Proceedings of WCCCE '98: Western Canadian Conference on Computing Education. Vancouver, 1998.

# A Research-Based Masters Program in the Workplace

*Mildred L. G. Shaw and Brian R. Gaines*
*Software Engineering Research Network*
*University of Calgary*
*Alberta, Canada T2N 1N4*

## Abstract

The Software Engineering Research Network (SERN) is funded by industry and administered by the Industrial Software Engineering Chair at the University of Calgary to support the dissemination of good practice in software engineering. One component of SERN's activities is a thesis-based masters program with a specialization in software engineering targeted on students with industrial experience in full-time employment. This program has dual objectives: to develop highly-qualified personnel; and to encourage industry-based software engineering research with a focus on good practice. The program commenced in September 1996, and this article describes experience in its operation.

## 1 Introduction

The Software Engineering Research Network is a joint venture of the Department of Computer Science and the Department of Electrical and Computer Engineering at the University of Calgary. It is sponsored by Motorola, Computing Devices, Perigon, Northern Telecom, the Government of Alberta and the University of Calgary. Other supporters who are contributing software or student awards include ARIS, IBM, Object Technologies Inc., Gemstone, International Databases Inc., Isotel Research Ltd., and Smart Technologies Inc. Membership in SERN is open to industry and research organizations concerned with applied software engineering. In particular, SERN supports its industrial sponsors through joint projects, a database of experience and best practice accessible through the World Wide Web, requirements engineering and software engineering workflow tools, and an industrial software engineering degree specialization at Masters level.

The primary focus of SERN is on software development as a manufacturing discipline, with objectives of establishing: a centre of excellence in research and development concerned with the knowledge flows and processes underlying the software manufacturing life cycle; research and graduate training activities that will develop highly qualified personnel in industrial software manufacturing; an organizational infrastructure at the University of Calgary supporting a long-term program of university-industry collaborative research on software manufacturing.

The organization of SERN is shown in Figure 1, and involves cross-disciplinary laboratories at the University undertaking fundamental research, joint projects with the Industrial Partners, and maintaining a database of knowledge, experience, problems and issues that can be widely accessed and will be used to manage the activities of the network. SERN operates as a networked organization providing associated university and industry personnel and students with a virtual institution which manages joint projects. There is a two-way flow of knowledge and personnel in which the Industry Partners introduce new problems, the University researchers introduce new technologies, and the relation between the problems and technologies is investigated through joint projects. Graduate students and university personnel work on the projects at the industry

sites, and that industry personnel are seconded to the University for varying periods of time. The network is open to new partners and to the introduction of new technologies from elsewhere. In particular, active links with a range of other relevant institutions are being developed.

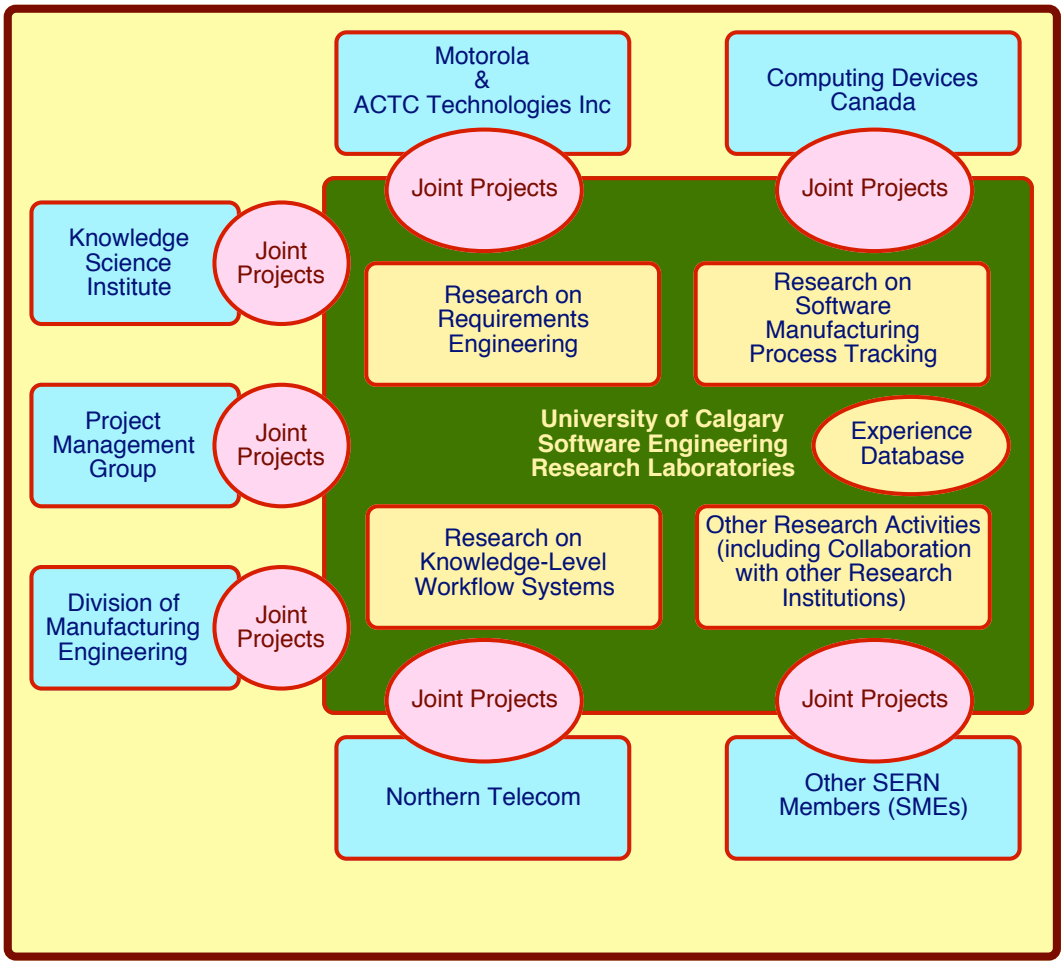


**Figure 1 Software Engineering Research Network**

## 2 Software Engineering Thesis-Based Masters Specialization

The primary dynamic of SERN is industry-university collaboration with an emphasis on real-world research and industrial good practice. The software engineering thesis-based masters specialization was established to encourage students to undertake research projects as part of their full-time industrial employment. On the one hand this enables the university to train students in research methodologies and to develop their capabilities to undertake major research projects and to analyze and communicate the outcomes. On the other it provides a source of research activities to SERN and its members which is firmly grounded in industrial practice and relates to issues of immediate industrial significance.

The learning objectives of the Software Engineering specialization are to enable students to:

- become familiar with research and methodologies in industrially relevant areas of software engineering;
- carry out an applied research project in software engineering.

There are two entry paths to the Software Engineering MSc. It is expected that students with an undergraduate degree in a science discipline will register in the graduate program of the Department of Computer Science, and those with an undergraduate degree in an engineering discipline will register in the graduate program of the Department of Electrical and Computer Engineering. Students applying for entry to the MSc will be expected to have at least one year of experience in software development in a professional environment. Additional experience will be taken into account when assessing a student's academic background and achievements. Background knowledge in C, C++, object-oriented design, and human-computer interaction is preferred.

The University of Calgary no longer has mandatory residency requirements for masters students, and the Departments do not impose residency requirements for the software engineering MSc. Students are currently required to attend the university to take courses but the majority are being organized as 3 hour sessions once a week in the evening so that there is no disruption of normal working hours. A typical pattern for a student in industry is to take one half course a semester (3 hours contact time per week), and plan the thesis research from the beginning as part of the research methods course (which is spread out throughout the project activity). It is realistic to complete an MSc in 2 years while in full-time employment.

## 3 Course Structure

The course component of the MSc consists of 2.5 full course equivalents selected from a specified list of courses which were developed in consultation with the industry partners. There are 6 required quarter courses which form the core of the program. An additional 2 half courses are taken as options from defined lists. The emphasis is on software process management, requirements engineering, project management and research methodology.

The core quarter-courses are:

- SENG 611: Requirements Engineering. The elicitation, modeling, expression and validation of requirements.
- SENG 613: Managing the Software Lifecycle. Further methodologies for requirements engineering. Applications of requirements engineering to the management of the lifecycle of software development from requirements elicitation through analysis, design, coding, testing, enhancement and reuse. Prerequisite SENG 611.
- SENG 621: Software Process Management. Analysis of software process maturity models from repeatability, though definition and management to optimization.
- SENG 623: Software Quality Management. Analysis of applicable quality measures for software processes, the role of reviews, metrics, and tools for the automatic derivation of quantitative measures. Prerequisite SENG 621.
- SENG 691: Managing the Software Engineering Research Process. Research methods for software engineering in an industrial setting. (Students register for course in semester when they will complete it).
- SENG 693: Trends in Software Engineering. Recent developments in various areas of software engineering. (Students register for course in semester when they will complete it).

In addition students select the equivalent of two half-courses from the following options:

- MOHR 621: Organizational Management

- MOHR 691: Project Human Resources and Organizational Effectiveness
- ENCI/POEN 691: Fundamentals of Project Management
- SENG 609.01 Software Process Modelling: Approaches & Languages. The motivation of software process modelling and its advantages. An introduction to and comparison of several process modelling languages. Q(3-1)
- SENG 609.02: Software Process Modelling: Enactment/Execution & Tools. Several tools for modelling software development processes are introduced. Methods and techniques for operationalising process models will be discussed. Prerequisite SENG 609.01.
- SENG 609.03: Object Theory.
- SENG 609.04: Software Design Patterns.
- SENG 609.05: Graphical User Interfaces: Design and Usability.
- SENG 609.06: Special Topics in Human Computer Interaction.
- SENG 609.07: Introduction to Software Reuse.
- SENG 609.08: Domain Analysis and Engineering.
- ENEL 619.02: Object Oriented Programming.
- ENEL 619.94: Development of Small Software Projects I.
- ENEL 619.95: Development of Small Software Projects II. Prerequisite ENEL 619.94
- ENCI 697: Project Planning and Control. Prerequisite ENCI 691
- ENMF 619.04: Object-Oriented Analysis and Design

Students are strongly advised to take at least one of the management half courses: MOHR 621, MOHR 691 or ENCI 691.

## 4 Learning Environment

The learning environment for the courses is unconventional and reflects the industrial experience of the students. After the first background lecture the students take over responsibility for presentations on the various topics in the course. The instructors' role is that of facilitator managing a process of debate and exploration rather than attempting to be an authority in the domain. In addition, an experienced industry manager participates as a second facilitator, typically entering a debate when it has come to a dead end or become unbalanced. The overall aim is to provide a supportive and nurturing learning environment in which experience and knowledge can be shared and ignorance displayed and errors made without censure but with ready access to diagnostic help. It becomes clear to the students that, while there are no easy answers to the core questions of industrial practice in software engineering, there are many useful perspectives and that simplistic answers generally have very limited applicability. An up-to-date specialized library is maintained specifically for this program and students rapidly become fluent in the latest software engineering literature, initially through background research for their presentations and ultimately through searches for material relevant to their research topic.

An overlapping cohort model has been found to be highly supportive of the processes outlined above. There are two major intakes into the program, one in September and one in January, of approximately equal size. The core courses are taught only once a year so that on each course there is a roughly equal mix of new students and those who have already taken courses and are

experienced in the learning environment. Students work in teams to develop presentations and those with prior experience of doing this essentially mentor those who are new to it. This has proved very effective in bringing new students rapidly into the learning culture of the program.

As in our undergraduate software engineering course (Shaw and Gaines, 1997c) student assignments are submitted on the World Wide Web making them accessible to others. A list server is used for continuing discussion of the course topics outside the class environment. These two features have made it possible for students whose companies allocate them to work in other locations, or who change jobs to companies in other locations world-wide, to continue to participate in the courses. Remote students who have already come to know their colleagues can continue to present and share material through the web and participate in discussion through the list server. We are currently beginning to address the challenge of making more material formally available to remote students and this early experience of students becoming remote through circumstances beyond their control, but maintaining effective ongoing participation, is proving very valuable in designing for the distance mode.

## 5 Student Experience

The required quarter courses on Software Process Management (621), and Software Quality Management (623) are designed such that the first one encourages reading of the literature, and learning about the topic, while the second looks more at the application of the topic. For instance, 623 assigns the exercise:

> Your company has begun their software process improvement (SPI) efforts by adopting the SEI CMM as their software process improvement framework. Your company has been currently assessed at Level 1 and is putting into place the plans to get to Level 2. SPI teams have been created to implement each of the Key Process Areas. In addition, your company has decided that it could benefit from Level 3 Key Process Area Peer Reviews and has decided to implement that as well.
>
> You have been assigned to a team to create a metrics program for a specific Key Process Area (KPA). Your responsibilities include :
>
> • the metric goals for the KPA.
>
> • Identifying the types of metrics necessary to measure the KPA and its requirements.
>
> • Setting up the actual metrics program which will identify such things as:
>  • who will collect the metrics;
>  • when will they be collected;
>  • how will they be used; and
>  • what kind of analysis and reports will be done.
>
> This is not a complete list. There may be other activities that may need to be performed.
>
> • Present the metrics program to the rest of the company using sample data to illustrate the metrics, how they will be used, etc.
>
> • Document the program in a metrics plan.

The students are divided into small groups and allocated a key process area to research the problem, decide on how it should be tackled, and present their findings to the rest of the class. It is interesting to note that although the students have been reluctant to disagree with each other in previous presentations, here they react very much like a regular industry group given the same problem — questioning, arguing and criticizing each other to tease out all the possible ramifications of the issues. It is in such situations that the industry manager has a lot to contribute as he discusses with the students the practical applications of this in the workplace of individuals in the class.

The students put all their work on the web, including the comments from others. For example, the pair who took the KPA of Software Quality Assurance included the section "Feedback from Others":

> Maybe because we took a very different way to present the SQA Metric Plan from the way others did, some people may have difficulty to follow our presentation, or have some confusion why we present in that way. Here I would like to give a few more words about our presentation and discuss some feedback from others concerning our presentation.
>
> The techniques that we adopted in our presentation were aligned with the main purpose of SQA: to improve both software development process and products. So the presentation had two main threads: how can SQA Metric Plan serves the purpose of improving software process quality, and how can SQA Metric Plan serves the purpose of improving software products quality.
>
> Following these two threads we elicited what we want to achieve and how through SQA Metrics Plan. Basically we cover the following aspects that most concerns come from:
>
> **What**: To address certain sub-area, or aspect in software products or process that may be the concern of management, or that may affect quality. For example, defects, non-conformance, software configuration management process, etc.
>
> **Why**: To give reasons why SQA program or management think the certain aspect is a concern, maybe it will affect the software product quality, cost too much, or schedule may be delayed, etc.
>
> **Who**: To address who will be affected by the plan. This include those groups that will be measured and those who will receive the reports.
>
> **How**: To present the metrics and the analysis methods that can be used to present the information embedded in those metrics. We thought that data analysis is the most important part of a metric plan. After data has been gathered, we must analyze it and get information out of it. Useful information usually comes from many metrics. That is why we categorize data into separate groups and analyze them together, and get needed information.
>
> Hereafter is some feedback we received after the presentation, hopefully this document along with SQA Metrics Plan will give a satisfactory answers to them.
>
> 1. Who are the audience of the program?
>
> The main purpose of SQA program is to give management assurance that everything in the project is going well. If something goes wrong, SQA should detect what is wrong and report to management so that corrective actions can be taken to minimize the impact. To make SQA program work, the support of developers is very important. Without their buy-in, SQA program will certainly go nowhere. The nonconformance analysis is to both developers and managers so that standards and procedures are followed.
>
> 2. What are roles and tasks of SQA measurement program?
>
> I think this one is being stated very clearly in SQA Metric Plan. Maybe because we have too little time in the presentation so we hadn't made this very clear. Please do check the metric plan if you have any questions.

Again, in the courses Requirements Engineering (611) and Managing the Software Lifecycle (613) a similar model is followed. In the first course the students look into various techniques for eliciting and recording requirements such as interviewing, concept maps, repertory grids, and affinity analysis. We than find that they use these techniques, not only in the content of what they are doing, but in presenting their work. One group presented an interactive Techniques Map with the following description:

> It is believed that a 'techniques map' as illustrated in Figure 1 presents a nice outline of the Requirements Engineering techniques that will be addressed in this report. It not only illustrates the techniques that will be discussed but also shows the possible combinations between them. This map will be used as a reference throughout this report. The reader is encouraged to point and click on any technique in order to learn about a short description and about its strengths and weaknesses. Also, pointing to any labeled relationship in the map enables the reader to learn about possible combinations between these techniques.

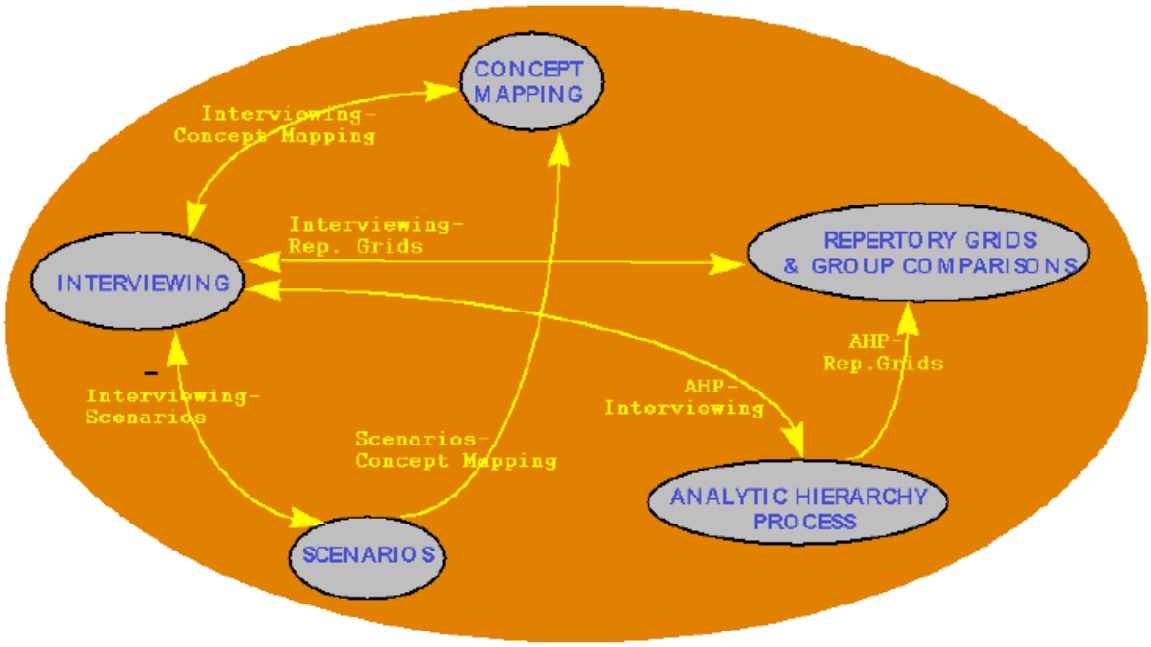


**Figure 1 The 'techniques map' illustrating an outline of the techniques discussed and the relationships among them**

Another group used repertory grids to compare the techniques used in their report:

> A repertory grid was used to compare various Requirements Engineering techniques. The techniques covered in this course were used as elements. Constructs consisted of quantitative vs. qualitative, understandability, scalability, usability, and time to execute.
>
> A FOCUS hierarchical structure and PrinCom analysis of the grid are shown. The results show a similarity between Repertory Grids and Group Comparisons, which is expected since the latter technique is an extension of the first. The results also show a similarity between Interviewing and Use-Case Analysis. This may be due to a common involvement with users and a shared goal of acquiring information about how they work. Analytic Hierarchy Process is different from the other techniques. AHP is quantitative whereas most of the other approaches are primarily qualitative, and it is the least scaleable technique.
>
> This comparison could be further enhanced by including more constructs to differentiate between Interviewing and Use-Case Analysis. The grid could also be exchanged with another group for comparison.

The principal components analysis is shown in Figure 2.

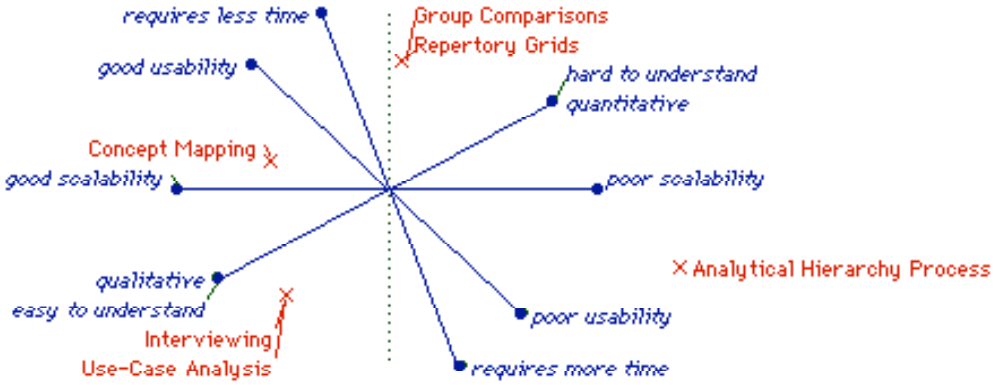


**Figure 2 The repertory grid from Group 2 of the requirement engineering techniques discussed in their report.**

In 613, each group also had to produce a practical exercise for the other groups to do in class, to make sure they had a full understanding of the topic being presented. The group looking at Checkland's (Checkland, 1981) Soft System Methodology (SSM) set the following exercise:

> **Objective**: This exercise is designed to help you to have a good understanding of what is SSM and how to use it in practice. Remember: There is no 'right' or 'wrong' answer for this exercise, what you need to present is your thinking about the problem situation and a relevant system model for your thinking. There are five requirements for this exercise, do as much as you can within 20 minutes. If you need hints for these questions, please refer the appendix.
>
> **Problem situation:** A local Calgary choral society always has the difficulty in obtaining nominations for its officers and committee and attracting people to participation in choral related works. As it is a performing society a number of non-choral tasks must be managed. How could this difficulty be addressed and examined?
>
> **Requirement 1**: Use a rich picture to address the "Problem Situation Expressed" stage in SSM for this problem situation.
>
> **Requirement 2**: Define a "Root Definition" for this problem situation.
>
> **Requirement 3:** Do a CATWOE analysis on your Root Definition.
>
> **Requirement 4**: Produce a "Conceptual Model" based on your Root Definition. (activities such as identify needs of local community, attract membership, general funds, attract audiences could be used for this problem situation)
>
> **Requirement 5**: Use the "Tabular Display" comparison method to perform the comparison between the real world and intellectual world.

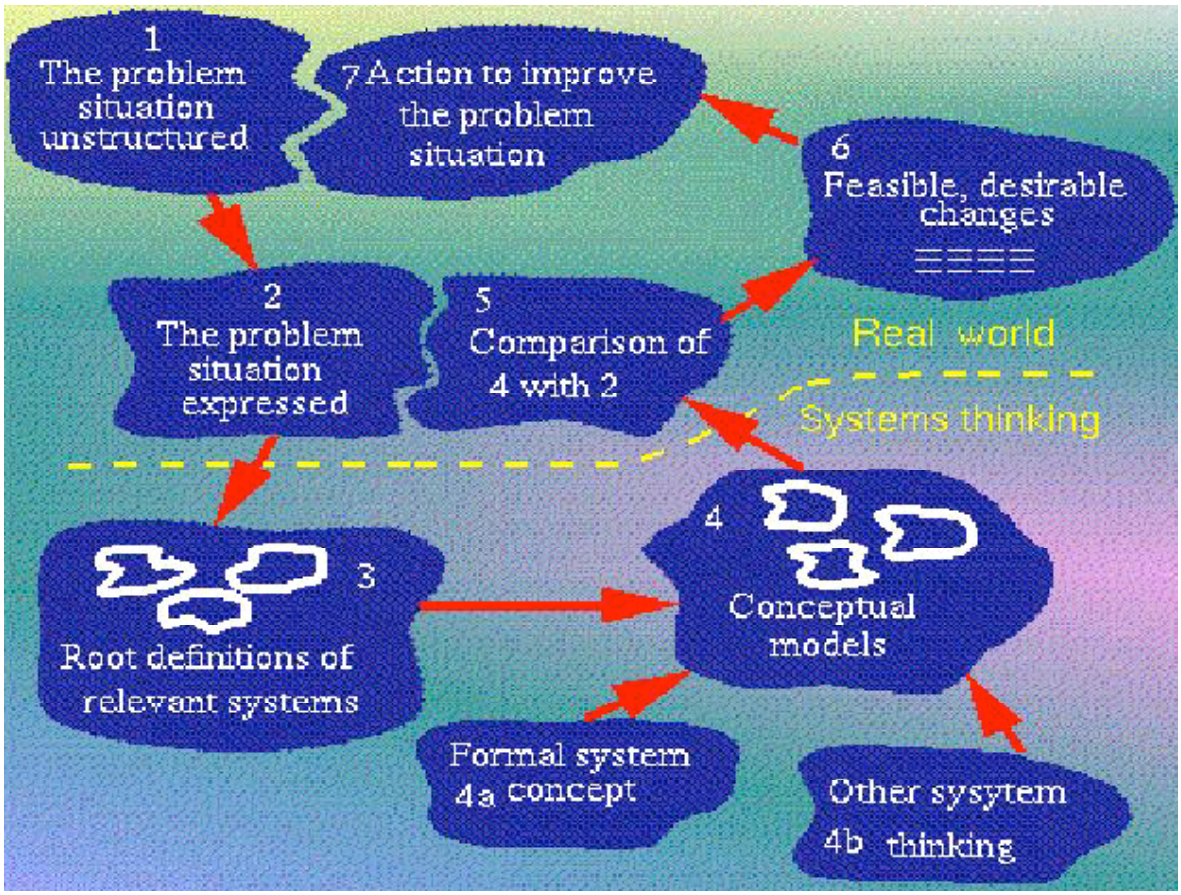


**Figure 3 Soft Systems Methodology concept map**

## 6 Research Methods Course

The research for the thesis is expected to be based on the ongoing activities in the workplace. It is expected that students will be employed in the software industry, and will work on their projects at the industry sites. This has made it important to address intellectual property issues from the outset. For example, in an experiment in which a particular approach to process improvement is investigated and software metrics are kept for a particular development, the final thesis need give no details relating to confidential information but would present the characteristics of the software being developed, the experience with process improvement and the plots of the relevant metrics. A similar approach can be taken for other topics and students are asked to develop a research plan that they can share with their employers from the outset so that difficulties over intellectual property rights are minimized.

Since the primary deliverable for a research degree is the thesis and the degree will be awarded principally through an examination committee assessing the thesis and the student's defence of it, a research methods course is essential to the program. The majority of students already have substantial experience of software development in industry, and some have come into contact with research issues, but virtually none of them have any experience in developing a research program, setting up experiments, collecting data, analyzing it appropriately, drawing appropriate conclusions and presenting all this effectively in a substantial, coherent document.

We have found in the past that the basics of research methods are best assimilated when students can perceive a need in their own experience or that of others, and hence a case-based approach has been adopted for this course with the cases being the ongoing research of students in the program. The course commences with the presentation and detailed analysis of the concept map for a research program shown in Figure 4.

The objectives defined on the left follow a logical sequence and also correspond to the typical chapter structure of a computer science masters thesis. The required activities to carry out these objectives provide the students with the framework for a research program. The students develop a concrete instance of this concept map for their own research program, and can do this in an active concept mapping tool (Gaines and Shaw, 1995b) that also links to material on the World Wide Web (Gaines and Shaw, 1995a; Gaines and Shaw, 1995c).

The assessment for the research methods course aims to develop the student's thesis:

**I Plan 1: 10%:** A statement of your current progress toward choosing a thesis topic including a list of possible topics. For each possible topic include:

1. possible supervisor at U of C
2. possible supervisor at work
3. a plan for how you propose to complete the research and thesis including milestones and schedule
4. a short list of the fundamental reading material for the research area.

**II Plan 2: 20%:** A 4000 word document on the web to include:

1. a short introduction stating the main topic area and importance of the topic, and background
2. aim and objectives of the work
3. current plan as in (3) above
4. revised reading list
5. concept map completed for your topic (see below)

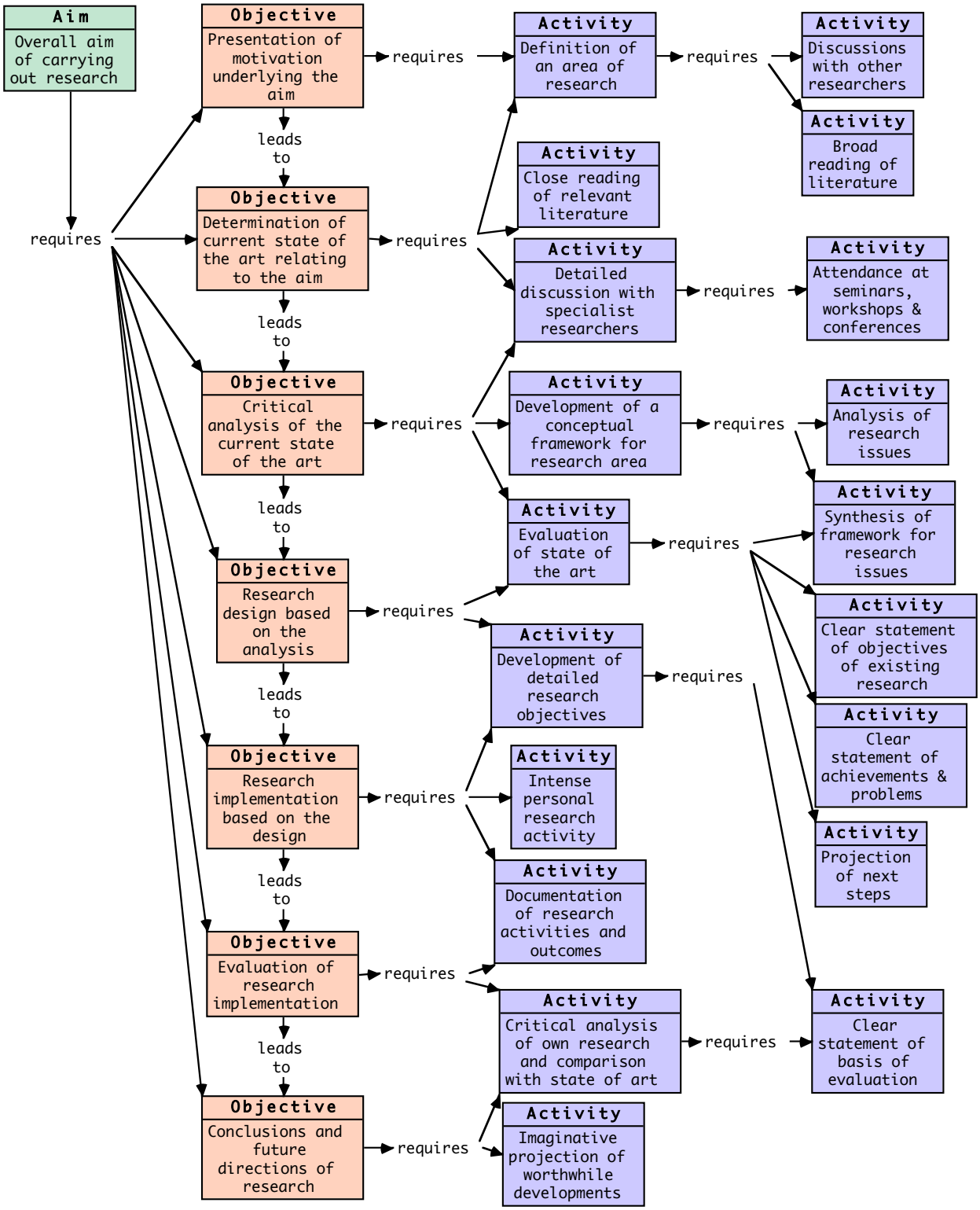


**Figure 4 Concept map for a research program**

**III Presentation 1: 10%:** Presentation of research ideas. To be done after Plan 2 to expose your ideas and get feedback. 20 mins + 10 mins discussion

**IV Literature Survey. 20%:** A 4000 word document on the web surveying the current state of the art relating to the aim, including extended reference section in standard format (check out one of our papers as a model).

**V Critical Analysis. 20%:** A word document on the web of a critical analysis of the current state of the art from IV above, leading to a research design based on this analysis. Include a short section on how you plan to implement your research design and a section on how you will evaluate your research implementation. Include also milestones and schedule for completion.

**VI Presentation 1I: 10%:** Presentation after V (critical analysis). 20 mins + 10 mins discussion

**VII Participation 10%:** Participation in discussion of other research presentations both in class and on the web.

## 7 Conclusions

This article has described some of the main features of a thesis-based masters program with a specialization in software engineering targeted on students with industrial experience in full-time employment. The program is one activity of the Software Engineering Research Network which is funded by industry and administered by the Industrial Software Engineering Chair at the University of Calgary to support the dissemination of good practice in software engineering.

The program builds on our objectives and experience in designing new learning environments based on facilitating collaborative learner interactions in what we have termed a *learning web* (Gaines and Shaw, 1997; Shaw and Gaines, 1997a; Shaw and Gaines, 1997b; Shaw and Gaines, 1997c). The learning web is a systemic approach to the modeling and support of knowledge processes in a learning society. It models various aspects of education as a means of disseminating expertise to create continuing and large-scale resources from short-lived and limited-capacity individual agents. The role of information technology is modeled as one of providing knowledge support systems that expedite the processes of knowledge formation and dissemination.

The learning web approach has been introduced into existing university courses and doing so provides the basis for its implementation on a larger scale transcending institutional boundaries. However, an essential prerequisite is the restructuring of existing pedagogical approaches, de-emphasizing received wisdom and authority of the instructor, and emphasizing collaborative learning and meta-reflection on all aspects of the learning process itself. The instructor becomes a facilitator of students learning to learn, and eventually that process of learning to learn must itself become an overt topic for discussion by students fully participating in its management. In the same way that in the current educational system the home prepares for the school and the school prepares for the university, the university must come to see itself as preparing students for a role in a learning society in which they have come to understand and manage their own processes of lifelong learning.

We are now involved in extending the approach reported in this paper to support professional development and lifelong learning in software engineering. This entails further modularization of course material, more extensive coverage of topics at senior undergraduate and graduate level, and more effective use of Internet technology to support workplace and distance learning. We see the need to partner with other institutions to do this cost-effectively and to sustain excellence in the program, and hope that this article will lead to new partnerships.

## Acknowledgments

Financial assistance for this work has been made available by Motorola, Computing Devices, Perigon, Northern Telecom, the Government of Alberta and the University of Calgary. Thanks are also due to the class of 1997 for the examples used to illustrate this article.

## References

Checkland, P. (1981). **Systems Thinking, Systems Practice**. Chichester, UK, Wiley.

Gaines, B.R. and Shaw, M.L.G. (1995a). Collaboration through concept maps. Schnase, J.L. and Cunnius, E.L., Ed. **Proceedings of CSCL95: Computer Support for Collaborative Learning**. pp.135-138. Mahwah, New Jersey, Lawrence Erlbaum.

Gaines, B.R. and Shaw, M.L.G. (1995b). Concept maps as hypermedia components. **International Journal Human-Computer Studies 43**(3) 323-361.

Gaines, B.R. and Shaw, M.L.G. (1995c). WebMap: concept mapping on the web. **World Wide Web Journal 1**(1) 171-183.

Gaines, B.R. and Shaw, M.L.G. (1997). Institutional transformation to a learning web. **Proceedings of ED-TELECOM 97 : World Conference on Educational Telecommunications**. pp.384-389. Charlottesville, VA, Association for the Advancement of Computing in Education.

Shaw, M.L.G. and Gaines, B.R. (1997a). Collaboration and negotiation through the learning web. **Proceedings of ED-TELECOM 97 : World Conference on Educational Telecommunications**. pp.1390-1391. Charlottesville, VA, Association for the Advancement of Computing in Education.

Shaw, M.L.G. and Gaines, B.R. (1997b). The learning web in the workplace. **Proceedings of ED-TELECOM 97 : World Conference on Educational Telecommunications**. pp.461-466. Charlottesville, VA, Association for the Advancement of Computing in Education.

Shaw, M.L.G. and Gaines, B.R. (1997c). Using the web to support cooperative learning in software engineering. Tsiknis, G.K., Ed. **Proceeedings of WCCCE'97: 2nd Western Canadian Conference on Computing Education**. pp.63-72. Nanaimo, BC, Malaspina College.