Preparing Computer Science Students for Global Software Development

Olly Gotel[1], Christelle Scharff[2], Sopheap Seng[3]

***Abstract* – This paper describes an innovative study undertaken in an undergraduate capstone software engineering course to give students exposure to the realities of global software development. In this study, students from Pace University in New York worked as part of an extended team with students from the Institute of Technology of Cambodia, developing software for Cambodian customers. This paper explains the goals of the study, describes the teams and their projects, and emphasizes the required logistics for a study of this nature. The findings are discussed, and some lessons for computer science education and global software development are provided. The paper finishes with a brief account of our continuing work in the area.**

Index Terms – Global Software Development, Requirements, Softer Skills, Team Work.

INTRODUCTION

To leverage the advantages of global working, it is understandable that organizations seek to recruit those graduates with a set of desired 'softer' skills that can augment their technical ability and thereby enable them to function as a competent member of a globally distributed and multi-disciplinary team. As educators, we need to respond to this situation by preparing students for the roles that they are likely to play in this global marketplace [1].

In spring 2005, we changed the nature of the group project in the capstone software engineering course for undergraduate computer science students at Pace University. These students worked with a group of students from the Institute of Technology of Cambodia (ITC) to form a team of students from both countries. We organized the projects so that: (a) the Cambodian students acted as customers and end-users – they knew the problem the proposed system was to tackle, the environment it was to operate in and had the authority to accept the work of the developers (or not); and (b) the Pace University students acted as developers – it was their responsibility to 'capture' the requirements for the system, propose design options, build the selected design and test the eventual system, while also handling requirements changes. Software development is an iterative and incremental process that relies on good communications and regular feedback. We therefore controlled and monitored aspects of the project relating to cultural differences, time and space complications, and choice of communications tool for sharing project artefacts and managing project activities.

In a simulated global software development setting like this, we suggest that students can acquire the technical knowledge and the 'softer' skills (e.g. leadership, mentoring, negotiation and communication skills) that are needed to thrive in a globally distributed and multi-cultural working environment. The experience was designed to raise student awareness of the issues associated with working in this manner and to provide early (i.e. safe) exposure to successful, and not so successful, working practices and tools. An additional objective was to provide students with a more balanced and first-hand view of the advantages, disadvantages and potential of global software development than is commonly portrayed in the popular media.

This paper describes the preparations that are required to set up a student project of this nature, discusses the key findings from this study, and suggests a number of wider lessons for computer science education and global software development. The paper also highlights details of our continuing work aimed at extending this educational model to incorporate additional worldwide institutions.

GLOBAL SOFTWARE DEVELOPMENT

'Offshore outsourcing' are two words that increasingly feature in computer science education forums. The three most recent SIGCSE conferences have included panel sessions on information technology offshore outsourcing [2, 3, 4]. The 2004 panel provided a status report on the current situation, and considered the likely future trend and its impact on educational institutions within the US. The 2005 panel took this one step further by examining the jobs and skills most affected by offshore outsourcing, exploring the impact on computer science and information systems curricula, and discussing the components of an initial response. The 2006 panel, drawing on a recent ACM report that considered the trends and forces behind the globalization and offshoring of software [1], began to look in more detail at the educational realities and some necessary curriculum responses.

The term 'offshore outsourcing' refers to sourcing work and services in another country. This is generally undertaken to realize the business benefits that arise from either employing cheaper labor or from being able to operate an around-the-clock operation. There are arguably other benefits, such as the perceived improvement in quality that can come

[1] Olly Gotel, Pace University, Seidenberg School of Computer Science, One Pace Plaza, New York, NY 10038, USA, ogotel@pace.edu.
[2] Christelle Scharff, Pace University, Seidenberg School of Computer Science, One Pace Plaza, New York, NY 10038, USA, cscharff@pace.edu.
[3] Sopheap Seng, Institute of Technology of Cambodia, BP 86, Blvd Pochentong, Phnom Penh, Cambodia, sophep.seng@itc.edu.kh.

1-4244-0257-3/06/$20.00 © 2006 IEEE **October 28 – 31, 2006, San Diego, CA**
36th ASEE/IEEE Frontiers in Education Conference

from specialization, but such claims can be less easy to substantiate [1]. Most organizations are concerned with their sourcing strategy, not with offshore outsourcing per se. With the growth of offshore insourcing, where global organizations employ their own workforce in a cheaper country, the issue for organizations is one of determining the best source for whatever they want to do, while balancing aspects such as risk, cost and agility of response.

Whether the impact of globalization has directly led to declining computer science enrolment or not is debatable. While we as educators traditionally focus on the necessary technical skills to equip our students for the workplace, we possibly leave some of the 'softer' skills that will differentiate a rounded and employable student from a non-employable student purely to chance. Importantly, we put students in teams that may bear little relation to the sorts of team they may be working in when they graduate. What is not debatable is that we need to prepare computer science students for distributed working in the global workplace.

RESEARCH AND TEACHING OBJECTIVES

The motivation for our study was to give computer science students some experience of the way in which they may be working when they graduate and for faculty to learn about how to integrate such experiences into the curriculum. The aim was to provide students with a real-life offshore outsourcing software development experience by getting them to collaborate with students from an unfamiliar country.

The capstone software engineering course generally brings together a group of three to four Pace University students to work on a small software development project. We decided to change this model from a co-located team to a team distributed across two time zones, uniting US students with Cambodian students from the ITC to work together as part of a global team of five to seven students. We chose Cambodia because the issues and barriers facing global working are more pronounced than in an established service providing country such as India – in addition to time zones, there are issues of language differences (Cambodian students speak Khmer, French, and little English), cultural differences, intermittent electricity, aging technology and limited Internet connectivity. ITC is a premier semi-public higher education school in Phnom Penh and currently the leading engineering school.

The objectives for the study were twofold.

- To examine the nature of the communication frameworks, processes and technologies required to support a reasonably sized distributed group project of this nature, noting that one of the parties resides in a developing country with an unreliable communications infrastructure. The intention was to gather data throughout the study to determine the predominant model and pattern of communication employed by students and to compare the findings with two control projects that were set up to be non-distributed in nature.
- To learn more about the necessary 'softer' skills that students require if they are to be effective members of a global team. The intention was to determine how best to integrate support for such skill development into computer science classes. This aspect is not discussed in this paper and will be the subject of a future paper.

SETTING AND PREPARATIONS

The spring 2005 software engineering capstone course at Pace University intended to introduce students to the challenges of global software development in a unique way. This required aligning and integrating with one of the specialist software engineering concentration courses at ITC.

Course Structure and Teaching Methodology

The Pace University software engineering course spanned a fourteen week period, three weeks of which were not part of the ITC course due to different academics calendars at the two institutions. During these three weeks, Cambodian students used their free time to participate in the project. The Pace University course consisted of lectures, tool presentations, labs, and discussions on relevant topics in global software development and project management. The course topics covered software processes, requirements, design and testing.

The project milestones were organized around one week for initialization of the project and team bonding, five weeks for requirements, three weeks for design, three weeks for coding/construction and two weeks for testing. The software engineering process model that was used was a loose waterfall model with iteration and feedback cycles, mostly for instructor control and visibility. In the requirement phase, requirements were captured using questionnaires and chat discussions. Templates were designed for the requirements, design and testing documents to help the students standardize their work.

Arrangements Prior to the Semester

The professors involved in this study discussed the basic idea in New York City in the summer of 2004 during a visit of the Cambodian professor. This was followed by many discussions on the countries, cultures, institutions, educational systems, academic calendars, students' background and the Internet access in Cambodia. The first important element that had to be taken into account was the twelve hours time difference between Cambodia and the US. Another important element was Internet access; Cambodian students only had day time access to the Internet from the ITC labs, from the *Agence Universitaire de la Francophonie* on a weekly basis, and from widespread cyber-cafes for $1 an hour (an expensive proposition for Cambodians knowing that the monthly average salary is around $60). The instructors had to design their syllabi in collaboration and decide on the use of communication tools (e.g. Yahoo! Groups, Yahoo! IM), CASE tools (e.g. Eclipse), and the software engineering process and communication protocols to be followed.

Student Population, Teams and Projects

In the spring of 2005 the software engineering class at Pace University comprised nineteen students and the class at ITC comprised thirteen students. The study therefore consisted of five global teams composed of three to four students from the US acting as developers and two to three Cambodian students acting as customers. Note the important reversal of

conventional onshore/offshore roles in this scenario. We explicitly designed the study this way to give the US students the opportunity to find out what it would be like to be on the receiving (and often maligned) side of the offshore/onshore scenario. Students were free to choose their local teams, but global team partners were assigned by the instructors.

Two ITC–specific themes were proposed as candidate projects. These intended to support two paper-based activities within ITC. The Cambodian students developed these themes to create their own more specific project proposals.

- Two student projects were proposed to design and develop an ITC Schedule Builder and Classroom Assignment System. The students decided to design a system that would generate trimester schedules and room assignments with respect to a set of courses to be taught that trimester, based on room availabilities and a professor's preferences (in terms of teaching time and room choice).
- Three student projects were proposed to design and develop an ITC Student Information System. The students decided to design a system that would manage student registration and provide a way to view a student's information on courses, grades, attendance records, etc.
- Three control projects were also set-up to compare the communication on the distributed projects with those of co-located projects. These two projects only involved Pace University students and the task was to work with personnel from the Pace University Housing and Residential Life services to create an events application.

Student Roles and Responsibilities

The US students were to act as developers. Their responsibilities were to capture the requirements and produce a requirement specification to be approved by the customer, propose design options organized in a design document, implement the software and test it, while concurrently handling requirements changes and integrating feedback. At the end of the semester the US students had to deliver the software to their customers. Additionally, US students were required to maintain a web page for the project, report regularly on the difficulties and problems arising with the ITC team, answer a weekly questionnaire concerning communications, archive all emails and chats, and document their experience with the software engineering process and communication protocol they followed. The US students prepared a mid-semester and final presentation where they presented their offshore experience and demonstrated their software to other students and faculty within Pace University. This was video taped for the Cambodian students.

The Cambodian students were to act as customers. Their responsibilities were to describe the software they wanted to be built and the context in which it was to operate. They had to also review and give feedback on requirements, design and testing documents, and periodically test the software and submit bug reports. Their responsibilities also included reporting on the difficulties and problems arising with the Pace University team and answering a weekly questionnaire about their communication activity. At the end of the semester, the Cambodian students had to decide whether to accept or reject the software that was built for them. The Cambodian students also prepared a presentation on their experience and demonstrated the software to their school.

Monitoring Student Work

In this study, we were interested in monitoring how distributed students interact and communicate on projects, determining the software engineering practices that work or do not work in a distributed setting, and examining the link between communications activity with the process undertaken, stages and milestones of a project, and the quality of the end product.

The students' work was monitored by enforcing fixed deadlines with review, feedback and iteration. The US students submitted several draft versions of requirements, design and testing documents that were annotated for consistency and improvement by the Cambodian students and professors. The software was delivered by increments to be tested by the Cambodian students on a weekly basis.

The local and global teams used emails, chats and face-to-face meetings for communicating. Emails were archived in Yahoo! Groups and chat transcripts were saved and posted on the students' web pages. Communication activity was recorded using an online weekly questionnaire administered in questionpro.com that the Cambodian and US students responded to during class time. The online survey examined (on a weekly basis) when the chat communications were carried out, how many emails were exchanged, how many face-to-face meetings were conducted, between whom the communications took place (between individuals, or within the local team or global team), the main topics covered during the communications (e.g. team bonding, management, requirements, design, testing, coding, quality, web page, etc.), whether the communications were more about planning work, checking work or a mixture of both, and whether the communications were useful or not.

Each team also maintained a blog (in blogger.com) containing information about their activities, discoveries and difficulties, as well as their reflection about the software engineering process they were following, the communication process they were using and the experience of global software development they were going through.

The US students were interviewed regularly during class time by the professor teaching the course and by an external evaluator to detect the problems and difficulties and inform further development of the study. The Cambodian students were interviewed during scheduled conference chats involving the Cambodian students and Cambodian and US professors.

Redundancy was introduced in monitoring the communications of the students. This was for cross-checking purposes and future analysis of the communications.

FINDINGS – ABOUT SOFTWARE DEVELOPMENT

Students encountered a mix of issues not unlike those commonly experienced in software development projects. Most of the problems were typical requirements engineering problems and not specific to the fact that the teams were distributed. However, we suggest that running a distributed

project of this nature makes these problems more pronounced for students. It is very easy for students to either ignore or work around typical requirements issues when they select their own project and are the owners of its requirements. Where a third-party customer owns the requirements (be they onshore or offshore), the students are accountable to a party other than themselves for the direction and quality of their work. It is not so easy to ignore emerging issues without an accompanying deterioration in 'whole team' relations when multiple perspectives need to be taken into account.

A number of the software development issues that students experienced were more related to the distributed nature of the project. These issues were associated with project management activities and 'whole team' collaboration, communication, coordination and awareness.

Making Assumptions

The study was designed to present students with two distinct educational contexts. The US and Cambodian educational systems are quite different in their philosophies. When a student has only had experience of one context, it is all too easy to assume that this is the only logical way to do things. The nature of the projects required students to explore the context and constraints of the problems they were tackling and ask far more searching questions than they traditionally would. In the process, students learned first-hand about how easy it is to make assumptions about what stakeholders need based on their own personal experiences. This was particularly evident in the ITC student information system projects where students discovered only mid-semester that the grades of the Cambodian students were out of 20, and not out of 100 or according to the US letter grading system. It is difficult for students to appreciate how many assumptions they typically make, let alone understand the consequence of making theses assumptions. Students in this study required more extensive discussions about requirements than in the past to understand the peculiarities of the customer domain.

Dealing with Changing Requirements

When students work on a project of their choosing with no external customer, it is easy to stabilize on the first set of requirements they write. In this study, the students had to interact with customers who demanded continual requirements changes and new functionality. Sometimes this was to address misunderstandings arising from false assumptions (above) and language problems. This experience forced students to learn about the inevitability of dealing with customers who change their minds, and to learn about the required working practices to deal with this situation, such as version control and requirements management. Requirements iteration was the main reason for most of the communication in each project.

Availability of Customers

Due to the twelve hour time difference and limited access to communications infrastructure in Cambodia, the US students had to learn that they cannot always get a timely response to their questions. Both teams had to be disciplined and organized in planning their interactions. This provided students with exposure to how important it is to assess the risks on a project and allow for contingency when planning across time zones, including how important it is to factor in country-specific and religion-specific vacations and events.

The US students started the project by expecting the Cambodian students to work to their schedule. However, the Cambodian students had a heavier class load than the US students (around 25 hours of class a week), with more serious penalties for failing; failing a class means repeating a year. The Cambodian students therefore wanted to work on Saturday mornings since this was one of their few free periods. Saturday morning Cambodian time translates to Friday evening US time. The project hence needed to revolve more around the customer's availability than the developer's, which is a situation very true to life. However, because the Cambodian students were serious about their education, they regularly put themselves out for the project. Both sets of students found that each side needs to make concessions if a project is going to be brought to a successful conclusion.

Scope Creep, Negotiation and Accountability

Most of the projects started off quite ambitious in scope. Students generally want to tackle something that is too ambitious; there is an obvious tendency to over-estimate abilities and over-commit. This is a common problem at the onset of most software development projects, but the consequence of such over-commitment is not always one a student team has to worry about. In this case, rather than simply making a commitment to a professor, the US students were making commitments to other students who thenceforth relied on the US students honoring their commitment to get their grade. A project of this nature put students in a position of inter-dependency. It becomes a group-shared risk.

Not surprisingly, the workload for this class overtook other classes for the US students as they attempted to meet commitments. The effort the students expended on this project was far greater than in software engineering class projects of the past. The end date was fixed so, as with most software development endeavors, either a reduction in scope was negotiated or the quality suffered to get the work completed on time. While we found that accountability bred pride and professionalism in students' work, there are still lessons to be learned about estimating and managing customer expectations.

The students gained valuable experience about being careful about what they promise. They found out about how critical it is to factor in their other commitments and responsibilities when estimating how much they can do. Four projects out of the five were accepted by the customers; most of them were prototypes because it is really difficult for students to develop completely functional software in a semester while simultaneously learning about the best practices and tools of software engineering. The project that was rejected did not end up with any implementation. The next obvious step may be to inject the concept of money and penalty clause into student projects.

Whole Team Concept and Clear Roles

The students had to organize themselves to get all the work completed on time. This required clear roles and responsibilities within the local team and across the whole team. In each team there was a leader, a communications officer and a quality assurance manager. The teams that performed the best were the ones where the leader acted like a leader and organized the team contributions, and where the communications officer managed the team communications and promoted awareness. The students did have trouble understanding that the role of the quality assurance manager does not begin when testing the software but on day one.

FINDINGS – ABOUT COMMUNICATION

The study monitored the communications, within the local team and extended team, through the use of weekly questionnaires, blog entries, and emails/chat archives. The objective was to study the models of communication used by the distributed teams and understand communication choices.

Overarching Communication Model

We saw three main styles of group communication.

- **Point-to-point:** one student in the local team communicated with one student in the local or non-local team on a random and as needed basis. Not every student was privy to every communication that occurred.
- **Broadcast:** one student in the local team communicated with all members of the local and/or non-local team simultaneously. All students received every communication.
- **Mediated:** communication between the local and non-local team was channeled through a consistent single member on each of the two teams who was responsible for organizing this and ensuring group contribution and awareness.

As part of our monitoring activities, we examined the ability of teams to satisfy milestones and the quality of their resulting work. We attempted to correlate this with the predominant style of extended group communication. While the findings cannot be generalized from a study of this size, we did find that the most successful student project, the project that maintained the highest standard of process and product quality throughout, was the one project that followed a mediator style for global communication (i.e. a single point of contact on each team). The least successful student project was one that consistently relied on closed and sporadic point-to-point communications. Broadcasting all items to all students, while it created the potential for shared awareness (crucial for global working), subjects students to unnecessary noise and non task-specific information. With no built in mechanism to acknowledge receipt, items are easily missed and document versions become out of step under such a model. Different models are likely to be more useful for sending different messages at different stages in a project. We did not have the data to explore this further, and this will be the focus of a more controlled experiment.

Preferred Modes of Communication

The study examined the mode of communication used for different categories of software development task. It focused on those activities associated with planning the work to be done and those activities associated with validating and providing feedback on work that had been completed. The study also considered activities that combined aspects of planning and validation. Students chose to make use of two forms of asynchronous communication, emails and blogs, and three forms of synchronous communication, chat sessions, telephone calls and face-to-face meetings.

The mode chosen to support local and global communication obviously differed. Results from the student weekly communication surveys showed that the most common way both the local and extended team communicated was using email. Face-to-face meetings were the second most common way to communicate, but this was only relevant in the context of a local team. Chat sessions were the third most popular way to communicate, again across both the local and extended teams. Emails were used evenly for planning and validation/feedback tasks, whereas local face-to-face meetings were predominantly undertaken to plan tasks, and extended team chats were predominantly used to conduct validation and feedback-oriented tasks across the whole team.

Intensity of Communication

An interesting finding was that the predominant mode of communication with the extended team changed over the lifetime of the project. At the onset of the project, chat was the preferred mechanism for getting to know members of the extended team, used for asking lots of early questions and getting feedback on understanding the problem and requirements. This mode slowed down and email became more prevalent as the project progressed and there was less perceived need for team bonding activities. Chat sessions only peaked around milestones, for instant feedback and validation. The three control projects did not engage the students as the global software development projects did. Communication with the personnel of Pace University Residential Life and Housing services was sporadic and primarily through emails.

Topics of Communication

The study examined the major topic of each communicative event. Again, the profile of the exchanges changed over time. At the onset of the project, much of the communication was about Cambodia, the US, education, jobs, movies, food, etc. Such discussions help with group bonding in the early stages of a project. As the projects progressed, the amount of non-project-specific communication slowed down as the focal topics became more project-specific. Irrespective of the mode, the major topic of most of the communications was associated with understanding and managing changing requirements.

Our survey findings strongly suggested that the most successful projects, i.e. those projects whose end products most closely meet customer needs, were those in which the students had taken more time out to discuss and find out non-project-specific information about their team members, their lives and their culture. The less interested the US team was in finding out about the Cambodians, and vice versa, the more technological the work and the more distant it became from satisfying the customers' real requirements.

LESSONS FOR EDUCATION AND PRACTICE

There are lessons for both computer science education and global software development practice from this experience.

Set-up and Managerial Costs

While it is possible to design a classroom experience that gives students a flavor of what it can be like to work as part of a global team, running a project of this nature requires considerable upfront preparation and on-going day-to-day management. Faculty members have to act more like program managers than educators to keep all the student projects aligned with the course topics and ensure teams are on track to complete. In this case, it was essential for the faculty members in the two institutions to meet face-to-face prior to the project and put many building blocks and ground rules in place. This is very typical in offshore outsourcing projects and is frequently a 'hidden' cost not factored into sourcing decisions.

The amount of prior work and control was a lesson about the realities of offshore outsourcing for the faculty members. In the future, students should possibly take more responsibility for setting up the projects and planning their end-to-end nature. Where this is handled for them, one of the trickiest and riskiest parts of a project (i.e. contract negotiation and lifecycle planning) is not experienced. This means that students come out of the experience with little knowledge of the initial set-up costs incurred by a project of this kind.

Just-in-time Learning

Where the aim is for students to gain both practical and educational experience in software engineering, it is important not to lose track of the latter. Students were learning about software development processes and techniques at the same time as which they were expected to apply them. In this situation, it is only possible for students to consider the immediate task at hand or, at most, one step ahead. It is difficult for students to plan and estimate for activities they do not yet understand, let alone know how to perform. Students therefore do what they need to do to get the job done. The software engineering best practices they are expected to apply take time to learn and this learning can slow them down. It is possible that this gives students a negative impression as to the utility of software engineering processes and techniques.

While such learning on the job readily occurs in professional practice, where practitioners have to learn about new emerging technologies to immediate deploy them, this may not be so desirable when attempting to educate students about the concepts underpinning the discipline of software engineering. It may be preferable to teach a course in software engineering such that students can practice their skills and experiment in the small, prior to bringing all these components together in an end-to-end fashion for a capstone project. With such a scenario, students would have a better comprehension of the scope and magnitude of what they are embarking upon.

Third-party Oversight

While running this study, we found it important to have an independent faculty member keep sight of the bigger picture. Those involved in the day-to-day issues can get caught up in the details of specific applications and immediate milestones. A third-party is able to look across projects for commonalities, differences and problem areas. They can monitor the quality of the processes and evolving products. This reflects the critical role of governance in offshore outsourced projects. There is an important need to educate students, not only about doing software development, but about the oversight and governance that is required to support global working.

CONCLUSIONS AND FUTURE WORK

The post course questionnaires indicated that the majority of the students had a positive learning experience. Most students were initially indifferent about offshore outsourcing. Following the course, they could articulate reasons both for and against working in this manner. The US students also gained a better awareness of the difficulties faced by technology workers in other countries. Furthermore, the end products of the global projects had more functionality than the control projects; these students had gone the extra mile.

The study brought to the forefront concern about what technical and 'softer' skills computer science students will need to employ to work as productive members of multi-cultural distributed software development teams. We are currently working on a project that is investigating the skills required and developing a maturity model to help teams assess their learning needs. Based on this, we are creating a learning resource to supplement classes. There is also a need to use collaborative tools to support global software development and we are actively investigating this aspect.

We propose extending the study approach to have both sets of students experience both sides of the offshore outsourcing equation. To this end, we are looking to involve other international institutions in our work. This semester (spring 2006) we are involving students from Delhi University in India as contractors working for the US students, who in turn are working for Cambodian students; their responsibility will be to deliver the database component for the project.

ACKNOWLEDGMENT

We would like to thank the Pace University and ITC students for their effort and commitment, the *Agence Universitaire de la Francophonie* for Internet access, and Dean Susan Merritt for her support with this international project.

REFERENCES

Aspray, W., Mayadas, A.F., and Vardi, M.Y. Globalization and Offshoring of Software, A Report of the ACM Job Migration Task Force, 2006.

Aspray, W., Mayadas, A.F., and Vardi, M.Y. "Educational Response to Offshore Outsourcing". *Proceeding of the Thirty-Seventh SIGCSE Technical Symposium on Computer Science Education*, Houston, Texas, USA, March 1-5, 2006.

Ferguson, E., Henderson, P., Huen, W., and Kussmaul, C. "T Offshore Outsourcing: Impact on CS/IS Curriculum". *Proceeding of the Thirty-Sixth SIGCSE Technical Symposium on Computer Science Education*, St. Louis, Missouri, USA, February 23-27, 2005, 258-259.

Ferguson, E., Kussmaul, C., McCracken, D., and Robbert, M.A. "Offshore Outsourcing: Current Conditions and Diagnosis". *Proceeding of the Thirty-Fifth SIGCSE Technical Symposium on Computer Science Education*, Norfolk, Virginia, USA, March 3-7, 2004, 330-331.