Chapter 2 Additional Background

While Chapter One provided an introduction to the model used as the foundation for SIMPRAC, and the literature supporting this approach. This chapter reviews additional literature that has influenced the final design and development. The chapter begins with an overview of problem-based learning in medicine, as the medical problem forms the clinical context in which learning about medical practice takes place. The chapter then goes on to review literature regarding the design of patient simulations, and summarises those elements that have been included in SIMPRAC. Consideration is then given to reflection, as the core learning aspect being explored. Finally, approaches to software evaluation are reviewed as a basis for developing and evaluating SIMPRAC.

2.1 Problem-Based Learning

In recent years there has been a shift from so-called conventional curricula to problem-based learning (PBL) curricula for the training of medical students. In the former, instruction in the basic biomedical sciences, including chemistry, mathematics, physics, anatomy and physiology is given prior to instruction on clinical problem-solving. In contrast, with problem-based learning, biomedical components and clinical components are fully integrated (Patel et al., 2000). According to Barrows (1986), problem-based learning has the potential to address at least four educational objectives. These are:

1. to structure knowledge within a clinical context;
2. to develop an effective clinical reasoning process;
3. to develop self-directed learning skills;
4. to increase motivation for learning.

However, not all forms of problem-based learning are able to achieve these outcomes. Barrows asserts that programs which do encourage these objectives should have the following characteristics:

1. The patient problem should allow free information inquiry that is hypothesis-driven and a tutor should be available to guide exploration, if necessary.
2. Exploration should initially be based on the student's existing knowledge, as this facilitates understanding and retention of new knowledge.
3. Ideally, there should be an evaluation process following self-directed study, where students evaluate the quality of information available and review their decision-making process.

In the design of SIMPRAC, these characteristics first point to the need to ensure users are not restricted in how they elicit information. Second, that learners are encouraged to develop hypotheses, based on the information they elicit, which in turn directs further exploration. Furthermore, it highlights the importance of the user being able to review their decision-making process.

In recent years, some medical schools have developed curricula using a problem-based learning approach, in which biomedical components and clinical components are fully integrated. Clinically meaningful problems are introduced at the beginning of the course on the premise that knowledge taught abstractly does not help students integrate this knowledge with clinical practice. Patel and colleagues (Patel and Groen, 1993, Patel et al., 2000) have investigated both in the short and longer term, differences in the outcome of conventional curricula, in which biomedical science concepts are taught before the student is introduced to clinical medicine, and problem-based learning

curricula. In their study, medical students from the conventional curriculum school did not integrate the basic science and clinical knowledge, except when they experienced difficulty with the patient problem. For students in the PBL course, basic science and clinical knowledge were tightly integrated and in many cases, they were not able to distinguish the two types of knowledge. Interestingly, the PBL students also developed elaborations that, while used for supporting the diagnosis, were not required to generate the diagnosis.

For example, in a case concerning bacterial endocarditis (a bacterial infection of the heart), a senior medical student in the PBL school noted that, “The patient’s circulatory distress appears to be related to systemic vasodilation”. They went on to elaborate, “This could explain both the wide pulse pressure with low systolic and diastolic levels and the bounding and disappearing pulse. Because flow (cardiac output, which = heart rate x stroke volume) is proportionate to blood pressure over resistance, any drop in total peripheral resistance would require the heart to increase stroke volume and/or heart rate in order to maintain blood pressure at a reasonable level” (Patel and Groen, 1993).

With the PBL curriculum, students are taught to use a hypothetico-deductive approach, also known as backward reasoning (hypothesis to data). Unfortunately, their elaborations were accompanied by a tendency to generate errors of scientific fact and flawed patterns of explanation, such as circular arguments. In contrast, explanations by conventional curriculum students were more likely to use forward directed reasoning (from data to hypothesis) and they generated fewer elaborations and fewer errors (Patel and Groen, 1993). Patel (2000) has also shown that these patterns of reasoning continue into post-graduate years.

The issue of directionality of reasoning is of interest, as it has been demonstrated that experts use forward reasoning, in which findings in the clinical case lead toward hypotheses about the patient's condition (Patel et al., 2000). These "expert" physicians do not make routine use of biomedical knowledge in diagnosing clinical cases. However, one would need to consider the educational backgrounds of these "experts" as it would not be surprising that forward reasoning predominated if they were trained using conventional curricula.

It is worthy to note that these findings are consistent with the argument made by Anderson et al. (1995b) that optimal learning occurs when a combination of abstract and situation-specific training is provided. In contrast to situated learning, and what they term "radical constructivism", which claims that all learning should occur in an authentic environment, these authors provide evidence that complex problems within complex environments can be overwhelming and impede learning. Instead, they believe that complex problems should initially be decomposed into their component activities and decontextualized so that principles can be learned and transferred. In the medical context, this may mean receiving instruction on how to take a history and examine a patient properly rather than trying to divine these skills while seeing a patient for the first time. On the other hand, they do see value in authentic environments once the required skills have been learned, as there are elements that can only be learned from the challenges of complex problems in different contexts.

With respect to SIMPRAC, the findings of Anderson et al. (1995b) and Patel (2000) warn that while simulation environments provide a useful, and necessary, context for

learning, they should not be used in isolation. On the other hand, there are some lessons that can only be learned within an authentic context. To this end, and consistent with the model illustrated in Figure 1 on page 15, SIMPRAC needs to be able to simulate the complexity of a real consultation, enable free hypothesis-driven enquiry, and enable the user to review their decision-making process.

2.2 Patient Simulations

In terms of problem-based learning and the constructivist approach to learning, patient simulations provide an excellent context for learning as they have the potential to provide realistic medical problems. In contrast to real patients, simulators do not get stressed or embarrassed, and can be available at anytime (Issenberg et al., 1999). Moreover, they usually have predictable behaviour enabling a standardised experience (Issenberg et al., 1999), essential where comparison between peers is desired. Patient simulations have been used as a method for teaching, as well as a means of assessing "clinical competence". Discussion in this thesis is concerned primarily with the former. Examples of the latter are quite useful in demonstrating the level of sophistication that can be achieved, and the approaches used to define competence and achievement.

Patient simulations involving trained actors provide the closest approximation to the real patient-doctor relationship, as both verbal and non-verbal cues can be accurately depicted in a standardised manner. Actors have been used during research of medical problem-solving by clinicians (Elstein et al., 1978) but this form of simulation is clearly impractical for continuing education purposes, as programs involving actors are relatively expensive to implement and there are a very large number of potential participants.

An early alternative, promulgated for teaching and assessment, is the tabbed item approach. After an initial introduction, a student is expected to choose from a list of possible actions or questions. Adjacent to the action or question is a response that is hidden from view by a tabbed overlay. Removal of the tab is equivalent to taking that action or asking that question and the response will be either the outcome of the action, an answer to the question, or a directive to move to another section in the case booklet (McCarthy and Gonnella, 1967, McGuire and Babbott, 1967). Cases of this type were generally scored according to efficiency, proficiency, errors of omission, errors of commission and a composite index of overall competency. Efficiency was defined as the percentage of choices which are helpful to the patient while, proficiency was defined as the percentage agreement with the criterion group in selecting procedures they regarded as indicated and avoiding those that are contraindicated. Such assessment schemes are still common in modern computer-based simulations (Myers et al., 2001).

Computer-based simulations have become more common for both teaching (Hayes and Lehmann, 1996, Bryce et al., 1997, Bergin and Fors, 2003) and assessment (Clyman and Orr, 1990). Formerly, computer simulations were run on a centralised computer facility but more modern simulations, involving multimedia elements, have been based on desktop systems. With the advent of the world-wide-web came the potential for these cases to return to the central server. However, at the time this project was commenced, the web-based cases (Hayes and Lehmann, 1996) were considerably less sophisticated than those using CD-ROM on desktop-based systems (Bryce et al., 1997). Friedman (1995) has provided an excellent outline of the features that need to be considered when developing a computer-based clinical simulation. When used with the description by Melnick (1990) of the computer based examination (CBX) system developed by the

American National Board of Medical Examiners (NBME) (Clyman and Orr, 1990), a minimum feature set for our own simulation can be developed. Below is a compilation of the ideas and description by Friedman and Melnick. Many of the ideas referred to below, have already been considered or implemented by others.

Single vs. Multiple Encounters

If multiple encounters are desired, there generally needs to be some form of clock for monitoring the progress of time. With the NBME system, participants are given the time and location of the patient at the beginning of the case. Interactions with the patient increment the clock by an appropriate amount of time. For example, if a question is asked, the clock might increment by only a few seconds, while an examination might increment the clock by several minutes. When requests are ordered, results are only available after an appropriate virtual time period has lapsed. Students are able to determine when in the future the patient is to be reviewed. In this way, the student or participant is able to manage the patient over several months of virtual time while spending less than an hour at the computer.

Menu vs. Natural Language Requests for Data

Using natural language creates a higher fidelity simulation and avoids cueing the student, but requires more sophisticated programming. However, some users may also become frustrated with trying to make their requests understood. The Marshall University site at http://medicus.marshall.edu/medicus.htm was able to provide an effective natural language interface over the world-wide-web although the resources required to build even a single case were very large (Hayes and Lehmann, 1996).

The NBME cases, although text based, are very sophisticated. Clinicians are able to write requests by typing the name of the request. The computer recognises

abbreviations, common misspellings, and generic and trade names. Over 2000 tests, procedures and treatments are available but not all are available at any particular location. Thus, before an MRI can be performed the patient might have to be transferred to a facility offering this service. All actions, treatment and investigations are recorded in a medical record format.

Volunteered vs. Requested Information

Simulations can vary significantly in the amount of information that is volunteered by the virtual patient. Some volunteer the entire history while others provide only a few introductory remarks. This variability may be desirable as one may provide minimal information to a general practitioner, reflecting the experience in general practice. In contrast, a summary in the form of a referral letter may be a more pertinent starting point for the allied health worker.

Interpreted vs. Uninterpreted Clinical Information

Again this can vary considerably between programs. One can provide the raw results such as chest x-rays and pathology results without comment or one can provide a text report. Alternatively, there may be some combination of the two in which an initial attempt at interpretation by the user enables comparison with the test report.

Deterministic vs. Probabilistic Progression

With deterministic progression, taking the same action always leads to the same clinical result. In the probabilistic approach each action is associated with a set of probable outcomes and each outcome has a medically realistic probability of occurring. While the latter approach has the potential for a much richer simulation environment, the initial development of the program is potentially far more complicated.

Instructional Intervention vs. Natural Feedback

Instructional intervention comes in at least two forms. The first is to provide constant or intermittent feedback to the student on their progress through the case, compared to some desired optimum. The second method requires the student to relate diagnostic hypotheses to clinical investigations or findings. Friedman does not favour either of these methods. He believes they can be distracting, in that they interrupt the natural flow of a case, and they impose a reasoning framework that is foreign to the student. On the other hand, during primary medical education, one can argue that this framework enables meta-cognitive support that facilitates reflection, and later abstraction. This increases the probability of the student being able to generalize and transfer their knowledge to different situations. Natural feedback involves the realistic progression of the case. If appropriate action is taken, the patient's health improves. In contrast, if inappropriate action is taken, the health of the patient will deteriorate and may necessitate additional action to restore the patient to good health.

Manual vs. Automated Case Development

Manual authoring is the traditional approach, where the case scenario is developed up front from start to finish. Here the outcomes are fixed, no matter what actions are taken. With computer technology, it is now feasible to generate cases from a knowledge base that stores information about the prevalence of disease and the probability of specific findings in the presence of that disease. This enables an outcome to depend probabilistically on the action of the student or doctor.

Scoring

As previously stated, many programs assess participants by determining the errors of omission, errors of commission, and derive indices from these. The NBME CBX (Clyman and Orr, 1990) assigns values to each action and for complex combinations of

actions may specify a sequence. (e.g. an antibiotic should be ordered within a set time of receiving a positive gram stain result, but only after a culture is requested). It should be noted that there have been problems with using computer simulations for assessment. One problem is that of case specificity. Fitzgerald. (1994) has reported on an assessment of the NBME's computer-based clinical simulation examination that is used to assess clinical skills performance for licensing examinations. Medical students were required to complete two CBX cases two weeks prior to a comprehensive clinical assessment (CCA) at the end of their third year of training. The CCA involved a multi-station test of clinical skills including history taking, x-ray interpretation, and electrocardiograph (ECG) interpretation. They found that there was considerable variation in people's performance on different cases and that there was significant variation in the correlation of simulation performance with the CCA scores. They attributed some of this variation to levels of experience. This case specificity is not surprising given the work of Elstein and Shulman (1978) that suggested that clinicians performance was domain specific and probably reflects their knowledge of a particular domain developed through their past experience.

Of more interest is the work published by Goran et al. (1973) which found that paper-based patient management problems lacked validity. He found people did better on paper-based patient management problems than in real clinical practice. While those who did poorly on paper-based patient management problems, did poorly in clinical practice, having done well on paper-based patient management problems, did not predict satisfactory clinical practice.

While the work of Fitzgerald (1994) and of Goran (1973) has highlighted problems of case specificity and apparent lack of validity, much of this work has, as noted by Melnick (1990), taken a dichotomous approach. That is, the practitioner is regarded as being right or wrong, and the number of right answers are summed and compared to the number of wrong answers. An alternative approach is to use the assessment information to improve learning. The work of Engel (1990), Schön (1987), and Anderson (1995b), all suggest that if the learner is able to reflect and or receive feedback on how their actions, both erroneous and meritorious, have influenced their decision making, then learning and performance may be improved.

With regard to the elements of a patient simulation listed above, those elements chosen for inclusion in SIMPRAC are listed in Table 1. The design decisions, as they relate to SIMPRAC, are discussed in more detail in Chapter 3 and Chapter 4.

Table 1: Attributes of a patient simulation used in SIMPRAC.

Attribute	Approach taken in SIMPRAC
Encounters	Multiple encounters are used, with an emphasis on the diagnosis and management of chronic illness.
Requests for data	A combination of approaches is taken. A pseudo natural language approach is taken for history taking and management, but selection from menus or lists is used for physical examination and investigations. The ability to select questions from lists was also included, as an alternative to the natural language approach.

Attribute	Approach taken in SIMPRAC
Volunteered Information	Minimal information is volunteered for medical practitioners and students, reflecting the most common scenario in clinical practice.
Interpreted versus uninterpreted information	All clinical information is uninterpreted. However, learners may get an interpretation on selected examinations, after they provide their own interpretation.
Progression	Progression within SIMPRAC is deterministic.
Feedback	A combination of instructional and natural feedback is provided. The instructional feedback is given at the end of each consultation. However, it is structured in a manner to encourage reflection, going beyond just an indication of being right or wrong.
Case Development	Case development is manual. The architecture for this is described in Chapter 4.
Scoring	Rather than just tallying errors of commission and errors of omission, SIMPRAC asks the user to reflect on the relative importance of each of their actions. That is, to reflect on the weight they gave each of the cues.

2.3 User Modelling for Reflection

Kay (2001) defines a user model as, “any individual, dynamic and explicit model the system holds about the user”. Such models may, “represent the user’s beliefs and preferences as well as other user attributes” (Kay, 2001). Recent research has explored how opening the user model to the user can be used to support reflection and improve

knowledge of the content and structure of a domain (Bull et al., 2003, Morales et al., 1999).

"Open" (Dimitrova, 2003) or "scrutable" (Kay, 2001) learner models enable the user to explore information that the computer has about their own knowledge state. Furthermore, they encourage the user to compare the computer model against their own beliefs about the state of their knowledge. By encouraging this comparison, especially where the computer model differs from the users' own beliefs, these systems represent powerful tools for supporting reflection (Bull et al., 2003). Open learner models can range from the relatively simple to those that are quite sophisticated, where the user is not only able to view or discuss the user model, but modify it in a collaborative process that has the potential to enhance the reflective process (Dimitrova, 2003). Learner models can be represented in a number of ways, including as text (Cimolino et al., 2003), graphs (Kay, 1997), or a combination of text and graphics (Bull et al., 2003)

2.4 Software Evaluation

Alexander, McKenzie, and Geissenger (1998) in their report on educational technology list a number of factors contributing to successful learning outcomes. In particular, they found that successful projects, underwent regular evaluation of both usability and student learning during design and development, and these evaluations informed re-design if required. Conversely, projects that were not successful were frequently only evaluated when the project was complete, involved limited or poor evaluation because of lack of time or resources, or did not evaluate the project in the context of use prior to implementation. These authors promote an integrated approach to evaluation involving formative evaluation during the design and development phases followed by summative

evaluation prior to and during implementation. An abbreviated version of their evaluation approach is shown in Table 2.

Table 2: Approach to evaluation based on the framework of Alexander, McKenzie, and Geissenger 1998.

Activity	Information Gathering	Purpose	Evaluation Questions	Methods
Design	Formative evaluation	To inform decisions made in the design of the technology-based learning program.	• What educational strategies should be used? • Is the user interface appropriate to task and conceptual models of the ideas?	• Review of literature to determine what is known about the way students learn the particular topic. • Walk-through rapid prototype or story boards with potential users and media experts.
Develop	Formative evaluation	To inform decisions made in the development of the technology-based learning program.	• Is the user interaction effective and efficient? • What and how are students learning? • What is the merit of this program?	• Observation, video-taping. User-tracking, interviews with/of users and expert reviewers. • Stimulated recall, critical incidents, think-aloud as target students use program. • Peer review.
Teach / Implement	Summative evaluation	To determine the worth of the technology-based learning program in the context of its use.	• What changes in understanding have students undergone as a result of using the IT project in this context? • Is the package a 'valid' approach to addressing the stated learning need?	• Pre and post tests, questionnaires, interviews. • Expert/peer review.

Nielsen (1993) provides a useful hierarchical model for assessing system acceptability that is summarised in Figure 2. At the top level one can assess the social or practical acceptability of the software system. The former pertains to the social context within which the software is used and in differing contexts may have different acceptability. For example, a computer system that matches health carer attendances with social security payments might be acceptable to administrators for reducing fraudulent claims

but may be totally unacceptable to those concerned with personal privacy. Practical acceptability can be assessed across a number of dimensions including cost, reliability, compatibility with other systems, and usefulness. The latter can in turn be evaluated in terms of utility: does the system achieve its objective, and usability, how easy is the system to use. Lastly, Nielsen provides five axes under which usability can be assessed including:

Learnability	How easy is it for a novice user to learn how to use the system or the expert user to learn infrequently used functions.
Efficiency	Once learned, are expert users able to have high productivity.
Memorability	Is the system easy to remember. Can casual users return to the system without having to go through the process of re-learning how to use it.
Errors	Users make few errors when using the system. That is, they are always able to achieve the intended task. Furthermore, users are able to recover from those errors that they do make.
Satisfaction	This is a subjective assessment and includes how well the user or learner liked the software.

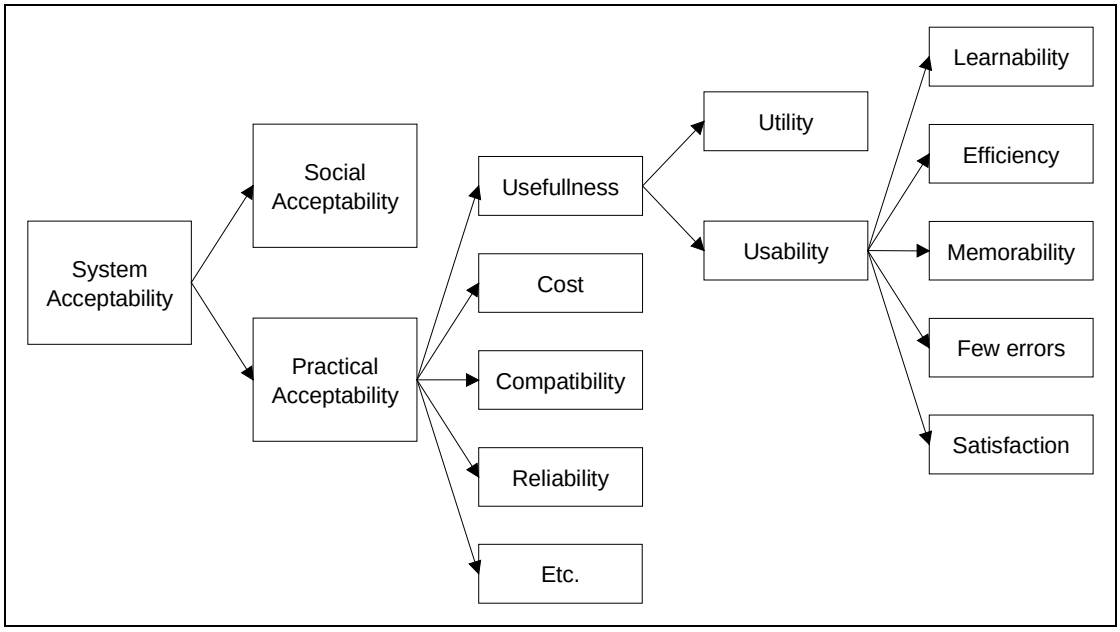

Figure 2: Model of attributes of system acceptability (Nielsen 1993)

The intelligent machines branch multimedia information in mobile environments laboratory at Georgia Tech Research Institute have a different model for evaluating user interfaces that has been adopted by Phillips et. al. (2000) in their, "Handbook for learning centred evaluation of computer-facilitated learning projects in higher education ". They evaluate along 10 axes listed below:

1. *Ease of use:* Perceived facility with which a user interacts with an interactive multimedia program.
2. *Navigation:* Perceived ability to move through the contents of an interactive program in an intentional manner.
3. *Cognitive load:* What demands does the program make on the user in terms of their ability to perceive options, conceptualise choices and make physical actions.
4. *Mapping:* The program's ability to track and graphically represent to the user their path though the program.

5. *Screen design:* How well the user interface complies with screen design principles especially in terms of text, icons, graphs and colour.
6. *Knowledge space compatibility:* How well does the network of concepts and relationships that define the mental schema used in creating the application and case match that of the user. That is, after appropriate exploration, can the user incorporate the knowledge space within their own existing schema.
7. *Information presentation:* Has the information been presented in an understandable form. The information may be present but may not be comprehended.
8. *Media integration:* Do the various media work together to form an effective whole.
9. *Aesthetics:* The artistic beauty or elegance of the program.
10. *Overall functionality:* Aspect related to the perceived utility of the program. Must be judged in relation to specific intended use that exists in the mind of the users.

From the report of Alexander et al. (1998) it is apparent that the resources required to perform a thorough evaluation of educational technology is a major impediment to the development of these projects. Given the complexity resulting from the multiple axes used in the frameworks by Nielsen (1993) and Phillips et al. (2000), it is not surprising that a thorough evaluation involving multiple users on multiple occasions is resource hungry. Nevertheless, according to Nielsen (1993), most interface problems can be identified with as few as 5 users using a simplified think-aloud evaluation.

The think-aloud methodology is a qualitative approach promoted by Nielsen (1993), in which each participant is asked to think-aloud, stating what they were doing and

thinking as they use the software under study. The experimenter merely observes and records the interaction. In cases where the participants appear to spend time thinking but not externalising, the experimenter encourages them to say what they are thinking. The experimenter may also intervene if the user becomes stuck, with this fact being recorded. With the standard form of the think-aloud evaluation, an audio and video recording is made of the interaction, and can be reviewed in detail. With the simplified think-aloud, the observations are recorded on audio tape, or they may just be recorded as handwritten notes.

In keeping with the literature on software development and evaluation, we undertook to run several evaluation cycles to inform the design and development of SIMPRAC. Furthermore, with the limited resources that were available, the use of the simplified think-aloud was the main methodology employed, to highlight those areas needing improvement, when conducting the user evaluations.