i=0;$i<count($lineas2);$i++) {
	$r_db=explode("\t",$lineas2[$i]);
	$ref_db[$i]=$r_db[0];
}
$lineas=explode("\n", $splicing);
$totales_mutaciones=count($lineas);
$mut_correct=0;
$mut_incorrect=0;
//$found=0;
for($k=0;$k<count($lineas);$k++) {

	$num += 1; // mutation ID counter

	$campos=explode("\t",$lineas[$k]);//en las coordenadas 1 y 3 tenemos los cambios y el numero
de codon

	echo "<h3>$k REF: $campos[0] Busqueda del IVS $campos[1] $campos[2] $campos[3] reemplazo
$campos[4]</h3>";

	$mutIVS = $campos[1];
	$mutOffset = $campos[3];

	if($campos[2]=="ds") { //donor
		$identifier = "E".$mutIVS;
	}
	else { //acceptor
		$identifier = "I".$mutIVS;
	}


	$splice_lines = explode("\n", $splice_junctions);
	for($i=0;$i<count($splice_lines);$i++) {

		$mutPos = "";

		$splice_fields = explode(" ", $splice_lines[$i]);

		$splice_fields[0] = remove_zeros($splice_fields[0]);
		$splice_fields[1] = remove_zeros($splice_fields[1]);

		if($splice_fields[0] == $identifier) {

			$splice_fields[2] = strip_sequence($splice_fields[2]);

			echo $splice_fields[2];

			if($campos[2]=="ds") { // donor

				for($j=0;$j<strlen($splice_fields[2]);$j++) {
					if(in_array(ord($splice_fields[2][$j]), $basesLowerCase) ) {

						if($mutPos == "") {
							if($mutOffset>0) {
								$mutPos = ($j+$mutOffset)-1;
							}
							else {
								$mutPos = $j+$mutOffset;
							}
						}
					}
				}
			}
			if($campos[2]=="as") { // acceptor

				for($j=0;$j<strlen($splice_fields[2]);$j++) {
					if(in_array(ord($splice_fields[2][$j]), $basesUpperCase) ) {

						if($mutPos == "") {$mutPos = $j+$mutOffset;}
					}
				}
			}

			//echo $mutPos;

			for($q=0;$q<strlen($dna);$q++) {
```

```
                        if(substr($dna, $q, strlen($splice_fields[2])) == strtoupper
($splice_fields[2])) {
                                echo " FOUND! <br>";
                                $found++;
                                $pp = ($q+$mutPos);
                                $pp_real = $pp+1;

                                $reemplazo = $campos[4][2];
                                $num_bases = count($reemplazo);
                                $phenotype=$campos[5];

                                //echo "<br>".$identifier."<br>";
                                echo "<br>Posicion: $pp_real <br>";
                                //echo "<br>Reemplazo: ".$reemplazo."<br>";

                                echo "<br><b>Variation Table</b><br>";
                                echo "ID_variation: $num<br>";
                                echo "ID_gene_RT: $gen<br>";
                                echo "ID_allele_num_RT: $allele<br>";
                                echo "Specialization_effect: $specialization_effect<br>";
                                echo "Specialization_mutant: $specialization_mutant<br>";
                                echo "Specialization_localization:
$specialization_localization<br>";
                                echo "ID_databank: $ID_databank<br>";
                                echo "ID_variation_DB: $campos[0]<br>";
                                echo "Description: $description<br><br>";

                                echo "<b>Precise Table</b><br>";
                                echo "ID_variation: $num<br>";
                                echo "Position: $pp_real<br>";
                                echo "Type: $tipo<br>";
                                echo "Ins_sequence: $reemplazo<br>";
                                echo "Ins_repetition: $ins_repetition<br>";
                                echo "Num_bases: $num_bases<br>";
                                echo "Pheotipo: $phenotype<br><br>";
                                //Aqui se debe hacer la insercion en la base de datos
                                //insert_data_variation($c1,$num,$gen,$allele ,
$specialization_effect ,$specialization_mutant ,$specialization_localization ,$ID_databank ,"$campos
[0]", $description );   // Insert a row using c1
                                //commit($c1);
                                //insert_data_precise($c1,$num,$pp_real,$tipo ,$reemplazo ,
$ins_repetition ,$num_bases,$phenotype);   // Insert a row using c1

                        }
                }
                //commit($c1);
                if(!isset($pp)) {
                        echo "<br><br>The HGMD splice junctions overview DNA sequence, is not
present in the DNA sequence";
                }

                $constrNucleotide = substr($dna, $pp, 1);

                $mismatch = 'FALSE';
                if(strtoupper($constrNucleotide) != $campos[4][0]) {

                        $mismatch = 'TRUE';
                        $incorrect[$mut_incorrect]=$campos;
                        $incorrect[$mut_incorrect][0]=$campos[0];
                        $mut_incorrect++;

                        echo "<br>";
                        echo "Nucleotide mismatch $constrNucleotide";
                        echo "<br>";
                }
                else {
                        $mut_correct++;
                }
            }
        }
}
?>
<TABLE style="border:1px solid green;"><tr><td>Mutaciones totales:</td><td><? echo
$totales_mutaciones; ?></td></tr><tr><td>Correctas</td><td><? echo $mut_correct; ?></td></
tr><tr><td>Incorrectas</td><td><? echo $mut_incorrect; ?></td></tr><tr><td>Found</td><td><? echo
$found; ?></td></tr></TABLE>
<TABLE style="border:1px solid green;">
```

```
<?
for($i=0;$i<count($incorrect);$i++){
?>
<tr>
<td><? echo $incorrect[$i][0]; ?></td>
<td><? echo "IVS ".$incorrect[$i][1]; ?></td>
<td><? echo $incorrect[$i][2]; ?></td>
<td><? echo $incorrect[$i][3]; ?></td>
<td><? echo $incorrect[$i][4]; ?></td>
<td><? echo $incorrect[$i][5]; ?></td></tr>
<?
}
echo "</TABLE>";
print_r($n_exones);
?>
```

13.3 SMALL INSERTIONS

```
<?php
// MYSQL DATABASE CONNECTION
include("includes/mysql_config.php");
$db = mysql_pconnect($dbhost,$dbuser,$dbpass);
mysql_select_db("$dbname",$db) or die('Cannot select database');

$gen="BRCA1";
$allele='1';
$specialization_effect ='M';
$specialization_mutant = 'O';
$specialization_localization = 'G';
$ID_databank = 'HGMD';
$description = '';
$tipo='IS';
$ins_sequence = '';
$ins_repetition = '';
$db = "(DESCRIPTION=(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = 158.42.186.17)(PORT = 1521)))
(CONNECT_DATA=(SID=genoma)))";
$num = 13999;
#$db= "";

# start Main program
echo "<h3> Proceso de Carga de ".$gen." - SMALL INSERTIONS </h3><p><hr><p>";


$join=""; // NCBI CDS data

$small_insertions = ""; // HGMD mutational entries

// FUNCTIONS
include("lib/functions.php");

$query = "SELECT sequence FROM sequence WHERE name='cdna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$cdna = remove_space($row->sequence);

$query = "SELECT sequence FROM sequence WHERE name='dna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$dna = remove_space($row->sequence);

//echo "<pre>".substr($cdna,0,60)."</pre>".strlen($cdna);
//echo  "<pre>".substr($dna,132543,6)."</pre><br>".strlen($dna);
echo "<pre>".$cdna."</pre>";//.strlen($cdna);
//echo  "<pre>".$dna."</pre><br>".strlen($dna);

$micDNA="";
$rangos=explode(",",$join);
for($i=1;$i<count($rangos);$i++) {
        $limites_exon[$i]=explode("..",$rangos[$i]);
        $longitud[$i]=$limites_exon[$i][1]-$limites_exon[$i][0]+1;
        $micDNA.=substr($dna,$limites_exon[$i][0]-1,$longitud[$i]);
}

//echo "<pre>".print_r($micDNA)."</pre>";
```

```
echo "<pre>".$micDNA."</pre>";//.strlen($micDNA);
echo strcmp($micDNA,$cdna);
for($i=0;$i<strlen($micDNA);$i++) {
	if($micDNA[$i]!=$cdna[$i]) {
		echo "<br>".$i;
	}
}

//busqueda de los codones
$lineas2=explode("\r",$small_insertions);
for($i=0;$i<count($lineas2);$i++) {
	$r_db=explode("\t",$lineas2[$i]);
	$ref_db[$i]=$r_db[0];
}
$lineas=explode("\n",$small_insertions);

$totales_mutaciones=count($lineas);
$mut_correct=0;
$mut_incorrect=0;
for($i=0;$i<count($lineas);$i++) {

	$num += 1; // mutation ID counter

	$campos=explode("\t",$lineas[$i]);//en las coordenadas 1 y 3 tenemos los cambios y el numero
de codon

	echo("<h3>REF: $ref_db[$i] Busqueda del codon $campos[2] cadena a añadir $campos[1]</h3>");

	$long=2;
	$b=busqueda($campos[2],$longitud,$limites_exon,$dna,$cdna);
	$sin_cad_exones=explode("_",$campos[1]);
	if(count($sin_cad_exones)>1) $campos[1]=$sin_cad_exones[0].$sin_cad_exones[2];
	echo("<br /><b>Busqueda del codon $campos[2] cadena a eliminar $campos[1]</b><br /><br />");
	$datos=explode("^",$campos[1]);
	//echo "<br>".$datos[0]."<br>";
	//echo "<br>".$datos[1]."<br>";
	$lista=$campos[1];
	$pos_ini=0;
	$insertados=0;
	$cadena_insertada="";

	for($k=0;$k<strlen(($lista));$k++) {
		if($lista[$k]=="a" ||$lista[$k]=="c" ||$lista[$k]=="t" ||$lista[$k]=="g"){
			$cadena_insertada.=strtoupper($lista[$k]);
			if($pos_ini==0) {
				$pos_ini=$k+1;
			}
			$insertados++;
		}
	}

	$num_bases = 0;
	$clean_list = str_replace("^", "", $lista);
	for($k=0;$k<strlen(($clean_list));$k++){
		if(ord($clean_list[$k])=="97" || ord($clean_list[$k])=="99" || ord($clean_list[$k])
=="103" || ord($clean_list[$k])=="116"){
			if($num_bases == 0) {$start_deletion = $k+1;}
			$num_bases += 1;
		}
	}

	//echo "<br> cadena ".$cadena_insertada."<br>";
	//echo "<br> insertados ".$insertados."<br>";
	//echo "<br> pos ini ".$pos_ini."<br>";
	$ref_longitud_total=strlen($campos[1])-1;//por el ^ que no cuenta hay que restar 1
	$ref_longitud_antes_gorro=strlen($datos[0]);
	$ref_longitud_despues_gorro=strlen($campos[1])-1-$ref_longitud_antes_gorro;
	$mut_dna_loc = ($start_deletion - ($ref_longitud_antes_gorro+1)) + $b[0];
	//echo "<br>ref long ".$ref_longitud_total."<br>";
	//echo "<br>ref long antes ".$ref_longitud_antes_gorro."<br>";
	//echo "<br>ref long despues ".$ref_longitud_despues_gorro."<br>";
	$nucleotide_pos = $b[0];

	echo 'Reference codon nucleotidal position: '.$nucleotide_pos.'<br />';
	//echo 'Nucleotidal location of the mutation in reference to DNA: '.$mut_dna_loc.'<br />';
	//echo 'Number of inserted nucleotides: '.$num_bases.'<br />';
```

```
        //echo substr($dna,$b[0]-1,1).substr($dna,$b[1]-1,1).substr($dna,$b[2]-1,1)."<br>";
        $codon_valido=substr($datos[1],0,3);
        //comprobacion de validez de los resultados
        //construimos la cadena del adn que nos dan a ver si coincide completa
        //SI LOS EXONES O LOS INTRONES FUERAN MUY CORTOS ESTO NO FUNCIONARIA
        $cadena_en_adn="";
        for($k=$ref_longitud_antes_gorro;$k>=1;$k--) {
                $cadena_en_adn.=substr($dna,$b[0]-1-$k,1);
        }

        $cadena_en_adn.=substr($dna,$b[0]-1,1).substr($dna,$b[1]-1,1).substr($dna,$b[2]-1,1);//debe
coincidir con $campos[1]
        $campos[1]=$datos[0].$datos[1];
        for($k=1;$k<$ref_longitud_despues_gorro-2;$k++) {
                $cadena_en_adn.=substr($dna,$b[2]-1+$k,1);
        }

        //le hacemos la insercionpara dejarla igual
        $cadena_con_inserciones="";
        for($k=0;$k<strlen($cadena_en_adn);$k++) {
                if ($k!=$pos_ini-2) {
                        $cadena_con_inserciones.=$cadena_en_adn[$k];
                }
                else {
                        $cadena_con_inserciones.=$cadena_insertada.$cadena_en_adn[$k];
                }
        }

        //echo "<br>".$cadena_en_adn." cadena construida <br>";
        //echo "<br>".substr($cadena_con_inserciones,0,strlen($campos[1]))." cadena construida
<br>";
        //echo "".strtoupper($campos[1])." cadena valida <br>";
        if(substr($cadena_con_inserciones,0,strlen($campos[1]))==strtoupper($campos[1])) {
                $mut_correct++;
        }
        else {
                $mut_incorrect++;
                $incorrect[$mut_incorrect]=$campos;
                $incorrect[$mut_incorrect][0]=$ref_db[$i];
                echo "INCORRECT<br>";
        }

        $pp = $mut_dna_loc;
        $reemplazo = $cadena_insertada;
        $phenotype=$campos[4];
        echo "<br><b>Variation Table</b><br>";
        echo "ID_variation: $num<br>";
        echo "ID_gene_RT: $gen<br>";
        echo "ID_allele_num_RT: $allele<br>";
        echo "Specialization_effect: $specialization_effect<br>";
        echo "Specialization_mutant: $specialization_mutant<br>";
        echo "Specialization_localization: $specialization_localization<br>";
        echo "ID_databank: $ID_databank<br>";
        echo "ID_variation_DB:  $ref_db[$i]<br>";
        echo "Description: $description<br><br>";

        echo "<b>Precise Table</b><br>";
        echo "ID_variation: $num<br>";
        echo "Position: $pp<br>";
        echo "Type: $tipo<br>";
        echo "Ins_sequence: $reemplazo<br>";
        echo "Ins_repetition: $ins_repetition<br>";
        echo "Num_bases: $num_bases<br>";
        echo "Pheotipo: $phenotype<br><br>";

        //Aqui se debe hacer la insercion en la base de datos
        //insert_data_variation($c1,$num,$gen,$allele ,$specialization_effect ,
$specialization_mutant ,$specialization_localization ,$ID_databank ," $ref_db[$i]",
$description );   // Insert a row using c1
        //commit($c1);
        //insert_data_precise($c1,$num,$pp,$tipo ,$reemplazo ,$ins_repetition ,$num_bases,
$phenotype);   // Insert a row using c1
        //commit($c1);
}
//commit($c1);
?>
```

```
<TABLE style="border:1px solid green;"><tr><td>Mutaciones totales:</td><td><? echo
$totales_mutaciones; ?></td></tr><tr><td>Correctas</td><td><? echo $mut_correct; ?></td></
tr><tr><td>Incorrectas</td><td><? echo $mut_incorrect; ?></td></tr></TABLE>
<TABLE style="border:1px solid green;">
<?
for($i=1;$i<=count($incorrect);$i++){
?>
<tr>
<td><? echo $incorrect[$i][0]; ?></td>
<td><? echo $incorrect[$i][1]; ?></td>
<td><? echo $incorrect[$i][2]; ?></td>
<td><? echo $incorrect[$i][3]; ?></td>
<td><? echo $incorrect[$i][4]; ?></td>
<td> <? echo $incorrect[$i][5]; ?></td></tr>
<?
}
echo "</TABLE>";
?>
```

13.4 SMALL DELETIONS

```
<?php
// MYSQL DATABASE CONNECTION
include("includes/mysql_config.php");
$db = mysql_pconnect($dbhost,$dbuser,$dbpass);
mysql_select_db("$dbname",$db) or die('Cannot select database');

$gen="BRCA1";
$allele='1';
$specialization_effect ='M';
$specialization_mutant = 'O';
$specialization_localization = 'G';
$ID_databank = 'HGMD';
$description = '';
$tipo="DE";
$ins_sequence = '';
$ins_repetition = '';
$db = "(DESCRIPTION=(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = 158.42.186.17)(PORT = 1521)))
(CONNECT_DATA=(SID=genoma)))";
$num = 11999;
#$db= "";

# start Main program
echo "<h3> Proceso de Carga de ".$gen." - SMALL DELETIONS </h3><p><hr><p>";


$join=""; // NCBI CDS data

$small_deletions = ""; // HGMD mutational entries

// FUNCTIONS
include("lib/functions.php");

$query = "SELECT sequence FROM sequence WHERE name='cdna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$cdna = remove_space($row->sequence);

$query = "SELECT sequence FROM sequence WHERE name='dna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$dna = remove_space($row->sequence);

//echo "<pre>".substr($cdna,0,60)."</pre>".strlen($cdna);
//echo  "<pre>".substr($dna,132543,6)."</pre><br>".strlen($dna);
echo "<pre>".$cdna."</pre>";//.strlen($cdna);
//echo  "<pre>".$dna."</pre><br>".strlen($dna);

$micDNA="";
$rangos=explode(",",$join);
for($i=1;$i<count($rangos);$i++) {

        $limites_exon[$i]=explode("..",$rangos[$i]);
        $longitud[$i]=$limites_exon[$i][1]-$limites_exon[$i][0]+1;
```

```
        $micDNA.=substr($dna,$limites_exon[$i][0]-1,$longitud[$i]);
}

//echo "<pre>".print_r($micDNA)."</pre>";
echo "<pre>".$micDNA."</pre>";//.strlen($micDNA);
echo strcmp($micDNA,$cdna);
for($i=0;$i<strlen($micDNA);$i++) {

        if($micDNA[$i]!=$cdna[$i]) {
                echo "<br>".$i;
        }
}

//busqueda de los codones
$lineas2=explode("\r",$small_deletions);
for($i=0;$i<count($lineas2);$i++) {
        $r_db=explode("\t",$lineas2[$i]);
        $ref_db[$i]=$r_db[0];
}
$lineas=explode("\n", $small_deletions);

/*echo "<pre>";
print_r($lineas) ;
echo"</pre>";*/
$totales_mutaciones=count($lineas);
$mut_correct=0;
$mut_incorrect=0;
for($i=0;$i<count($lineas);$i++) {

        $num += 1; // mutation ID counter

        $campos=explode("\t",$lineas[$i]);//en las coordenadas 1 y 3 tenemos los cambios y el numero
de codon

        //insert_data_variation($c1,1000+$i,$gen,'1' ,'M' ,'O' ,'G' ,'1' ,"$campos[0]"," " );   //
Insert a row using c1
        //commit($c1);

        echo("<h3>REF: $ref_db[$i] Busqueda del codon $campos[2] cadena a eliminar $campos[1]</
h3>");
        //echo "<pre>";
        $long=2;
        $b=busqueda($campos[2],$longitud,$limites_exon,$dna,$cdna);
        $sin_cad_exones=explode("_",$campos[1]);
        if(count($sin_cad_exones)>1) {
                $campos[1]=$sin_cad_exones[0].$sin_cad_exones[2];
        }

        echo("<br /><b>Busqueda del codon $campos[2] cadena a eliminar $campos[1]</b><br /><br />");
        $datos=explode("^",$campos[1]);
        //echo "<br>".$datos[0]."<br>";
        $lista=$campos[1];

        $pos_ini=0;

        $num_bases = 0;
        $clean_list = str_replace("^", "", $lista);
        for($k=0;$k<strlen(($clean_list));$k++){
                if(ord($clean_list[$k])=="97" || ord($clean_list[$k])=="99" || ord($clean_list[$k])
=="103" || ord($clean_list[$k])=="116") {
                        if($num_bases == 0) {
                                $start_deletion = $k+1;
                        }
                        $num_bases += 1;
                }
        }

        $ref_longitud_total=strlen($campos[1])-1;//por el ^ que no cuenta hay que restar 1
        $ref_longitud_antes_gorro=strlen($datos[0]);
        $ref_longitud_despues_gorro=strlen($campos[1])-1-$ref_longitud_antes_gorro;
        $mut_dna_loc = ($start_deletion - ($ref_longitud_antes_gorro+1)) + $b[0];
        $nucleotide_pos = $b[0];

        //echo 'Reference codon nucleotidal position: '.$nucleotide_pos.'<br />';
        //echo 'Nucleotidal location of the mutation in reference to DNA: '.$mut_dna_loc.'<br />';
        //echo 'Number of deleted nucleotides: '.$num_bases.'<br />';
        //echo 'Mutation internal ID: '.$num.'<br />';
```

```
        //echo substr($dna,$b[0]-1,1).substr($dna,$b[1]-1,1).substr($dna,$b[2]-1,1)."<br>";

        $codon_valido=substr($datos[1],0,3);
        //comprobacion de validez de los resultados
        //construimos la cadena del adn que nos dan a ver si coincide completa
        //SI LOS EXONES O LOS INTRONES FUERAN MUY CORTOS ESTO NO FUNCIONARIA

        $cadena_en_adn="";
        for($k=$ref_longitud_antes_gorro;$k>=1;$k--) {
                $cadena_en_adn.=substr($dna,$b[0]-1-$k,1);
        }
        $cadena_en_adn.="^".substr($dna,$b[0]-1,1).substr($dna,$b[1]-1,1).substr($dna,$b[2]-1,1);//
debe coincidir con $campos[1]

        for($k=1;$k<$ref_longitud_despues_gorro-2;$k++) {
                $cadena_en_adn.=substr($dna,$b[2]-1+$k,1);
        }

        echo "<br>".$cadena_en_adn." cadena construida <br>";
        echo "".strtoupper($campos[1])." cadena valida <br>";
        if($cadena_en_adn==strtoupper($campos[1])) {
                $mut_correct++;
        }

        else {
                $mut_incorrect++;
                $incorrect[$mut_incorrect]=$campos;
                $incorrect[$mut_incorrect][0]=$ref_db[$i];
                echo "INCORRECT<br>";
        }

        $pp = $mut_dna_loc;
        $phenotype=$campos[3];
        echo "<br><b>Variation Table</b><br>";
        echo "ID_variation: $num<br>";
        echo "ID_gene_RT: $gen<br>";
        echo "ID_allele_num_RT: $allele<br>";
        echo "Specialization_effect: $specialization_effect<br>";
        echo "Specialization_mutant: $specialization_mutant<br>";
        echo "Specialization_localization: $specialization_localization<br>";
        echo "ID_databank: $ID_databank<br>";
        echo "ID_variation_DB:  $ref_db[$i]<br>";
        echo "Description: $description<br><br>";

        echo "<b>Precise Table</b><br>";
        echo "ID_variation: $num<br>";
        echo "Position: $pp<br>";
        echo "Type: $tipo<br>";
        echo "Ins_sequence: $reemplazo<br>";
        echo "Ins_repetition: $ins_repetition<br>";
        echo "Num_bases: $num_bases<br>";
        echo "Pheotipo: $phenotype<br><br>";
        //Aqui se debe hacer la insercion en la base de datos
        //insert_data_variation($c1,$num,$gen,$allele ,$specialization_effect ,
$specialization_mutant ,$specialization_localization ,$ID_databank ," $ref_db[$i]",
$description );   // Insert a row using c1
        //commit($c1);
        //insert_data_precise($c1,$num,$pp,$tipo ,$reemplazo ,$ins_repetition ,$num_bases,
$phenotype);   // Insert a row using c1
        //commit($c1);
}

//commit($c1);
?>
<TABLE style="border:1px solid green;"><tr><td>Mutaciones totales:</td><td><? echo
$totales_mutaciones; ?></td></tr><tr><td>Correctas</td><td><? echo $mut_correct; ?></td></
tr><tr><td>Incorrectas</td><td><? echo $mut_incorrect; ?></td></tr></TABLE>
<TABLE style="border:1px solid green;">
<?
for($i=1;$i<=count($incorrect);$i++){
?>
<tr>
<td><? echo $incorrect[$i][0]; ?></td>
<td><? echo $incorrect[$i][1]; ?></td>
<td><? echo $incorrect[$i][2]; ?></td>
<td><? echo $incorrect[$i][3]; ?></td>
```

```
<td><? echo $incorrect[$i][4]; ?></td>
<td> <? echo $incorrect[$i][5]; ?></td></tr>
<?
}
echo "</TABLE>";

?>
```

13.5 SMALL INDELS

```
<?php
// MYSQL DATABASE CONNECTION
include("includes/mysql_config.php");
$db = mysql_pconnect($dbhost,$dbuser,$dbpass);
mysql_select_db("$dbname",$db) or die('Cannot select database');

$gen="BRCA1";
$allele='1';
$specialization_effect ='M';
$specialization_mutant = 'O';
$specialization_localization = 'G';
$ID_databank = 'HGMD';
$description = '';
$tipo="ID";
$ins_sequence = '';
$ins_repetition = '';
$db = "(DESCRIPTION=(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = 158.42.186.17)(PORT = 1521)))
(CONNECT_DATA=(SID=genoma)))";
$num = 12999;
#$db= "";

# start Main program
echo "<h3> Proceso de Carga de ".$gen." - SMALL INDELS </h3><p><hr><p>";


$join=""; // NCBI CDS data

$small_indels=""; // HGMD mutational entries

// FUNCTIONS
include("lib/functions.php");

$query = "SELECT sequence FROM sequence WHERE name='cdna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$cdna = remove_space($row->sequence);

$query = "SELECT sequence FROM sequence WHERE name='dna' ";
$result = mysql_query($query) or die("SELECT Error: ".mysql_error());
$row = mysql_fetch_object($result);

$dna = remove_space($row->sequence);

//echo "<pre>".substr($cdna,0,60)."</pre>".strlen($cdna);
//echo  "<pre>".substr($dna,132543,6)."</pre><br>".strlen($dna);
echo "<pre>".$cdna."</pre>";//.strlen($cdna);
//echo  "<pre>".$dna."</pre><br>".strlen($dna);
$micDNA="";

$rangos=explode(",",$join);
for($i=1;$i<count($rangos);$i++) {

        $limites_exon[$i]=explode("..",$rangos[$i]);
        $longitud[$i]=$limites_exon[$i][1]-$limites_exon[$i][0]+1;
        $micDNA.=substr($dna,$limites_exon[$i][0]-1,$longitud[$i]);
}

//echo "<pre>".print_r($micDNA)."</pre>";
echo "<pre>".$micDNA."</pre>";//.strlen($micDNA);
echo strcmp($micDNA,$cdna);
for($i=0;$i<strlen($micDNA);$i++) {

        if($micDNA[$i]!=$cdna[$i]) echo "<br>".$i;
}
```

```
//busqueda de los codones
$lineas2=explode("\r",$small_indels);
for($i=0;$i<count($lineas2);$i++) {
	$r_db=explode("\t",$lineas2[$i]);
	$ref_db[$i]=$r_db[0];
}

$lineas=explode("\n", $small_indels);
$totales_mutaciones=count($lineas);
$mut_correct=0;
$mut_incorrect=0;
for($i=0;$i<count($lineas);$i++) {

	$num += 1; // mutation ID counter

	$campos=explode("\t",$lineas[$i]);//en las coordenadas 1 y 3 tenemos los cambios y el numero
de codon

	echo("<h3>REF: $ref_db[$i] Busqueda del codon $campos[3] cadena a añadir $campos[2] en
$campos[1]</h3>");
	//echo "<pre>";

	$long=2;
	$b=busqueda($campos[3],$longitud,$limites_exon,$dna,$cdna);
	$sin_cad_exones=explode("_",$campos[1]);
	if(count($sin_cad_exones)>1) $campos[1]=$sin_cad_exones[0].$sin_cad_exones[2];
	echo("<b>Busqueda del codon $campos[3] cadena a eliminar $campos[2] en $campos[1]</b><br /
><br />");
	$datos=explode("^",$campos[1]);
	//echo "<br>".$datos[0]."<br>";
	$lista=$campos[1];

	$num_bases = 0;
	$clean_list = str_replace("^", "", $lista);
	for($k=0;$k<strlen(($clean_list));$k++){
		if(ord($clean_list[$k])=="97" || ord($clean_list[$k])=="99" || ord($clean_list[$k])
=="103" || ord($clean_list[$k])=="116"){
			if($num_bases == 0) {$start_deletion = $k+1;}
			$num_bases += 1;
		}
	}

	$del_seq = substr($clean_list, $start_deletion-1, $num_bases);
	//echo $del_seq;

	$ref_longitud_total=strlen($campos[1])-1;//por el ^ que no cuenta hay que restar 1
	$ref_longitud_antes_gorro=strlen($datos[0]);
	$ref_longitud_despues_gorro=strlen($campos[1])-1-$ref_longitud_antes_gorro;
	$mut_dna_loc = ($start_deletion - ($ref_longitud_antes_gorro+1)) + $b[0];
	$nucleotide_pos = $b[0];

	//echo 'Reference codon nucleotidal position: '.$nucleotide_pos.'<br />';
	//echo 'Nucleotidal location of the mutation in reference to DNA: '.$mut_dna_loc.'<br />';
	//echo 'Number of substituted nucleotides: '.$num_bases.'<br />';
	//echo 'Mutation internal ID: '.$num.'<br />';

	//echo "codon: ".substr($dna,$b[0]-1,1).substr($dna,$b[1]-1,1).substr($dna,$b
[2]-1,1)."<br>";
	//echo "Cadena a insertar: ".strtoupper($campos[2])."<br>";
	$codon_valido=substr($datos[1],0,3);
	//comprobacion de validez de los resultados
	//construimos la cadena del adn que nos dan a ver si coincide completa
	//SI LOS EXONES O LOS INTRONES FUERAN MUY CORTOS ESTO NO FUNCIONARIA
	$cadena_en_adn="";
	for($k=$ref_longitud_antes_gorro;$k>=1;$k--) {
		$cadena_en_adn.=substr($dna,$b[0]-1-$k,1);
	}
	$cadena_en_adn.="^".substr($dna,$b[0]-1,1).substr($dna,$b[1]-1,1).substr($dna,$b[2]-1,1);//
debe coincidir con $campos[1]

	for($k=1;$k<$ref_longitud_despues_gorro-2;$k++) {
	$cadena_en_adn.=substr($dna,$b[2]-1+$k,1);
	}

	echo "<br>".$cadena_en_adn." cadena construida <br>";
	echo "".strtoupper($campos[1])." cadena valida <br>";
	if($cadena_en_adn==strtoupper($campos[1])) {
```

```
                $mut_correct++;
        }
        else {
                $mut_incorrect++;
                $incorrect[$mut_incorrect]=$campos;
                $incorrect[$mut_incorrect][0]=$ref_db[$i];
                echo "INCORRECT<br>";
        }

        $pp = $mut_dna_loc;
        $reemplazo = $campos[2];
        $phenotype=$campos[4];
        $num_bases_ins = strlen($reemplazo);
        $num_bases_del = strlen($del_seq);

        echo "<br><b>Variation Table</b><br>";
        echo "ID_variation: $num<br>";
        echo "ID_gene_RT: $gen<br>";
        echo "ID_allele_num_RT: $allele<br>";
        echo "Specialization_effect: $specialization_effect<br>";
        echo "Specialization_mutant: $specialization_mutant<br>";
        echo "Specialization_localization: $specialization_localization<br>";
        echo "ID_databank: $ID_databank<br>";
        echo "ID_variation_DB:  $ref_db[$i]<br>";
        echo "Description: $description<br><br>";

        echo "<b>Precise Table</b><br>";
        echo "ID_variation: $num<br>";
        echo "Position: $pp<br>";
        echo "Type: $tipo<br>";
        echo "Ins_sequence: $reemplazo<br>";
        echo "Del_sequence: $del_seq<br>";
        echo "Ins_repetition: $ins_repetition<br>";
        echo "Num_bases (Inserted): $num_bases_ins<br>";
        echo "Pheotipo: $phenotype<br><br>";
        echo "Num_bases (Deleted): $num_bases_del<br>";

        //Aqui se debe hacer la insercion en la base de datos
        //insert_data_variation($c1,$num,$gen,$allele ,$specialization_effect ,
$specialization_mutant ,$specialization_localization ,$ID_databank ," $ref_db[$i]",
$description );   // Insert a row using c1
        //commit($c1);
        //insert_data_precise($c1,$num,$pp,$tipo ,$reemplazo ,$ins_repetition ,$num_bases,
$phenotype);   // Insert a row using c1
        //commit($c1);
}
//commit($c1);
?>
<TABLE style="border:1px solid green;"><tr><td>Mutaciones totales:</td><td><? echo
$totales_mutaciones; ?></td></tr><tr><td>Correctas</td><td><? echo $mut_correct; ?></td></
tr><tr><td>Incorrectas</td><td><? echo $mut_incorrect; ?></td></tr></TABLE>
<TABLE style="border:1px solid green;">
<?
for($i=1;$i<=count($incorrect);$i++){
?>
<tr>
<td><? echo $incorrect[$i][0]; ?></td>
<td><? echo $incorrect[$i][1]; ?></td>
<td><? echo $incorrect[$i][2]; ?></td>
<td><? echo $incorrect[$i][3]; ?></td>
<td><? echo $incorrect[$i][4]; ?></td>
<td> <? echo $incorrect[$i][5]; ?></td></tr>
<?
}
echo "</TABLE>";

?>
```

13.6 IMPRECISE

```
<?php
// FUNCTIONS
include("lib/functions.php");
// MYSQL DATABASE CONNECTION
include("includes/mysql_config.php");
$db = mysql_pconnect($dbhost,$dbuser,$dbpass);
```

```
mysql_select_db("$dbname",$db) or die('Cannot select database');

$gen="BRCA1";
$db = "(DESCRIPTION=(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = 158.42.186.17)(PORT = 1521)))
(CONNECT_DATA=(SID=genoma)))";
$num = 16999;
#$db= "";

# start Main program
echo "<h3> Inserting the imprecise mutations of the BRCA1 gene</h3><p><hr><p>";


$grossdeletions=""; // HGMD gross deletion mutational entries


$grossinsertions=""; // HGMD gross insertion mutational entries

$complexrearrangements=""; // HGMD complex rearrangement mutational entries

echo '<h1>Gross Deletions</h1>';
$rows = explode("\n", $grossdeletions);
//print_r($rows);
for($i=1;$i<count($rows)-1;$i++) {
        $fields=explode("\t",$rows[$i]);
        $num += 1;
        $phenotype=$fields[2];
        echo $num.' '.$fields[0].' '.$fields[1].'<br />';
        echo "Phenotipo: $phenotype<br>ID: $num<br>";
        //insert_data_variation($c1,$num,$gen,'1' ,'M' ,'O' ,'C' ,'HGMD' ,"$fields[0]"," " );
        //insert_data_imprecise($c1,$num,$fields[1],$phenotype);
}
//commit($c1);
echo '<br /><br />';
$num = 17999;
echo '<h1>Gross Insertions</h1>';
$rows = explode("\n", $grossinsertions);
for($i=1;$i<count($rows)-1;$i++) {
        $fields=explode("\t",$rows[$i]);
        $phenotype=$fields[2];
        $num += 1; //running counter
        echo $num.' '.$fields[0].' '.$fields[1].'<br />';
        echo "Pheotipo: $phenotype<br>ID: $num<br>";
        //insert_data_variation($c1,$num,$gen,'1' ,'M' ,'O' ,'C' ,'HGMD' ,"$fields[0]"," " );
        //insert_data_imprecise($c1,$num,$fields[1],$phenotype);
}
//commit($c1);
echo '<br /><br />';
$num = 18999;
echo '<h1>Complex Rearrangements</h1>';
$rows = explode("\n", $complexrearrangements);
for($i=1;$i<count($rows)-1;$i++) {
        $fields=explode("\t",$rows[$i]);
        $phenotype=$fields[2];
        $num += 1; // running counter
        echo $num.' '.$fields[0].' '.$fields[1].'<br />';
        echo "Pheotipo: $phenotype<br>ID: $num<br>";
        //insert_data_variation($c1,$num,$gen,'1' ,'M' ,'O' ,'C' ,'HGMD' ,"$fields[0]"," " );
        //insert_data_imprecise($c1,$num,$fields[1],$phenotype);

}
//commit($c1);
echo '<br /><br />';
?>
```

13.7 FUNCTIONS.PHP

```
<?php
function remove_numbers($string) {
        $vowels = array("1", "2", "3", "4", "5", "6", "7", "8", "9", "0");
        $string = str_replace($vowels, '', $string);
        return $string;
}

function remove_space($str) {
        $str = str_replace(array("\n", "\r", "\t", " ", "\o", "\xOB"), '', $str);
        return $str;
```

```
}

function get_codon($cdna, $codon_nr) {

        $codon = 0;
        for ($i=0; $i <= strlen($cdna); $i++)
        {
                if(substr($cdna, $i, 1) == " ") {

                        $codon += 1;
                        if($codon == $codon_nr) {
                                $output = substr($cdna, $i-3 , 3);
                        }
                }
        }
        return $output;
}

function get_query_seq($cdna, $mutated_codon, $left, $right) {

        $codon = 0;
        for ($i=0; $i <= strlen($cdna); $i++)
        {
                if(substr($cdna, $i, 1) == " ") {

                        $codon += 1;
                        if($codon == $mutated_codon) {

                                $cdna_nwsp = remove_space($cdna);

                                $actual = ($i-($codon-1));
                                $pointer = ($actual-3)-($left);
                                $right = $left+$right+3;

                                $cdna_query_seq = substr($cdna_nwsp, $pointer, $right);
                        }
                }
        }
        return $cdna_query_seq;
}

function match_cdna_dna($dna, $cdna_sequence, $start) {

        if(substr($dna, $start, strlen($cdna_sequence)) == $cdna_sequence) {
                $output = 'TRUE';
        }
        else {
                $output = 'FALSE';
        }
        return $output;
}
function insert_data_variation($conn,$p1,$p2,$p3,$p4,$p5,$p6,$p7,$p8,$p9)
{
        $sql="insert into matthijs.VARIATION
  values('$p1','$p2','$p3','$p4','$p5','$p6','$p7','$p8','$p9')";
  //echo "$sql<br>";
        $stmt = ociparse($conn,$sql);
  ociexecute($stmt,OCI_DEFAULT);
  echo $conn." INSERTADO REGISTRO EN VARIATION\n\n";
}
function insert_data_precise($conn,$p1,$p2, $p3, $p4, $p5, $p6,$p7)
{   $sql="insert into matthijs.PRECISE
  values('$p1','$p2', '$p3', '$p4', '$p5', '$p6', '$p7')";
   //echo "$sql<br>";
        $stmt = ociparse($conn,$sql);
  ociexecute($stmt,OCI_DEFAULT);
  echo $conn." INSERTADO REGISTRO EN PRECISE\n\n";
}

function insert_data_imprecise($conn,$p1,$p2,$p3)
{   $sql="insert into matthijs.IMPRECISE
  values('$p1','$p2','$p3')";
   //echo "$sql<br>";
        $stmt = ociparse($conn,$sql);
  ociexecute($stmt,OCI_DEFAULT);
  echo $conn." INSERTADO REGISTRO EN PRECISE\n\n";
}
```

```
function commit($conn)
{ ocicommit($conn);
  //echo $conn." committed\n\n";
}

function rollback($conn)
{ ocirollback($conn);
  echo $conn." rollback\n\n";
}

function select_data($conn)
{ $stmt = ociparse($conn,"select * from veronica_nuevo.GENE");
  ociexecute($stmt,OCI_DEFAULT);
  echo $conn."----selecting\n\n";
  while (ocifetch($stmt))
  echo $conn." <".ociresult($stmt,"ID_SYMBOL").">\n\n";
  echo $conn."----done\n\n";
}
//calculo del nucleotido para un codon
function busqueda($codon,$longitud,$limites_exon,$dna,$cdna){
//$codon=2698;
        $c=($codon-1)*3+1;

        for($j=0;$j<3;$j++) {//para los 3 nucleotidos

                $l=0;

                for($i=1;$i<count($longitud)+1;$i++) {

                        if($longitud[$i]+$l<($c+$j)) {
                                $l+=$longitud[$i];
                        }
                        else {
                                $pos[$j]=$limites_exon[$i][0]+($c+$j-1-$l);
                                break;
                        }
                }
        }

        //echo "<br>Posiciones del codon: ".print_r($pos)."<br>";
        //echo substr($dna,$pos[0]-1,3)."<br>";
        //echo substr($cdna,$c-1,3)."<br>";
        return $pos;
}
?>
```