Gynaecological Patient Information Management System:

System Tests

Team Pentec:

Ruth Ojo 12042804
Liz Joseph 10075268
Trevor Austin 11310856
Maria Qumayo 29461775
Lindelo Mapumulo 12002862

Final Version

August 28, 2015

Contents

1 Introduction **2**

2 Features and Items tested **2**
2.1 Features and Items tested 2

3 Functional Testing **4**
3.1 PIMS Login 4
3.2 Login and Admnistartive user 4
3.3 PIMS Notifications 4
3.4 Login and Admnistartive user 4
3.5 PIMS Edit Profile 5
3.6 Login and Admnistartive user 5
3.7 PIMS Add User 6
3.8 Login and Admnistartive user 6
3.9 PIMS Artificial Inteligence 6
3.10 Login and Admnistartive user 6
3.11 PIMS Statistics 7
3.12 Login and Admnistartive user 7
3.13 PIMS Predictions 7
3.14 Login and Admnistartive user 7

4 Non-functional Testing **8**
4.1 Usability 8
4.2 Scalability 9
4.3 Performance 10
4.4 Maintainability 10
4.5 Reliability 11
4.6 Secutity 12
4.7 Monitorability 12

5 Remarks **13**
5.1 Risks and issues 13
5.2 Product quality 13
5.3 Possible improvements 14

6 Conclusion **14**

1 Introduction

This document documents and taracks the necessary information required to effectively define the approach to be used in the testing and evaluationg of the Patient Information Management System designed by the group Pentect for the Kalafong

The following Patient Infromation Management System Use cases that were thoroughly tested are:

- User Login
- PIMS Artificial Intelligence
- PIMS Statistics
- PIMS Notifications
- PIMS Space

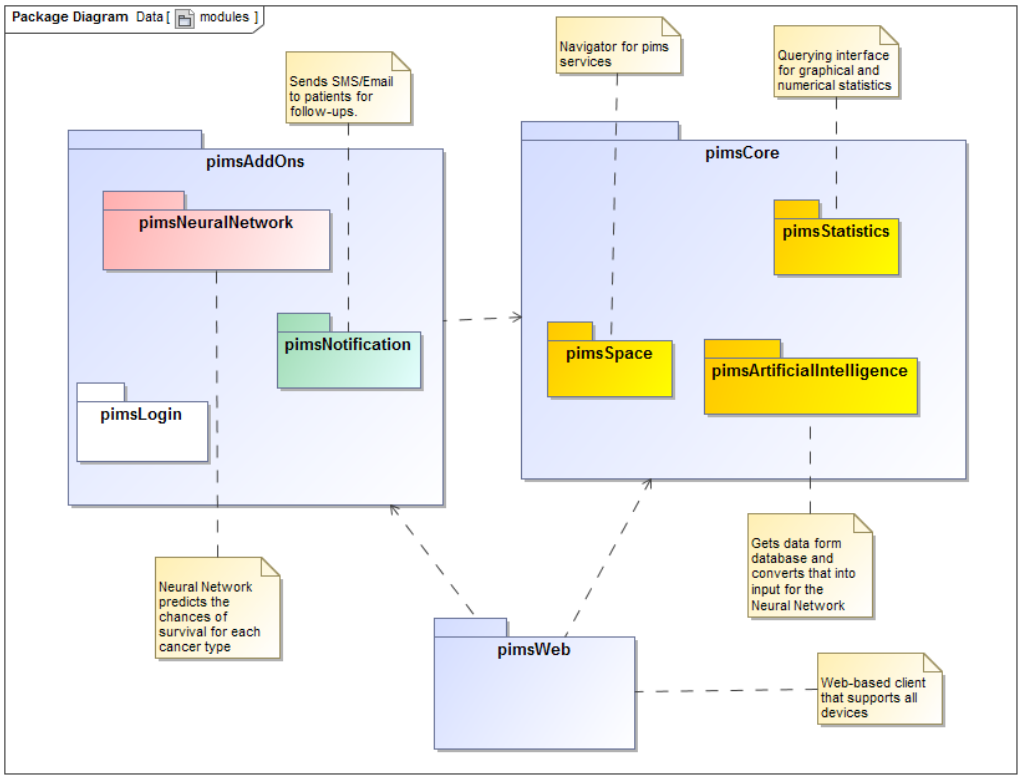
Testing was done on the system mainly for the following 5 reasons:

- To ensure that the system meets both functional and non functional requirements arroding to the given spesifications
- System stress, that is to make sure that the system does not fail with multiple users or any other factors because it is expensive to resolve and fix at a later stage.
- To handle and resolve System failures and bugs appropriatly and in good time.
- To point out the defects and errors that were made during and after the development phases.
- To ensure that the final product is of top, professiona, software engineering standards.

2 Features and Items tested

2.1 Features and Items tested

Our testing involved looking at the core functional requirements as given in our client's spesification document. As well as those we added on as "Nice to have". The following use cases and features as depicted in the PIMS master node module were tested.



3 Functional Testing

3.1 PIMS Login

3.2 Login and Admnistartive user

PIMS Login

Functional Testing - for each use case tested * either success or a list of violations of the contract requirements (pre- and post-condition violations or data structure requirements) * a test coverage analysis reporting which percentage of the use cases have been covered by the testing

2. Non-functional testing/assessment - any performance, scalability, maintainability, reliability, usability, ... problems identified with evidence for the identified problem.

- The Buzz Space system has to accommodate and host a multitude of users concurrently, thus making it prone to various malfunctions and glitches
- The Space needs to be monitored in real time at all times to ensure relevance of topics and subject matters.
- The rating and tagging functionality need to be fair and accurate
- Sufficient feedback and updates of the Buzz Space state must be provided to the users. Users that create threads or just comment on one.
- General software control and application usage..

3.3 PIMS Notifications

3.4 Login and Admnistartive user

PIMS Login

Functional Testing - for each use case tested * either success or a list of violations of the contract requirements (pre- and post-condition violations or data structure requirements) * a test coverage analysis reporting which percentage of the use cases have been covered by the testing

2. Non-functional testing/assessment - any performance, scalability, maintainability, reliability, usability, ... problems identified with evidence for the identified problem.

- The Buzz Space system has to accommodate and host a multitude of users concurrently, thus making it prone to various malfunctions and glitches
- The Space needs to be monitored in real time at all times to ensure relevance of topics and subject matters.
- The rating and tagging functionality need to be fair and accurate
- Sufficient feedback and updates of the Buzz Space state must be provided to the users. Users that create threads or just comment on one.
- General software control and application usage..

3.5 PIMS Edit Profile

3.6 Login and Admnistartive user

PIMS Login

Functional Testing - for each use case tested * either success or a list of violations of the contract requirements (pre- and post-condition violations or data structure requirements) * a test coverage analysis reporting which percentage of the use cases have been covered by the testing

2. Non-functional testing/assessment - any performance, scalability, maintainability, reliability, usability, ... problems identified with evidence for the identified problem.

- The Buzz Space system has to accommodate and host a multitude of users concurrently, thus making it prone to various malfunctions and glitches
- The Space needs to be monitored in real time at all times to ensure relevance of topics and subject matters.
- The rating and tagging functionality need to be fair and accurate
- Sufficient feedback and updates of the Buzz Space state must be provided to the users. Users that create threads or just comment on one.
- General software control and application usage..

3.7 PIMS Add User

3.8 Login and Admnistartive user

PIMS Login

Functional Testing - for each use case tested * either success or a list of violations of the contract requirements (pre- and post-condition violations or data structure requirements) * a test coverage analysis reporting which percentage of the use cases have been covered by the testing

2. Non-functional testing/assessment - any performance, scalability, maintainability, reliability, usability, ... problems identified with evidence for the identified problem.

- The Buzz Space system has to accommodate and host a multitude of users concurrently, thus making it prone to various malfunctions and glitches
- The Space needs to be monitored in real time at all times to ensure relevance of topics and subject matters.
- The rating and tagging functionality need to be fair and accurate
- Sufficient feedback and updates of the Buzz Space state must be provided to the users. Users that create threads or just comment on one.
- General software control and application usage..

3.9 PIMS Artificial Inteligence

3.10 Login and Admnistartive user

PIMS Login

Functional Testing - for each use case tested * either success or a list of violations of the contract requirements (pre- and post-condition violations or data structure requirements) * a test coverage analysis reporting which percentage of the use cases have been covered by the testing

2. Non-functional testing/assessment - any performance, scalability, maintainability, reliability, usability, ... problems identified with evidence for the identified problem.

- The Buzz Space system has to accommodate and host a multitude of users concurrently, thus making it prone to various malfunctions and glitches

- The Space needs to be monitored in real time at all times to ensure relevance of topics and subject matters.
- The rating and tagging functionality need to be fair and accurate
- Sufficient feedback and updates of the Buzz Space state must be provided to the users. Users that create threads or just comment on one.
- General software control and application usage..

3.11 PIMS Statistics

3.12 Login and Admnistartive user

PIMS Login

Functional Testing - for each use case tested * either success or a list of violations of the contract requirements (pre- and post-condition violations or data structure requirements) * a test coverage analysis reporting which percentage of the use cases have been covered by the testing

2. Non-functional testing/assessment - any performance, scalability, maintainability, reliability, usability, ... problems identified with evidence for the identified problem.

- The Buzz Space system has to accommodate and host a multitude of users concurrently, thus making it prone to various malfunctions and glitches
- The Space needs to be monitored in real time at all times to ensure relevance of topics and subject matters.
- The rating and tagging functionality need to be fair and accurate
- Sufficient feedback and updates of the Buzz Space state must be provided to the users. Users that create threads or just comment on one.
- General software control and application usage..

3.13 PIMS Predictions

3.14 Login and Admnistartive user

PIMS Login

Functional Testing - for each use case tested * either success or a list of violations of the contract requirements (pre- and post-condition violations

or data structure requirements) * a test coverage analysis reporting which percentage of the use cases have been covered by the testing

2. Non-functional testing/assessment - any performance, scalability, maintainability, reliability, usability, ... problems identified with evidence for the identified problem.

- The Buzz Space system has to accommodate and host a multitude of users concurrently, thus making it prone to various malfunctions and glitches
- The Space needs to be monitored in real time at all times to ensure relevance of topics and subject matters.
- The rating and tagging functionality need to be fair and accurate
- Sufficient feedback and updates of the Buzz Space state must be provided to the users. Users that create threads or just comment on one.
- General software control and application usage..

4 Non-functional Testing

4.1 Usability

Description

The PIMS should be easily Maintainable in future. Thus needs to be flexible and extensible.

Justification

Software always needs new features or bug fixes. Maintainable software is easy to extend and fix, which encourages the software's uptake and use.

Mechanism

1. Strategy:
    - Open-source resources: using open source resources to minimize update costs in the future.
    - Iterative development and regular reviews: This will help us improve system quality.

- Prevention is better than cure: Here we get others(lecturers) to review your code, to make sure its clean.The cleaner our code, the cheapre it is to maintain.
- Version control:Thsi will help keep our code, tests and documentation up to date and synchronised. It will also help us keep track of progress.
- Documentation: Relevant documentation will help future developers understand the software and system as a whole.

4.2 Scalability

Description

The PIMS should be easily Maintainable in future. Thus needs to be flexible and extensible.

Justification

Software always needs new features or bug fixes. Maintainable software is easy to extend and fix, which encourages the software's uptake and use.

Mechanism

1. Strategy:
    - Open-source resources: using open source resources to minimize update costs in the future.
    - Iterative development and regular reviews: This will help us improve system quality.
    - Prevention is better than cure: Here we get others(lecturers) to review your code, to make sure its clean.The cleaner our code, the cheapre it is to maintain.
    - Version control:Thsi will help keep our code, tests and documentation up to date and synchronised. It will also help us keep track of progress.
    - Documentation: Relevant documentation will help future developers understand the software and system as a whole.

4.3 Performance

Description

The PIMS should be easily Maintainable in future. Thus needs to be flexible and extensible.

Justification

Software always needs new features or bug fixes. Maintainable software is easy to extend and fix, which encourages the software's uptake and use.

Mechanism

1. Strategy:

    - Open-source resources: using open source resources to minimize update costs in the future.
    - Iterative development and regular reviews: This will help us improve system quality.
    - Prevention is better than cure: Here we get others(lecturers) to review your code, to make sure its clean.The cleaner our code, the cheapre it is to maintain.
    - Version control:Thsi will help keep our code, tests and documentation up to date and synchronised. It will also help us keep track of progress.
    - Documentation: Relevant documentation will help future developers understand the software and system as a whole.

4.4 Maintainability

Description

The PIMS should be easily Maintainable in future. Thus needs to be flexible and extensible.

Justification

Software always needs new features or bug fixes. Maintainable software is easy to extend and fix, which encourages the software's uptake and use.

Mechanism

1. Strategy:

    - Open-source resources: using open source resources to minimize update costs in the future.
    - Iterative development and regular reviews: This will help us improve system quality.
    - Prevention is better than cure: Here we get others(lecturers) to review your code, to make sure its clean.The cleaner our code, the cheapre it is to maintain.
    - Version control:Thsi will help keep our code, tests and documentation up to date and synchronised. It will also help us keep track of progress.
    - Documentation: Relevant documentation will help future developers understand the software and system as a whole.

4.5 Reliability

Description

The PIMS should be easily Maintainable in future. Thus needs to be flexible and extensible.

Justification

Software always needs new features or bug fixes. Maintainable software is easy to extend and fix, which encourages the software's uptake and use.

Mechanism

1. Strategy:

    - Open-source resources: using open source resources to minimize update costs in the future.
    - Iterative development and regular reviews: This will help us improve system quality.
    - Prevention is better than cure: Here we get others(lecturers) to review your code, to make sure its clean.The cleaner our code, the cheapre it is to maintain.

- Version control:Thsi will help keep our code, tests and documentation up to date and synchronised. It will also help us keep track of progress.
- Documentation: Relevant documentation will help future developers understand the software and system as a whole.

4.6 Secutity

Description

The PIMS should be easily Maintainable in future. Thus needs to be flexible and extensible.

Justification

Software always needs new features or bug fixes. Maintainable software is easy to extend and fix, which encourages the software's uptake and use.

Mechanism

1. Strategy:
    - Open-source resources: using open source resources to minimize update costs in the future.
    - Iterative development and regular reviews: This will help us improve system quality.
    - Prevention is better than cure: Here we get others(lecturers) to review your code, to make sure its clean.The cleaner our code, the cheapre it is to maintain.
    - Version control:Thsi will help keep our code, tests and documentation up to date and synchronised. It will also help us keep track of progress.
    - Documentation: Relevant documentation will help future developers understand the software and system as a whole.

4.7 Monitorability

Description

The PIMS should be easily Maintainable in future. Thus needs to be flexible and extensible.

Justification

Software always needs new features or bug fixes. Maintainable software is easy to extend and fix, which encourages the software's uptake and use.

Mechanism

1. Strategy:

    - Open-source resources: using open source resources to minimize update costs in the future.
    - Iterative development and regular reviews: This will help us improve system quality.
    - Prevention is better than cure: Here we get others(lecturers) to review your code, to make sure its clean.The cleaner our code, the cheapre it is to maintain.
    - Version control:Thsi will help keep our code, tests and documentation up to date and synchronised. It will also help us keep track of progress.
    - Documentation: Relevant documentation will help future developers understand the software and system as a whole.

5 Remarks

5.1 Risks and issues

5.2 Product quality

Description

The PIMS should be easily Maintainable in future. Thus needs to be flexible and extensible.

Justification

Software always needs new features or bug fixes. Maintainable software is easy to extend and fix, which encourages the software's uptake and use.

Mechanism

1. Strategy:

- Open-source resources: using open source resources to minimize update costs in the future.
- Iterative development and regular reviews: This will help us improve system quality.
- Prevention is better than cure: Here we get others(lecturers) to review your code, to make sure its clean.The cleaner our code, the cheapre it is to maintain.
- Version control:Thsi will help keep our code, tests and documentation up to date and synchronised. It will also help us keep track of progress.
- Documentation: Relevant documentation will help future developers understand the software and system as a whole.

5.3 Possible improvements

6 Conclusion