AN EXPLORATORY COMPARATIVE ASSESSMENT OF .NET AS A PEDAGOGICAL TOOL FOR TEACHING OBJECT ORIENTED SYSTEMS DESIGN

Peter Meso, Ph.D.

pmeso@cis.gsu.edu

404-651-3848

Jens Liegle, Ph.D

jliegle@cis.gsu.edu

404-651-3850

Georgia State University

Dept. of CIS

35 Broad St.

Atlanta, GA 30303

Proposed Track: Current Issues and Trends in Information Systems Education

AN EXPLORATORY COMPARATIVE ASSESSMENT OF .NET AS A PEDAGOGOCAL TOOL FOR TEACHING OBJECT ORIENTED SYSTEMS DESIGN

ABSTRACT

We employ the **Technology Acceptance Model (TAM)** of information technology (IT) innovation diffusion to assess the suitability and fit of a new IT, the .NET suite of technologies, as a pedagogical tool for teaching a technical information systems (IS) course. The reactions of students who adopted this technology for the completion of a class project requiring them to design and build an object-oriented distributed-system are examined, and their performance compared to that of students who opted for the more conventional technologies. Results of this study indicate that the factors that led to the selection of .NET over the other technologies were consistent with the TAM theory. Those project-teams that opted for .NET performed significantly better on the implementation/deployment part of the project, and reported significantly less technical difficulties than those who used the conventional technologies. Thereby the results confirm the **usefulness** and the **ease-of -use** of the technology, as per TAM, in facilitating the completion of the design project. Thus this study confirms the effectiveness of TAM and similar IT innovation diffusion theories as approaches for assessing the pedagogical fit and suitability of specific IT for teaching specific IS courses.

INTRODUCTION

The teaching of most scientific courses requires the use of definite technologies either within a laboratory setting or individually outside of the classroom. Teaching of technical IT courses suffers from the same burden. However, whereas many scientific courses requir e instructor's attention to technical support only during the class period, the nature of information technology, especially in the present Internet era, demands that the IS instructor attends to technical problems and systems administration issues on an on-going basis – even when not on duty. This tends to be the case even where separate personnel are available to perform technical-support duties, because students naturally communicate their technical problems directly to the instructor. Further, the perception that students have about the effectiveness of the instructor – and the true worth of the course – are directly affected by how well the selected technology functioned, its ease of use, and its usefulness in allowing the students to fulfill pertinent course requirements. The effectiveness with which all of their technology-related difficulties are resolved also impacts their perception about the course and its instructor. These perceptions have a direct bearing on the instructor-evaluations by students that are heavily used in most institutions to assess the teaching effectiveness of instructors. Therefore, identifying IT tools and technologies that are liked by students, yet useful, effective and relevant to the tasks and assignments to be completed in the course, is of significant importance to the instructor. Such tools/technologies may leverage the performance and effectiveness of the instructor.

The complexity of the technology, or its cumbersomeness, may also mitigate the students' ability to grasp and understand the core body of knowledge being disseminated in the course. According to pedagogical research, effective teaching tools and technologies enhance the learning capability of students and make the mastery of difficult principles much more simpler (Liddle, Brown et al., 1995; Janicki and Liegle, 2001). This research also points out that the teaching tools and technologies that prove to be effective in most cases are those that (1) are easy to use and easy to learn, (2) map a clear and direct path from the problem to its correct solution, allows for hand-on-learning or learning-by-doing rather than passive learning such as demonstrations by an instructor, and (3) minimize the technological barriers between student and the core-knowledge or princ iples being disseminated to the student (Janicki and Liegle, 2001). According to IT innovation diffusion theories, these very same principles propel the rapid diffusion of a new information technology. In other words, those technologies that bear the properties that enhance their adaptation will be more readily embraced and accepted by any type of user-group, including students. This holds true even in the context of an IS course. This means that the theories of IT innovation diffusion may be beneficial and effective at identifying the suitability and fit of particular emerging information technology as a pedagogical tool for a select type or group of IS courses.

In this paper, we employ the Technology Acceptance Model (TAM) to explore the diffusion an emerging suite of technologies, .NET, as a pedagogical tool for teaching object-oriented systems design with specific emphasis on the design of distributed web-based systems. The .NET suite of technologies is selected because it is a new technology recently launched to the public and, so far, having experienced minimal use as a pedagogical tool. This paper demonstrated that TAM can be used as a basis of identifying the suitability and fit of a particular technology for teaching a select body of IS knowledge. This is especially relevant when the technology is new and its benefits vis a vie the more established technologies largely unknown and untested. The paper is structures as follows. The ensuing section provides a definition of TAM, its use in past studies on IT diffusion, and extends this to demonstrate its usefulness in pedagogical research. Part 3 presents the research methodology and hypotheses. Part 4 reports the observed results while part 5 provides inferences drawn from these results and discusses the contributions of the study. Part 6 presents our conclusions and provides directions for future research.

TECHNOLOGY ACCEPTANCE MODEL AND PEDAGOGY IN IS EDUCATION

The Technology Acceptance Model (TAM) is the leading theory used to explain diffusion of information technology by individuals in business and industry (Gallivan, 2001; Chircu et a., 2000; Straub et al., 1997). Research abounds on the use of TAM in explaining individual adoption and acceptance of IT and the antecedent and consequent factors that propel such diffusion within groups and organizations (Davis, 1989, Davis et al. 1989; Szanja, 1996; Agrawal and Prasad, 1997; Thompson, Higgins and Howell, 1991; Moore and Benbassat; 1991; Karahana, Straub and Chervany, 1999; Gallivan, 2001). While TAM has been widely used in IS diffusion research, it has rarely been used to explain the selection of an IT tool or technology to support the teaching of a technical IT course. Neither has past studies focused on studying why certain IT tools and technologies are more favored than others in the teaching of certain IT skills and domain knowledge. Such studies have tended to compare particular technologies or skill sets with respect to predefined outcomes such as subject's productivity, subject's cognitive performance and some output artifacts (e.g. higher quality analysis diagrams, higher quality program code, etc).

The Technology Acceptance Model states that the factors that propel the diffusion of an Information technology are its ease-of-use and its usefulness (Davis, 1989; Gallivan, 2001). It has been used to explain diffusion using user-perception measures as well as actual usage measures. Within the context of an IT course, we expect that students will be attracted to that technology that is actually easy to use and directly relevant to the course requirement tasks that they must complete or technology that they perceive as bearing these traits. Therefore, assessing the reactions of students toward a particular technology can determine the effectiveness of that technology as a pedagogical tool for the course in question. Where students have a choice between two or more technologies, a comparative assessment of their reactions to each becomes possible. This provides a means for determining the technology that best fits the course, based on the feedback received from the students. Such feedback can be collected as perception measures – measuring perceived ease of use and perceived usefulness of the select technologies, or as actual measures – assessing the performance, productivity or quantity of problems/questions (degree of dependence of students on expert counsel) by students as they use the select technologies.

In this study we selected to use actual measures to assess the diffusion of a new technologies, .NET, and inferring from the observed behavior of the students whether .NET provides as good a pedagogical fit as a tool for teaching of a technical IS course, i.e. object-oriented design, as does conventional object-oriented technologies.

THEORY AND HYPOTHESES

According to the TAM theory, two factors propel the diffusion of an information technology – its ease of use, and its usefulness. With respect to technologies for systems design and construction, the technology is perceived to be easy to use if it allows the designer to effortlessly, or in simple straightforward steps:

- translate design models (blue prints) into program code,
- ensure that the specifications outlined in the design models (blue prints), remain are mapped into the functional program code without any loss in structure, logic, or elements stipulated in the design models
- integrate the multiple program-code modules or components into one seamless and unitary system, and
- deploy the system into production.

Said otherwise, a technology that provides intuitive-like features that include cues, graphical user interfaces, wizards, menus, and similar artifacts to guide the user through well established automated procedures of converting specifications into functional program modules or components, to integrate these components into a functional unitary system, and to deploy the system into production is easy to use. Such a system simplifies the systems building function by allowing the user to focus on the systems solution, rather than on how to operate the technology. Therefore, ease-of-use relates to the navigability, user-interfaces, automated capabilities, and the functionality of the technology – the ability to quickly and effortlessly discover how the technology works and how to use it.

The .NET framework comes with a new integrated development suite, Visual Studio.NET (VS.NET). VS.NET is a significant step forward from prior releases, in that it not only provides the programmer with an extensive online help, a fully integrated editor with intellitype –auto completion of method arguments, excessive wizard support and graphical drag/drop construction of user interfaces such as menus, windows, buttons etc., but it does this in a totally integrated manner. The underlying byte code allows developers to write part of the code in one language, and use it from another language, i.e. define classes in C# and use them in a Visual Basic.NET program. The integrated debugger will switch from one language to another in real-time. The consequence for a development team are that each participant can develop in the programming language they are most familiar with, and the different parts can be easily integrated and debugged – without the overhead of COM. This was especially helpful for the students of this study, since they came from one of three programming tracks (C++, Java, and Visual Basic) and often did not know the other languages at all. And since students from either track were familiar with the basic Visual Studio environment and Windows technology in general, the initial learning curve for this tool was very low.

Usefulness relates to the ability of the system to do what it was designed to do. A tool or technology is useful if the user is able to achieve pre-defined end-goals by following the pre-defined directions of how to use the tool or technology. For example if a user is not able to successfully deploy the system that has been designed while following the prescribed procedure to the letter when using a said tool, then that tool is not useful. Usefulness relates to the capabilities of the technology rather than on how the technology works, or how to navigate within the technology. It has nothing to do with user interfaces, but rather with whether the technology is able to deliver the anticipated outputs. A technology is useful if it allows the user to obtain the desired outputs. In the context of systems development, these outputs are a functioning production system that meets established quality standards. Therefore an IS technology for building IS solutions is useful if it enables the designer to create information-system solutions that are:

- Efficient (solutions with minimal lines of code and high through-put),
- Robust
- Scalable
- Reliable
- Capable of rigorous data quality and integrity (data-validation, data verification, run-time error detection and correction, etc)

Here again Visual Studio.NET had a number of advantages: Since Microsoft rigorously tested it through numerous beta releases, the reliability and robustness was not much of an issue. Scalability is superb, since it is designed as an enterprise level programming environment. Efficiency is greatly enhanced through the use of multiple wizards that generate most of the code for the programmer, i.e. user interfaces and database connectivity. Error detection etc. is supported by the most powerful debugger that exists to date, which allows the programmer to start debugging a client side application and seaming less switch to the code of the server side program – in the same session!

Java technologies, on the other hand, are traditionally known to be difficult to learn and use. This has largely been to the fact that most use text-based interfaces, and the various tools required to complete a project were usually not integrated in one comprehensive suite of programs. In recent years, an increasing number of java integrated development environments (IDE) vendors have built graphic user-interfaces into their IDEs and some have also implemented wizards to simplify the systems construction, testing and deployment processes. On the whole however, these tools remain less advanced that the Visual Studio.NET in terms of user-support. Most still require that a separate deployment tool be initiated when a developer transitions from code-generation and testing to system deployment. Therefore we hypothesized that .NET will be easier to use than conventional java-based technologies. That is, users will require less technical information about how the .NET technology works in order to successfully use .NET to develop an IS system solution than they would require were they to use java based technologies. Hence we expect the following hypothesis to be rejected:

> *H1: There will be no significant difference in observed ease-of use between student groups who use .NET and those who use J2EE/JSP technologies.*

Given that both technologies have been developed to address object-oriented enterprise systems development, and both have capabilities for developing web-based systems, we hypothesized that there would be no significant difference in the usefulness of both technologies for developing systems solutions. In other words, the capabilities demonstrated by .NET in facilitating the construction of a web based distributed object oriented information system, would also be evident in conventional java technologies. Hence we expect that the following hypothesis be supported:

> *H2: There will be no significant difference in observed usefulness between student groups who use .NET and those who use J2EE/JSP technologies.*

The usefulness of the emergent technology, .NET, was assessed by the performance of the student project-teams on part 3 of the project. This is the part in which they constructed the system solution as per the design specifications developed in part 1 and part 2 of the project respectively. The ease-of use of the technology was assessed by the how these student project-teams performed on the presentation assignment. In this assignment, each team demonstrated the technology they used to build the system, outlining how that technology supported the generation of program code, the debugging and testing of the same, the integration of the various programs into one system and the deployment of the final solution as a production system. They also demonstrated the finished solution (the system they had built) to confirm that it indeed met the stipulations outlined in the project narrative. The presentation assignment was evaluated both by fellow students and the instructor of the course. The course instructor evaluated all parts of the project.

RESEARCH METHODOLOGY

This study was conducted over a period of one semester at a leading south-eastern university. The subjects were senior undergraduate students, all majoring in computer information systems, that were enrolled for the senior level systems design course. They were drawn from two sections of the same course, taught by the same instructor, in which they completed a systems design project requiring them to design and build an object-oriented web based distributed enterprise information systems. The project teams were formed by students self-selecting themselves into groups of three to five members. In total 13 teams were formed. The instructor provided the narrative for the project. The project had three parts to it: a conceptual design part, a detailed design (implementation design) part and a systems-building part. All groups used the rational-rose software as the standard CASE tool in the completion of the conceptual design part and the detailed design part of the project. They were allowed to select any technology or set of technologies of their choice for the systems building part of the project. The technologies they selected thus determined what treatment group they belonged to. In total four different technologies were selected. These were .NET (3 teams), Conventional ASP (3 teams), Java J2EE/JSP technology (6 teams), and HTML/CGI (1 team). Students also made a presentation of their project at the end of the semester.

The independent variable in this study was the technology used by a group to build the information system solution. The dependent variables, derived directly from TAM theory, were ease-of-use and usefulness of the technology respectively. Results from the Project's third part (building the system) were used assess the **usefulness** of the .NET suite of technologies (the emergent technology) in comparison to the more established technologies. Results from the Students' project team presentations were used to assess the .NET suite of technologies' **ease-of-use** in comparison to these other technologies. Through the course of the semester, students received instruction on object-oriented design using UML and practical demonstrations on how to use the Rational Rose CASE tool to draw the design models. The students completed three examinations in the course of the semester. We used the results from the first part of the project and from the average total exam performance of each group to test for the differences among the treatment groups in terms of their knowledge of the subject matter. Demographic date on the students was also collected through a survey questionnaire and analyzed for any intertreatment-group differences. Results from the second part of the project were dropped from the study because they did not meet the condition of normally distributed data as measured by the skewness (-1.997) and the kurtosis (4.372) statistical tests (See Table 1).

Table 1: Test for Normality of Data in Study's Variables

	N	Min	Max	Mean	Std. Dev	Skewness		Kurtosis	
Scores						Statistic	Std. Err	Statistic	Std. Err
Exam	62	16.40	29.00	25.0258	2.3343	-.852	.304	2.062	.599
Project Part 1	62	90.00	100.00	97.0000	3.1100	-.716	.304	-.547	.599
Project Part 2	62	61.00	100.00	90.8226	9.4288	-1.997	.304	4.372	.599
Project Part 3	62	64.00	99.00	77.5000	11.6348	.809	.304	-.758	.599
Project Total	62	23.40	29.90	26.5323	1.7899	.355	.304	-.587	.599
Presentation	62	80.00	98.00	88.1129	5.7747	.119	.304	-1.148	.599
Valid N (listwise)	62								

The nature of the study, being to use diffusion theory to assess the suitability and fit of a new technology for the pedagogical support of a technical IS course, mitigated the need for allocating technologies to student groups. An analysis of their performance in the first two parts of the project, and in the total exam scores revealed that there was no significant difference across the treatment groups (See Table 2).

TABLE 2: ANOVA Comparison of Differences in Competency and Knowledge of Subject Matter Across Treatment Groups

				Sum of Squares	DF	Mean Square	F-Value	Test for Significance (P-value)
Exam Total	Between Groups		(Combined)	17.489	4	4.372	.791	.536
		Linear Term	Unweighted	.148	1	.148	.027	.871
			Weighted	.185	1	.185	.034	.855
			Deviation	17.304	3	5.768	1.044	.380
	Within Groups			314.890	57	5.524		
	Total			332.379	61			
Project Part 1	Between Groups		(Combined)	113.509	4	28.377	3.395	.015
		Linear Term	Unweighted	26.355	1	26.355	3.153	.081
			Weighted	29.490	1	29.490	3.528	.065
			Deviation	84.019	3	28.006	3.350	.025
	Within Groups			476.491	57	8.359		
	Total			590.000	61			

Likewise, the analysis of demographic data confirmed a lack of significant difference in the age, work, experience, prior experience with UML and prior experience with object-oriented programming across the treatment groups. Therefore, the lack of significant differences across treatments with respect to their knowledge of the subject matter, and their demographic set-up confirms the similarity of the groups and allows for the statistical comparison of the effects of the technologies they selected on their respective performance in both the project's third part and its presentation.

RESULTS

Descriptive statistics were evaluated for the student-group performances on each part of the project and on the presentation assignment. First we ran tests to confirm if the data collected from the study was normally distributed. This was done because the treatment groups in this study had not been deliberately designed. Rather, in keeping with the principle of diffusion, the subjects had been allowed to select their own development technologies – and these technologies established the respective treatment groups for the study. Skewness and kurtosis tests confirmed that the data used as measures for each of the hypotheses above, namely Performance on Part 3 of the project, and Performance on the Presentation, was normally distributed (Table 1). Therefore, we were able to use it in the ANOVA analysis to statistically analyze the differences between the treatment groups, with respect to the hypotheses.

Analysis of variance (ANOVA) tests confirmed lack of support for hypothesis H1 and lack of support for hypothesis H2. Table 4 presents the results of the ANOVA. The p values of .000 and .000 indicate that the differences in the performance of groups that used .NET (Means 90 and 95 respectively) and those that used java technologies (Means 74 and 86 respectively) are significant both in the usefulness (Project Part 2) and the ease-of-use (Presentation) constructs. As hypothesized in H1, .NET teams were evaluated more highly on the ease-of-use construct than Java Teams (p- value = 0.000). This indicates that students found .NET to be an easier to use technology than the java family of technologies. Contrary to the expectations expressed in H2, student teams that adopted .NET scored significantly better on the usefulness of technology construct than those who decided to use the conventional java-based technologies (p- value = 0.000).

TABLE 4: Comparison of .NET to Java Technologies on usefulness (Proj. 3) and Ease of Use (PRESENT) variables

		Descriptive Statistics				Levine's test		t-test for equivalence of means				
	Tech.	N	Mean	Std. Dev.	Std. Err Mean	f	sig	t	df	sig	Mean Diff.	St. Error Diff.
Proj. 3	.NET	16	90.0625	11.2574	2.8144	4.476	0.04	5.175	46	.000	15.4375	2.983
	Java	32	74.6250	8.9181	1.5765							
Present.	.NET	16	95.0625	2.9545	.7386	24.292	0.000	6.606	46	.000	8.4375	1.277
	Java	32	86.6250	4.6472	.8215							

These findings confirm both the ease of use and the useful of .NET as a suite of technologies as a pedagogical tool for teaching systems design, especially the design of object-oriented distributed web based IS solutions. Further, the findings provide support for the relevance of TAM as both a theory and framework for evaluating the pedagogical fit and suitability of a new technology with respect to a specific technical oriented IS course. TAM allowed us to empirically and objectively compare the effectiveness of a new technology. The findings, that the new technology in this study outperformed the existing technology, by allowing the students who used it to achieve the stipulated course objective of building a quality solution to a systems design problem and perform better at the task than those who did not, provides grounded evidence that the new technology is a better pedagogical tool than the existing technology. However, this conclusion is limited to the specific context in which the emergent technology was compared to existing technologies and cannot be generalized to all IS courses, or even to systems design courses that take on a different focus (e.g process oriented as opposed to object-oriented design).

We conducted a detailed ANOVA to identify if these observed differences applied even across the control technologies of the conventional ASP and HTML/CGI that were adopted by some groups within this class. Table 5 presents the results of this analysis. While groups that adopted the .NET suite of technologies significantly outperformed all other treatment groups, there was also a significant difference between the groups that adopted Java technologies and those that adopted HTML/CGI in the EASE-OF-USE construct (p value= 0.003). No significant difference was evident between groups that used conventional Java and those that used conventional ASP in both the EASE-OF-USE and the USEFULLNESS constructs. Likewise, there was no significant difference in the usefulness construct between the teams that used Java based technologies, and those that used HTML/CGI. The teams that used conventional ASP did not differ in any significant way from those that used HTML/CGI on both the ease-of-use and usefulness constructs. This analysis reveals that the emergent technology, .NET was more easy-to-use and more useful than the control technologies of conventional ASP and HTML/CGI, besides outperforming the Java based technologies on the same two constructs.

As stated earlier, these findings simply confirm the suitability of the new and emergent technology as a pedagogical tool for a well defined technical IS course. They do not provide support for the notion that the emergent technology is a superior technology to the existing technologies.

Table 5: Multiple Comparisons Anova Analysis using Bonferroni's test for difference in means

			Mean Diff. (I-J)	Std. Error	Sig.	95% Confidence Interval	
Dep. Variable	(I) TECHNO	(J) TECHNO				Lower Bound	Upper Bound
Project Part 3	.NET	.J2EE/JSP	15.4375	2.6809	.000	8.1140	22.7610
		Old ASP	17.7292	3.6482	.000	7.7631	27.6953
		HTML/CGI	25.0625	4.4860	.000	12.8079	37.3171
	J2EE/JSP	.NET	-15.4375	2.6809	.000	-22.7610	-8.1140
		Old ASP	2.2917	3.3036	1.000	-6.7330	11.3163
		HTML/CGI	9.6250	4.2105	.156	-1.8771	21.1271
	Old ASP	.NET	-17.7292	3.6482	.000	-27.6953	-7.7631
		J2EE/JSP	-2.2917	3.3036	1.000	-11.3163	6.7330
		HTML/CGI	7.3333	4.8837	.832	-6.0078	20.6745
	HTML/CGI	.NET	-25.0625	4.4860	.000	-37.3171	-12.8079
		J2EE/JSP	-9.6250	4.2105	.156	-21.1271	1.8771
		Old ASP	-7.3333	4.8837	.832	-20.6745	6.0078
Presentation	.NET	.J2EE/JSP	8.4375	1.1390	.000	5.3259	11.5491
		Old ASP	9.5069	1.5500	.000	5.2726	13.7413
		HTML/CGI	15.0625	1.9060	.000	9.8558	20.2692
	J2EE/JSP	.NET	-8.4375	1.1390	.000	-11.5491	-5.3259
		Old ASP	1.0694	1.4036	1.000	-2.7649	4.9038
		HTML/CGI	6.6250	1.7889	.003	1.7381	11.5119
	Old ASP	.NET	-9.5069	1.5500	.000	-13.7413	-5.2726
		J2EE/JSP	-1.0694	1.4036	1.000	-4.9038	2.7649
		HTML/CGI	5.5556	2.0750	.058	-.1127	11.2239
	HTML/CGI	.NET	-15.0625	1.9060	.000	-20.2692	-9.8558
		J2EE/JSP	-6.6250	1.7889	.003	-11.5119	-1.7381
		Old ASP	-5.5556	2.0750	.058	-11.2239	.1127

DISCUSSION

These findings indicate that using TAM may facilitate the identification and selection of suitable IT tools for teaching technical IS courses. Selection of such technologies, for which .NET is an example, provide gains to instructors of technical information systems courses by mitigating the course-administration burden that such courses place on the instructor.

This paper would be incomplete without some documented observations of the instructor in during the life-time of the course. In retrospect, these observations explain or point to the reasons why the student teams that adopted .NET may have ended registering a more superior performance than their colleagues. First, the student-groups that selected to use .NET and simple text editors (ASP, HTML/CGI) were able to install and configure their selected technologies without the instructor's assistance. However, all but one of the student groups that used the Java technologies types of technologies experienced installation and set-up problems significant enough to have them consult with the instructor. This was likely due to the complex nature of Java-based technologies, most requiring add-ons and components from different vendors in order to function. The most common problems that students had regarding technology installation and configuration had to do with establishing database connectivity and getting the server-side run-time environment to function correctly. Most of the course preparation and administration time in technical IS courses is spent on issues relating to installing and configuring the IS technologies to be used by the class, and administering/maintaining these through the lifetime of the course. Thus the first lesson learned from this study is that a technology that allows the students to perform these tasks simply and effectively, without excessive dependence on the instructor, greatly reduces the technological burden of delivering the course. Both the instructor and the students benefit in this set-up. Students get to learn the nuances of a technology that otherwise only the instructor knows, and gain confidence in technology management. The instructor, on the other hand, is able to allocate more course time to addressing subject knowledge issues rather than administrative issues. As such providing students with more in-dept knowledge of the subject matter becomes not only possible, but feasible, even within the time constrains of a single semester.

Second, groups that selected .NET required little or no assistance in solving problems regarding how to use the IS tool, and how to generate, compile and run software code using the tool. Their consultations largely involved translation of UML symbols, diagrams and logic into appropriate software-code and persistent date repositories. This observation was also seen among the teams that used a mix of HTML/Dynamic HTML and server-side scripting using CGI. This is because they used simple text editors to complete their project. Student groups that selected to use java had significantly more difficulty in learning how to use the integrated development environment they has selected without assistance from the instructor. This tended to inhibit their focus on, and performance in completing the actual IS development project. Said otherwise, because groups that opted for .NET and simple text editors spent less time in learning tool specific knowledge, and they were able to allocate more time to learning domain specific knowledge (systems design knowledge needed to complete the IS project). On the other hand, those groups that opted for Java-based tools spent so much time learning tool-specific knowledge that they had little time left to concentrate on domain specific knowledge. Therefore, the second key lesson learned from this study is that selecting an easy-to-use, easy to learn IS tool as per the parameters of TAM allows for more, if not all, class time to be used in teaching the core knowledge and principles pertinent to the technical IS course being taught rather than in teaching how to use a specific IS tool that supports, facilitates, or implements the learned core-knowledge. The result is a purer pedagogical coverage of the core-knowledge in the teaching of technical information systems courses.

Still, more studies need to be conducted before our findings can be generalized. In addition, one should keep in mind that .NET is a fairly new technology and that students who opted for using .NET for their project are therefore early adaptors – a group that often consist of students that are very comfortable with new technology and usually the better IS students overall.

CONCLUSION

We employed the **Technology Acceptance Model (TAM)** of information technology (IT) innovation diffusion to assess the suitability and fit of a new IT, the .NET suite of technologies, as a pedagogical tool for teaching a technical information systems (IS) course. Results of this study indicate that the factors that led to the selection of .NET over the other technologies were consistent with the TAM theory. Those project-teams that opted for .NET performed significantly better on the implementation/deployment part of the project, and reported significantly less technical difficulties than those who used the conventional technologies. They also performed better on the presentation of the project. Thereby the results confirmed the **usefulness** and the **ease-of -use** of the technology, as per TAM, in facilitating teaching of an object oriented systems design course. Further, they confirmed the effectiveness of TAM, and similar IT innovation diffusion theories, as approaches for assessing the pedagogical fit and suitability of specific IT for teaching specific IS courses.

BIBLIOGRAPHY

Agarwal, R., and Prasad, J., A Conceptual and Operational Definition of Personal Innovativeness in the Domain of Information Technology, Information Systems Research, 9, 2, (1998), 204-215.

Chircu, Alina M, Kauffman, Robert J, Limits to value in electronic commerce-related IT investments, Journal of Management Information Systems, 17,2, (2000), 59-80

Cooper, Randolph B., Bhattacherjee, Anol, Preliminary Evidence for The Effect of Automatic Responses to Authority on Information Technology Diffusion, Database for Advances in Information Systems, 32, 3, (2001), 36-50

Davis, F.D., Perceived Usefulness, Perceived Ease-of-Use and User Acceptance of Information Technology, MIS Quarterly,13, 3, (1989), 319-339.

Davis, F.D., Bagozzi, R.P, and Warshaw, RR., User Acceptance of Computer Technology: Comparison of Two Theoretical Models, Management Science, 35, 8, (1989), 982-1003.

Gallivan, Michael, Organization Adoption and Assimilation of Complex Technological Innovations: Development and Application of a New Framework, Database for Advances in Information Systems, 32,3,(2001), 51-85

Janicki, Thomas N. and Liegle, Jens O.:Development and Evaluation of a Framework for Creating Web-based Learning Modules. Journal of Asynchronous Learning Networks, 5, 1 – (June 2001), 58-84

Karahanna, E., Straub, D.W., and Chervany, N.L., Information Technology Adoption Across Time: A Cross-Sectional Comparison of Pre-Adoption and Post-Adoption Beliefs, MIS Quarterly, 23, 2, (1999), 183-213.

Liddle, J., Brown, K., Slater, A., MacDonnchadha, S. : Utilising Multiple Training Strategies within Intelligent Industrial Training Systems, AI-ED 95 workshop "Authoring shells for intelligent Tutoring Systems", Washington D.C August 1995, 1-7

Straub, Detmar W., Mark Keil and Walter Brenner, Testing the Technology Acceptance Model across Cultures: A Three Country Study, Information & Management, 31, 1, (1997), 1-11

Szajna, B., Empirical Evaluation of the Revised Technology Acceptance Model, Management Science, 42, 1, (1996), 8592.

Thompson, R.L., Higgins, C.A., and Howell, J.M., Personal Computing: Toward a Conceptual Model of Utilization, MIS Quarterly, 15, 1, (1991), 124-143.