# CITS4401 Software Requirements and Design
# Practical Assignment

(worth 25% of total assessment)

**Due: Friday, May 1st, 2015 @4pm**

**Aim:** The aim of this group assignment is to test your ability to design and implement a medium sized software system and to demonstrate your ability to manage and deliver an oral presentation of your work.

It is recommended that you group with another student to do the assignment. If you choose to work on your own, the workload will be very high. The same marking criteria will be used whether the assignment is done collaboratively or individually.

## 1. System objectives

Your task for this assignment is to design and implement a *Swimming Pool Automated Checking System* (SPACS). SPACS is a software application to assist in the task of keeping privately owned back-yard swimming pools in a safe condition. In particular, it will attend to the chemical balance and the functioning of the pump, filter, and chlorinator.

Although the SPACS should be a program running on a server with users communicating with SPACS via the internet, for this assignment, we accept the implementation of the system as a computer program installed on a single computer used by different categories of users. You may choose to implement the system using any of the following languages: Java, C, C++, or Python. If you prefer to use a language not in the list above, please see me as soon as possible. Please note that languages that are too old (e.g., Cobol, Fortran) or not widely used (as installing system programs to run your codes may not be feasible) will not be acceptable.

## 2. System context

1. The SPACS is a central server program that manages the collection and analysis of measurements from pools around a city (or similar geographical area).
2. It will require each pool to be registered with a local pool chemical supply shop that is also connected to the system.
3. The system relies on the installation of automated **pool testing units** (PTUs) which connects to the internet through the household modem. The PTUs and programs installed on the hardware units are developed by an electrical/chemical engineering firm separately from the SPACS.
4. The web-based interfacing between SPACS and the PTUs is the responsibility of the SPACS development team.
5. The web-based interfacing between pool owners or pool shop administrator and the SPACS is also the responsibility of the SPACS development team.
6. The data collected by each PTU will include the date/time of measurement and the following: The Ph (acidity level); ORP(chlorine level); TA (Total Alkalinity); temperature; water hardness level; the date/time of the last operation of the filter, water flow rate through the filter, the chlorinator status and the water level status at that time.

   There will be indication of any missing data items and of the status of any PTU alarms (e.g., water level too low, pump fails to operate).
7. For each value that is measured, and for several other values that can be computed from them, there is a smaller “recommended range” and a wider “safe range” that have been pre-determined. If a pool’s value goes outside the recommended range, then it means that, within the next month or so, some treatment program should be undertaken to bring it back in line. If a

pool’s value goes outside the safe range, then this means that the pool should not be used until some treatment is undertaken.

## 3. Functional requirements

### 3.1 Regular Check-In

1) The PTU program running on the hardware of the PTU is programmed to gather pool measurements at a certain time every day. It is also programmed to send in the latest data to the SPACS across the Internet at a certain time every 5 days. This is referred to as a *regular check-in*.
2) The PTU program will first attempt to authenticate itself to the SPACS via a login ID code and password.
3) The SPACS will indicate successful authentication or that authentication is incorrect.
4) If authentication is successful then the PTU program will submit data from its latest set of measurements to the SPACS.
5) The SPACS will record the submitted measurements and then acknowledge receipt to the PTU program.
6) If the PTU program receives acknowledgment before a set time-out period, then it terminates.
7) The SPACS will process the submitted data to check for alarm values, missing values, or values outside safe ranges for that pool which indicate a need for some urgent action. If this is needed, appropriate emails are sent to the pool owner and the registered pool shop for the pool. If emails are not needed or have been sent if they are needed, then that is the end of regular check-in.
8) If the PTU program is notified that its authentication is incorrect, then it will not attempt a regular check-in again until it is reset with login and password data.
9) If the PTU program does not receive a response from the SPACS to its authentication attempts or to its submission of data, then it is programmed to attempt to check-in again at regular intervals until it is successful.

### 3.2 Urgent Check-In

1) The PTU program is programmed to gather pool measurements at a certain time every day. If it detects a measurement value outside the safe range, then it is programmed to report to the SPACS immediately, e.g., if the PTU program receives any of the alarm signals from the PTU (such as water level too low, pump fails to operate). This is referred to as an *urgent check-in*.
2) The PTU program will first attempt to authenticate itself to the SPACS via a login ID code and password.
3) The SPACS will indicate successful authentication or that authentication is incorrect.
4) If authentication is successful, the PTU program will submit data from its latest set of measurements to the SPACS. This will include the measurements as in a regular check-in and the status of any alarms.
5) The SPACS will record the submitted measurements and then acknowledge receipt to the PTU program.
6) If the PTU program receives acknowledgment before a set time out period, then it terminates.
7) The SPACS will process the submitted data to check for alarm values, missing values, or values outside safe ranges for that pool which indicate a need for some urgent action. If this is needed, then appropriate emails are sent to the pool owner and the registered pool shop for the pool. If emails are not needed or have been sent if they are needed, then that is the end of urgent check-in. (Note that it is possible that the PTU program and the SPACS test algorithm disagree on whether urgent action is needed for a particular pool with particular measurements).
8) If the PTU program is notified that its authentication is incorrect, then it is programmed to notify the house owner and the pool shop. It will not attempt a check-in again until it is reset.

9) If the PTU program does not receive a response from the SPACS to its authentication attempts or to its submission of data, then it is programmed to attempt to check-in again at regular intervals until it is successful.

### 3.3 No Check-In Alarm Notification

1) If the SPACS fails to get a measurement data set sent by the PTU program during any 5 day period, then it will send an appropriate email to the pool owner and the pool shop.

### 3.4 Regular Status Report

1) During the first week after a new pool is registered with the SPACS and at monthly intervals after that, the SPACS will prepare a status report for the pool based on the latest set of measurements.
2) The report is prepared by an algorithm on the SPACS. The algorithm indicates the latest measurements, trends in certain measurements over the month, indications of any measurements which are outside of recommended ideal ranges and recommendations for a pool treatment program involving the staged addition of appropriate chemicals.
3) The report is emailed to the pool owner and the pool shop administrator.
4) The report is also recorded for subsequent viewing by the pool owner, the pool shop administrator, and the SPACS administrator.
5) If any emails fail to be sent, then the SPACS server program attempts to resend them at regular intervals and notifies the SPACS administrator.

### 3.5 View Report

1) At any time for any pool, the owner and pool shop administrator may, after authentication to the SPACS, view the latest Status Report over the Internet.

### 3.6 Remove/Add/Edit Pool

1) Only a pool shop administrator may add, remove or edit details of a pool.
2) This is done, after authentication, via a web form.
3) When a pool shop administrator adds a pool to the system, the SPACS will record the information about that pool that is needed for identification, contact purposes and for determining safe and ideal measurements for the pool and recommendations for treatment programs. This includes the contact person, i.e., the pool owner's name, address, and email. It also includes the shop which is the current authenticated user and which is registered as the contact shop for that pool. It also includes the pool, filter, pump and chlorinator type and pool capacity. The SPACS will check the information entered and report any clear mistakes before acknowledging acceptance.
4) When adding a new pool, the pool shop administrator will enter a login name and password for the pool owner and a login code and password for the PTU program for the pool. The SPACS will check these data before acknowledging acceptance. Logins must be unique and passwords must conform to several security requirements (e.g., each password must contain at least one upper and lower case letter and at least one digit).
5) The SPACS will acknowledge when registration of the new pool is complete and send a welcome email to the pool owner.
6) Only the registered shop for a particular pool may request its removal from the system. Then all records for that pool are removed from the SPACS and an email is sent to the owner.
7) Only the administrator of the registered shop for a particular pool may edit the recorded details about that pool. New information, such as logins, passwords etc, are checked before any changes are made.

### 3.7 Remove/Add/Edit Shop

1) Only the SPACS administrator may add, remove or edit details of a shop.
2) When a shop is added to the system, SPACS will record the information about that shop that is needed for identification and contact purposes. This includes the shop administrator's name,

address, and email. The SPACS will check the information entered and report any clear mistakes before acknowledging acceptance.

3) When the SPACS administrator enters a login name and password for the pool shop's administrator, the SPACS will check these data before acknowledging acceptance. Logins must be unique and passwords must conform to several security requirements.
4) The SPACS will acknowledge when registration of the new shop is complete and send a welcome email to the pool shop administrator.
5) On removal of a shop, all records for that shop are removed and an email is sent to the shop administrator.
6) The SPACS administrator may edit the recorded details about a shop. New information, logins, passwords etc are checked before any changes are recorded.
7) The SPACS administrator can transfer a pool or a set of pools registered with one shop to be registered with another shop. The SPACS will send emails to all the shops and owners concerned.

## 4. Quality Requirements of the SPACS

### 4.1 Performance

4.1.1 SPACS shall show no visible deterioration in response time as the number of pools or shops in the system increases.

4.1.2 SPACS shall require a reasonably small amount of memory so that enough of it is permanently resident on the server to provide quick service.

4.1.3 SPACS shall load as quickly as comparable productivity tools on whatever environment it is running in.

4.1.4 SPACS will respond to client web activity in a timely and convenient way.

### 4.2 Reliability

4.2.1 SPACS shall be available for use as much as comparable productivity tools.

### 4.3 Usability

4.3.1 SPACS shall provide a standard style of user interface so that users do not have to learn a new style of interaction.

4.3.2 Users will be able to understand the layout and options of the SPACS User Interface.

4.3.3 Notification and email messages generated by SPACS shall be clear, succinct, and polite and free of jargon.

### 4.4 Portability

4.4.1 SPACS will be implemented on a platform that allows easy re-hosting on different hardware and OS.

### 4.5 Modifiability

4.5.1 SPACS will be implemented using modern programming practices that maximize the maintainability and re-usability of designs and code.

4.5.2 SPACS will be implemented in such a way that alternative PTU programs could be used easily without affecting the logic of the design.

### 4.6 Future Requirements

4.6.1 Support for spas and other types of water features which have quite different measurements and ranges of acceptable values.

4.6.2 Support for human reports of problems to be handled.

4.6.3 Support for scheduling of maintenance visits by pool technicians.

## 5. What You Need to Submit

### 5.1 A Report on Your Design of the SPACS (70%)

1. **Functional modelling.** Provide a use case diagram containing appropriate choice of actors and use cases which covers all the functionality of the given requirements definition but does not invent any new functionality. In the diagram, you should use the client's language and not new or ambiguous terms. Boundary between this system and external systems should be clearly identified and justified. For each use case in your diagram, you should provide a table that describes the use case (see the lecture note for the template).

2. **Object modelling.** Provide a class diagram to show the significant relationships between objects identified in the SPACS. Reasonable roles and multiplicities should be given in your diagram. For each class in your class diagram, supply the attributes (including the types) and major operations. Again, use the client's language and do not invent new or ambiguous terms. Names of attributes and operations should be explained. To avoid putting too much information into a single diagram, it is recommended that your main class diagram for the SPACS should contain only the names of all the classes. For each class in the main class diagram, you can then draw a separate class diagram showing the attributes and operations.

3. **Dynamic modelling part a.** Provide 5 sequence diagrams chosen from any 5 use cases in your use case diagram above. Identify in each sequence diagram the participating actors and objects. Explain the type (*entity*, *boundary*, or *control*) of each object. Clearly label the messages sent between the participating objects. Your sequence diagram should be consistent with your class diagrams.

4. **Design constraints.** List all the design constraints for the SPACS and prioritize them.

5. **Subsystem decomposition.** Describe how you would decompose SPACS into subsystems. Describe the services provided by each subsystem. Justification of your subsystem decomposition (such as coupling and cohesion) should be given. Are there other ways to decompose the system?

6. **Dynamic modelling part b.** Supply 2 Statechart diagrams for 2 selected subsystems of your choice. Give a brief description for each diagram. Choose meaningful names for the states and transitions. Again, use the client's terminology.

7. **Design patterns.** List two design patterns that would be appropriate to be used in the system. Describe which part of the system that each pattern is suitable and justify why you chose these patterns.

### 5.2 Implementation/Coding of SPACS (30%)

Due to the large scope of the SPACS, you are required to code only a small subset of the system. Your implementation should include

- the adding/removing/editing of pool shops in SPACS by the SPACS administrator;
- the adding/removing/editing of pools by the pool shop administrator.

  To reduce the amount of information about each pool that you need to implement, you may consider only the following attributes: the dimension (length, width, and depth) of the pool, the material that the pool is made of, and whether the pool is in-ground or above-ground (search the web, e.g., http://en.wikipedia.org/wiki/Swimming_pool, for information about different pool materials and realistic pool dimensions). Note that each pool should include information about the contact person as well. You may assume that the contact person is the same as the pool owner and that each pool shop has only one pool shop administrator.

- the viewing of status reports by the pool owners, pool shop administrators, and SPACS administrator.

You should create some appropriate fictitious data and these data should be stored in files external to the system (i.e., do not hard-code the data in your code). You are free to choose any data format (including but not limited to csv, xls, xlsx, relational database, Java serialization, simple text files, or in-house binary files). Any addition, removal, editing tasks above should directly change these data files. The status reports  can be fictitious also.

To make it easier for me to test run your code, your program should have a simply GUI that is easy to follow. Your GUI layout will not be marked; however, the correctness of the code and the ease of navigating the program will be marked. If you intend to use Java, you may find it useful to use NetBeans which supports drag-and-drop of graphics/GUI items. BlueJ may be useful for general Java class editing and testing. You may be able to draw class diagrams directly in BlueJ.

Supply a one-page **readme.txt** file (in plain text format) which should describe how to compile and run your code.  In your **readme.txt** file, please include the logins and passwords of a few users that I can use to test run your code. Please also indicate the role of each user in the system.

If you know how to write a Makefile, then supply that as well.

### 5.3 Work sharing

This section is only relevant if the assignment is done  collaboratively by two students. How the work was split between the group members? This should be 1/4 to 1/2 page long. You may refer to earlier parts of the report when you describe about this, but do not duplicate what you have already written.

### 5.4 Submission instructions

All the submissions should be done via the cssubmit website:

**https://secure.csse.uwa.edu.au/run/cssubmit**

- For your design report in part 5.1, submit a pdf file. Please supply the full names and student numbers of all the team members on the first page of the pdf file.
- For your computer codes in part  5.2, zip all the program source files, data files, readme.txt file, and the Makefile (optional) together and submit the zip file.

Submit all the files to cssubmit by the due day and time. If worked collaboratively, any one of the group members may submit. Please make sure that you agree between yourselves who is responsible for the submission. Do not submit multiple copies of the assignment.

### 5.5 Presentation/seminar

All the team members must be present to give the seminar. Each seminar will be approximately 20-30 minutes long (including question time). Please spend approximately 60-70% of the time on explaining your design for part 5.1 and the remaining time on showing how your code works. There is no need to create fancy Powerpoint slides. Your assignment will be marked mostly on your submitted  files. However, a 20% penalty on the total mark will be imposed if a team is absent from the seminar. In the seminar, you should focus on the important aspects of your design as there may not be sufficient time to go through all the fine details of your design.
More details about the seminar timetable will be given later.

**Late submission policy:** http://web.csse.uwa.edu.au/students/assessments/late-submissions

*Du Huynh*, Unit Coordinator.