December 2005

Critical Factors in Software Adoption

Dan Port
University of Hawaii at Manoa

Ann Takenaka
University of Hawaii at Manoa

David Klappholz
Stevens Institute of Technology

Critical Factors in Software Adoption

Dan Port
Ann Takenaka
University of Hawaii at Manoa
{dport, atakenak}@hawaii.edu

David Klappholz
Stevens Institute of Technology
aklappho@stevens.edu

Abstract

The software engineering literature abounds with accounts of "failed" software projects, i.e., projects that came in far over schedule, and/or far over budget, or were cancelled, before completion, with the loss of significant investment. An oft-mentioned whose cause has been the subject of little or no serious study is that of shelfware, i.e., successfully completed software that is never adopted. The work described here is, to our knowledge, the first empirical study of the causes of shelfware, a study that has already resulted in the reduction of shelfware in one of the organizations studied. Our purpose is to identify and understand those factors that can lead to the non-adoption of successfully completed software, with the idea of monitoring those factors during the software development process, in order to maximize the probability of adoption – or to kill the project if adoption is judged to be unlikely.

Keywords: shelfware, software adoption, software economics

1. Introduction

The software engineering literature abounds with accounts of "failed" software projects, i.e., projects that came in far over schedule, and/or far over budget, or were cancelled, before completion, with the loss of significant investment [Standish 2004] . There are myriad papers on cost and schedule estimation [Boehm et al 2000], on various approaches to development process, from agile [Beck 2004, **Crispin 2002,** Jeffries 2000], to planned (heavy weight) [Paulk 1995], and on tailoring process to the right point, on the continuum between agile and planned [Boehm,Turner 2004] – all aimed at bringing in software development projects within budget and schedule, and with all functional and non-functional requirements met. But, on-time, within-budget, top-quality software is far from "successful" software.

An oft-mentioned phenomenon, whose cause has been the subject of little or no serious study is that of shelfware, i.e., successfully completed software that is never adopted. The term "shelfware" applies to both shrink-wrapped software that is bought/licensed but never used and custom software, intended for a specialized application – either developed from scratch or customized COTS. The work described here is, to our knowledge, the first empirical study of the causes of shelfware, a study that has already resulted in the drastic reduction of shelfware in one of the organizations studied. In what follows, we will restrict our attention to custom software. Our purpose is to identify and understand those factors that can lead to the non-adoption of successfully completed software, with the idea of monitoring those factors during the software development process, in order to maximize the probability of adoption – or to kill the project if adoption is judged to be unlikely.

Questions of interest include:

- -What are the factors that contribute to the non-adoption of custom-developed software when both the developers and the client are satisfied with it?
- -Is there a set of factors whose satisfaction is necessary for adoption?
- -Is there a set of factors whose satisfaction is sufficient for adoption?
- -Is adoption predicated upon some weighted sum of satisfaction levels' rising above a threshold value?
- -How can awareness, during software development, of identified factors, and of their relative influences on (non)adoption increase the probability of adoption – or of early (justified) project termination?

2. Initial (University) Study

Our initial study of the shelfware issue resulted from frustration with the low rate of adoption of software developed by students in USC's CS577, a two-semester graduate course in Software Engineering. CS577 students, many of whom have industry experience, spend a full academic year, working in groups of five, developing (real) e-service applications for (real) faculty and staff clients. (Only projects that are judged to have high probabilities of ultimate success are allowed to proceed past the first semester.) For numerous examples see [Boem et al 1999].

By 1997, a few years after the course was first offered, it was noted that, despite the fact that almost all teams whose projects persisted beyond the end of the first semester were producing first-rate software – first-rate in the eyes of both client and developer – a less-than-desirable fraction of that software ended up being put into regular use. The immediate issue of interest was finding the factors that were causing the problem – and either eliminating or reducing them – because of the fear that low adoption rates would discourage faculty and staff from participating as clients in future offerings of the course. Of equal interest was the likelihood that answers to course-related adoption questions would lead to answers to the issue of shelfware in industry.

The first step in investigating the problem was to interview clients of completed projects. The result of these interviews was the identification of nine (non)adoption-related factors, the first five of which relate to the client:

1. Domain-Knowledgeable client: "Domain knowledge" refers to knowledge of the client organization and of those business processes relevant to the software product. The issues over which the client must have command in order to be deemed domain knowledgeable include:
   - -the rationale for the product
   - -the goals of the product
   - -the business processes that the product encompasses
   - -the impact of the product on the organization, including its business processes
   - -the conditions necessary for the product to provide value to the organization
2. Representative client: A client is deemed representative if s/he
   - -has been granted the right to represent the organization in dealing with the developers

-has the authority to mandate the use of the software product if s/he deems it to be acceptable.

3. Focused client: A client is deemed to be focused if
   -s/he pursues a well-defined set of consistent organizational objectives for the software under development
   -those objectives are relatively stable over the software's development cycle
4. Collaborative client: A client is deemed collaborative if s/he is willing and able to work with the developers and/or vendors, to maintain a shared vision of the product
5. O&M-sensitive client: A client is deemed O&M-sensitive if s/he is willing and able (empowered), at the appropriate point(s) within the software product's life cycle, to allocate resources necessary for operation and maintenance of the product. Such resources may include: salaries for (hardware and software) maintenance staff, for database administrators, and for system administrators; and COTS licensing fees.

In addition to the four client-related factors, the following four environment-related factors were identified:

6. Stable environment: For the purposes of this discussion, the environment includes the organization's micro-environment and its macro-environment.
   -The microenvironment includes:
      -other departments within the organization
      -suppliers
      -marketing intermediaries
      -customers
      -competitors
      -various publics, where a "public" is any group that has an actual or potential interest in, or impact on, an organization's ability to achieve its objectives. For example, financial publics are banks, investment houses, and stockholders that can influence the organization's ability to obtain funds, and, thereby, affect the IT budget.
   -The macro-environment includes:
      -demographic forces
      -economic forces
      -natural forces
      -technological forces
      -political forces
      -cultural forces
7. (pre-deployment) Site preparation, for example: timely installation and setup of appropriate computer hardware, of security devices, and of climate control equipment if necessary; and allocation of staff for adequate hours of operation.
8. (pre-deployment) Software preparation, for example: installation of the appropriate operating system and of any needed COTS products

9.(pre-deployment) Staff preparation, for example: training, in operation and maintenance, of users, of system administrators, and of database administrators

Starting in academic year 1997-1998, USC CS577 students were instructed in the meanings of the factors listed above, and were given scorecards to use in assessing them for their projects. The notation used in the scorecards, and repeated below in Table 1, is as follows:

- "+": the factor was adequately addressed
- "(+)": the factor was partially addressed
- "-": the factor was inadequately addressed

Table 1, below, shows the data thru the end of academic year 2000-2003.

Application	Client Characteristics					Transition Preparation				Outcome	
	Focused	Representative	O&M Resources	Collaborative	Domain Knowledge	Software	Site	People	Stable Env.	Client Success	Adopted
1996-1997											
EDGAR Business Data	+	+	-	+	+	+	-	-	+	+	-
Medieval Manuscripts	+	-	-	+	+	+	-	-	+	+	-
Technical Reports	+	+	-	+	+	+	-	-	-	+	-
Latin American Pamphlets	+	+	-	+	+	+	-	-	+	+	-
Cinema-TV	+	+	+	+	+	+	+	(+)	-	+	(+)
Image Archives	-	-	-	+	-	+	-	-	+	-	-
1997-1998											
S-Charts	+	+	+	+	+	+	(+)	+	(+)	+	(+)
Global Express	+	+	+	+	+	+	-	+	-	+	-
Hancock Virtual Museum	+	+	(+)	+	+	+	+	+	-	+	-
Serial Control Records	+	+	+	+	+	+	+	+	(+)	+	(+)
B-School Working Papers	+	+	+	+	+	+	+	+	+	+	+
1998-1999											
Data Mining	+	+	+	+	+	+	+	+	(+)	+	+
Dissertations	+	+	+	+	+	+	+	(+)	+	+	(+)
Hispanic Archive	+	+	+	+	+	+	+	+	-	+	-
WWI Archive	+	+	+	+	+	+	+	+	-	+	(+)
1999-2000											
Oversized Object Viewing	+	+	+	+	+	+	+	+	+	+	+
East Asian Ingest	+	+	+	+	+	+	+	+	+	+	+
New Books List	+	+	+	+	+	+	+	+	(+)	+	(+)
Chicano/Latino Serials	+	+	+	+	+	+	(+)	+	-	+	-
Vacation/Sick Leave Tracking	+	+	+	+	+	+	+	+	-	+	-
Business/Reference Q&A's	+	+	+	+	+	-	-	-	-	-	-
2000-2001											
Web Mail	+	+	+	+	+	+	+	+	+	+	+
Full Text DB Search	+	+	+	+	+	+	-	+	+	+	-

Dental Journal ToC	+	+	+	+	+	+	+	+	+	+	+
Pathology Slides	(+)	+	+	+	+	(+)	(+)	(+)	(+)	(+)	-
Arch/F. Arts Slides	(+)	+	+	+	+	+	-	-	-	+	-
Velero Archive	+	+	+	+	+	+	+	+	+	+	+
2001-2002											
Dental Booklist	+	+	+	+	+	+	+	+	+	+	+
Velero Map	+	+	+	+	+	-	-	-	-	-	-
Collab. Service	+	+	+	+	+	-	-	-	-	+	-
eResources Query	+	+	+	+	+	+	+	+	+	+	+
Collab. Mgmt	+	+	+	+	+	+	+	+	(+)	+	(+)
Risk Tool (DART)	+	+	+	+	+	+	-	-	(+)	+	+
O-Tree eBASE	+	+	+	+	(+)	+	(+)	+	(+)	+	+
MBASE/BORE	+	+	+	+	+	(+)	+	+	-	+	-
QM/BORE	+	+	+	+	+	(+)	+	+	-	+	-
2002-2003											
Dia UML	+	+	+	+	+	+	+	+	+	+	+
Geotech Data	+	+	+	+	+	+	+	+	+	+	+
Leavey Sched.	+	+	+	+	+	(+)	(+)	(+)	+	(+)	(+)
QMIS	+	+	+	+	+	+	+	+	+	+	+
Caroline's Closet	+	+	+	+	+	+	+	+	+	+	+
MSCS-SE DB	+	+	+	+	+	+	+	(+)	(+)	+	(+)
CSE Tech Report	+	+	+	+	+	+	+	+	(+)	+	(+)
CRM Tool	+	+	+	+	+	+	+	+	+	+	+

Table 1

The results shown in Table 1 suggest that:

-all nine factors much be satisfied in order for total adoption.

-partial adoption requires satisfaction of all but at most two factors, each of the latter of which must be at least partially satisfied.

Table 1 shows clearly that if the client does not deem the project a success, the software will not be adopted. The converse, however, is not true. That is, the client's deeming the project successful does not imply that it will be adopted. (In any case, client success is a project outcome rather than a "factor" in the pursuit of the project.). As a start on addressing the question of the relationship between the identified factors and software adoption,, we fit an ordinal logistic model to the data in Table 1 after assigning each factor a score of -1 if not addressed adequately, 0 if addressed partially, and 1 if fully addressed. These sum of weights is used as the predictor and is correlated with an outcome value of -1 if not adopted, 0 if partially adopted, and 1 if fully adopted. Note that this sum represents the "degree of excess" of satisfied over unsatisfied factors. The questions of interest are:

(Necessity): Is there a minimum sum required for adoption?
(Sufficiency): Is there a sum over which the software is guaranteed to be adopted?
(Criticality): How narrow is the threshold for adoption?

Note that we assume that the factors are interchangeable with respect to software adoption and that satisfaction, partial satisfaction, and non-satisfaction are equally weighted. While these

assumptions may not be entirely reflective of the actual natures of the factors, they serve as a baseline for future refinements of this study. (Indeed, non-uniform weightings and exchanges can only serve to improve the predictive model.)

Figure 1 shows the resulting models for software adoption and for client success; they fit the data remarkably well, with high confidence (R square value of 0.43 where random data has a value of less than 0.01 based on the 44 observations). Based on the Adoption Prediction model of Figure 1, the identified factors are indeed "critical" in that there are very sharp thresholds between adoption and non-adoption. A project with a score of less than 8 has a 10% chance of non-adoption. Just 3 points less and this jumps to 85%. Even with the maximum 9 points, the chance of full adoption is only 80%, which drops sharply to 20% at 7 points. Partial adoption is dicey at this point hovering a little under 50%.

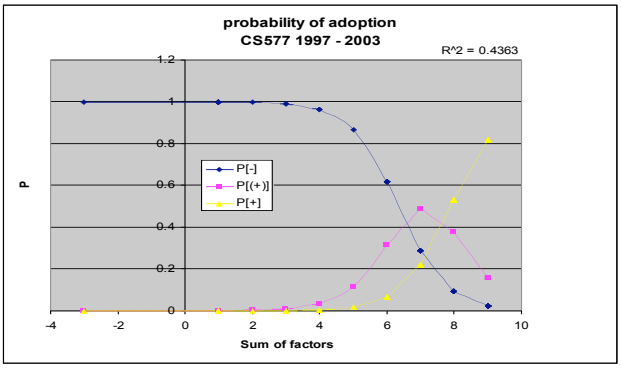

Figure 1: Adoption Prediction

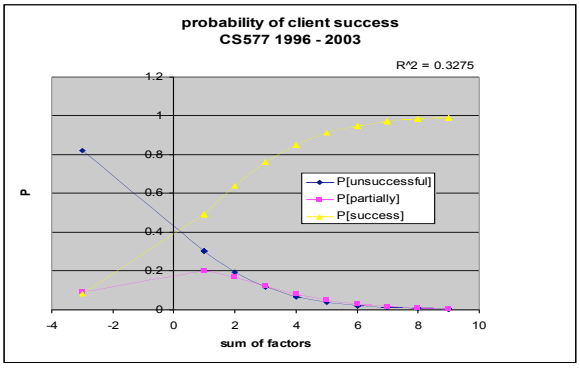

Figure 2: Client Success Prediction

Figure 1 shows very clearly that the identified factors are critical for adoption in that the weighted sum of factors must go over a threshold value before adoption has more than a negligible probability. In terms of necessity, Figure 1 indicates that non-adoption is monotonically decreasing. Thus if there is an excess of at least 4 satisfied factors, non-adoption can be avoided (but only barely!) and adoption becomes more likely as the sum of factors increases. However, even though adoption increases monotonically after an excess of, there is no "sufficient" set of factors for adoption unless one is willing to accept an 80% chance of adoption with all 9 factors satisfied. Our interpretation of this result is that there are likely more factors we must consider. Indeed, for later studies, including the industry study discussed below, we have added 3 more factors which now seem to enable sufficiency (over 95% chance of adoption for successful projects).

It must be noted that while the decreasing and increasing nature of the non-adoption and adoption curves is a mathematical feature of the logistic model, it is the fact that these curves are flat and non-flat at the ends which is a feature of the data itself. For example, if the data were random, these curves would be nearly linear throughout.

As to prediction of client satisfaction, consider Figure 2. It is clear that the factors are not necessary nor are they sufficient or critical. In fact, the "success" curve is really not that far from linear and is only "flat" very near the end. Nonetheless, if there is an excess of at least 4 satisfied factors over the likelihood of client satisfaction is over 85%. Thus for CS577 projects, it is relatively easy to achieve "client success." We attribute this to the fact that university clients are often reluctant to criticize student projects for fear that criticism will result in low grades. Furthermore, university clients often deem their participation worthwhile even when they do not end up with the software they had originally hoped for —or with no working software at all. In fact, university clients often make comments such as "now I know more about what I really want and how difficult it will be to build" and "I now know that this idea isn't really feasible." (When student compassion is not a factor, however, client success is a strong predictor of both adoption and non-adoption as will be seen in the discussion, below, of software development projects in industry.)

Introduction of scorecards in 1997-1998, with the attendant pre-vetting, by faculty, of potential clients, and awareness on the part of developers of the relevant issues had the following results:

Year	Weighted Score	notes
1996-1997	0.08	No critical factors were used for vetting projects
1997-1998	0.40	5 client-related critical factors were used to filter potential clients
1998-1999	0.50	4 additional non client-related critical factors used to filter
1999-2000	0.42	Smaller number of projects this year
2000-2001	0.50	
2001-2002	0.50	
2002-2003	0.72	Students explicitly track and use critical factors during development

Table 2

The weighted score is calculated by assigning 1 point for a partially adopted project, 2 points for an adopted project, and no points for a non-adopted project and then dividing by 2 times the number of projects (this is a weighted average favoring adopted projects). The more adopted and partially adopted projects the closer the score is to 1.

3. Industry Study

Encouraged by the university study, we embarked upon an industry study in 2002. For the industry study, three additional factors were introduced and the assessment of factor satisfaction was upgraded from informal conversations with clients and developers to the more formal approach of administering survey instruments to members of the client organizations.

The new factors extend beyond just the client as a key stakeholder, to all key stakeholders. They are:

10. Shared vision: A shared vision is deemed to exist if the following are created and maintained by all key stakeholders:
    - -the (organizational) goals and objectives for the software product under development
    - -the details of the project whereby the product is being developed

(A shared vision should include: the identity of the target customer, a statement of need or opportunity, and a compelling reason to develop or to buy/license and customize the product.)

11. Business Case: A product's/project's business case is deemed to be satisfactory if it justifies the project based upon qualitative and quantitative analysis, both financial and other-than-financial:
    - -The financial analysis may include: ROI estimates, NPV, break-even, payback, and projected profit and loss from investment
    - -The other-than-financial analysis may include: quality improvements, response time improvements, ease of use improvements, and productivity improvements
12. Win-Win Stakeholder agreement: A software development project is deemed to be driven by Win-Win stakeholder agreement if:
    - -all critical stakeholders are identified at the start of the project
    - -the initial shared vision is agreed to in such way as to maximize the number of critical stakeholders' high priority win conditions that are met
    - -any issues or conflicts that arise during the course of the project are resolved in such way as to maximize the number of critical stakeholders' high priority win conditions that are met

In the USC study, the researchers, CS577 faculty, were participants – on the order of department managers – in the development projects studied, so they had direct access to developers during the development process, and to clients both during and after development was finished.

In the industry study, development projects were studied retrospectively, i.e., after their completion. Participants in the industry study were chosen randomly from the membership of the advisory board of the Department of Information Technology Management at the University of Hawaii at Manoa. The organizations they represent cover the following industries: banking, education, finance, healthcare, hospitality, tourism, and public utilities. The individuals involved included CIOs, MIS Managers, Directors of IT, Assistant Superintendents, and an application manager within an outsourced IT department. In each case, the industry participant was deemed the investigated project's client because s/he was in charge of acquiring the software product in question and had a hand in determining the schedule, budget, and cost of the project. In all cases, the project/product studied was deemed, by the client, to have successfully produced the desired software.

The research methodology chosen for the industry study was the use of survey instruments. Each of the twelve identified factors (see above) was represented by at least three survey questions. The following are examples:

DOMAIN KNOWLEDGEABLE CLIENT

1. The client was uncertain about one or more of the following aspects of the product:
    a. The rationale for the product
    b. The goals of the product
    c. The business processes that the product encompasses
    d. The impact of the product on the organization
    e. The necessary conditions for the product to provide value

2. The client was familiar with the existing OR expected user workflows needed to acquire or develop a software solution.
3. The client was familiar with the business processes, policies, and user workflows of the existing system automated or manual.
4. The client anticipated what changes in the business processes were feasible should the need arise in order to acquire or develop the software product.
5. The client was familiar with the starting point for the project.
   e.g. the previous automated or manual system, the lack of a system, the resources available to build the system?

As a safety check, one question for each factor is pitched on the negative side of the other questions. For example, see the first question (above) for Domain Knowledgeable Client.

In all cases, clients responded to the survey questions in the context of an interview by one of the researchers; some interviews were conducted in-person, and others over the phone. In the first two interviews client responses were open-ended, with the result that it was difficult to get voluble respondents to come to the point. In the remaining interviews, respondents were given a choice of six options for each question: strongly agree, agree, disagree, strongly disagree, don't know and not applicable. Questions were re-written, for clarity, as the interviews progressed, with the idea of refining them to the point at which a face-to-face interview would, eventually, not be necessary.

Besides the problem of ambiguously-worded questions, a problem encountered in the development of any survey instrument, and one that we believe has been largely solved, the pilot industry study was subject to two additional problems:

-complex organizational structure and the attendant difficulty of getting by with a single client
-respondent error, accidental or not (the "halo" effect).

The pilot study did not take into account the phenomenon of cross-functional structure. "Cross-functional structure" means temporary teams, consisting of members of different units of the client organization, selected to bring knowledge of their specific areas of expertise to the project. In this type of structure there are lateral lines of communication which are not familiar in a strictly bureaucratic structure; while this type of structure has definite advantages, it creates multiple overlays of authority and influence. Authority is divided or shared among multiple project managers with the attendant problem, from the research project's point of view, of selecting a representative client.

In a cross-functional structure, the interpretation of questions like the following is problematic:

-Did someone other than the client have control over whether or not the product was accepted?
-Did s/he influence the decision to use it?

In a cross-functional organization, the client would have to answer yes to both of these questions, because s/he did not have sole authority to mandate the use of the product. S/he may have fit our

definition of "client" because s/he was charged with acquiring the software and knew the situation that called for it, but thereafter s/he shared authority – or had no real authority -- for the duration of the development project. In this case, "representativeness" of the client is muddied. The following additional questions remain, in the case of cross-functional development, for the continuation of the research presented here:

-Should "the client" consist of one member of each sub-team?
-If so, how should the survey questions be posed?
-and, of much greater interest, do cross-functional overlays of authority and influence pose a threat to software adoption.?

An additional problem not encountered in the university study is the "halo effect." In its simplest version the halo effect is when the client wants to re-interpret history to match the end results. For example, if the project was adopted, the client will say the project was a success and re-interpret the project factors in a positive light regardless of the facts. If the project was not adopted, all the factors may be negatively colored in order to rationalize the non-adoption. In the pilot industry study this appeared to sometimes happen when the client was at the level of CIO. In a number of cases, when the CIO-level client had been too far from the day-to-day execution of the project under investigation, the researchers were referred to the project manager, who appeared fearful of divulging all relevant information lest it find its way back to his/her boss. Cross validation, with both negatively- and positively-phrased questions often suggests that this phenomenon is occurring. When it was detected, or when responses appeared to be less than completely consistent, respondents were phoned or emailed for clarification. Results of the industry study are shown in Figure 3.

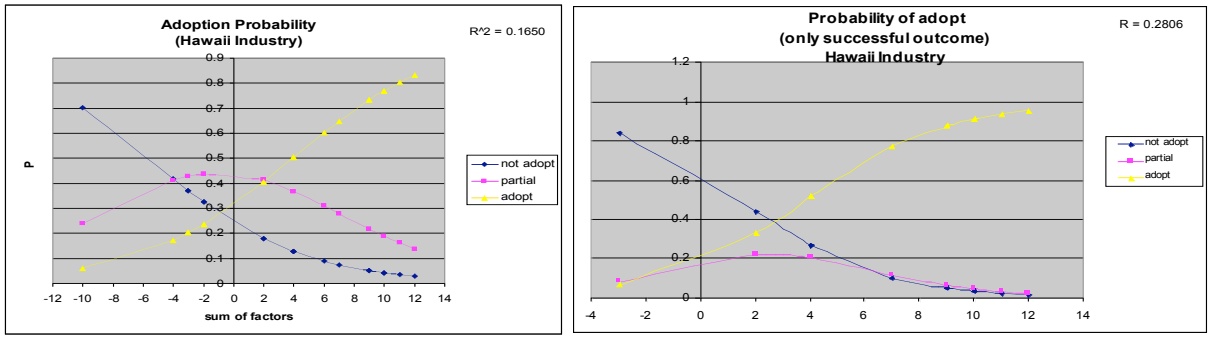

Figure 3a: Hawaii industry adoption prediction, Figure 3b: only "client success" projects

Figure 2 shows the model of likelihood of client success given a sum of factors. This model indicates that the sum of factors is a better predictor of this than adoption. This to some extent reflects the aforementioned "halo effect" where the factors are biased towards the perceived outcome. Nonetheless it is remarkable how "critical" the sum of factors appears to be. To get a good chance of success there need be at least 7 more factors satisfied than unsatisfied. If there are more than 4 factors unsatisfied factors than satisfied, chances are the client will say the project was not a success. As indicated earlier, without the "student compassion" effect, client success has a strong correlation with both adoption and non-adoption. This can be seen by correlating the

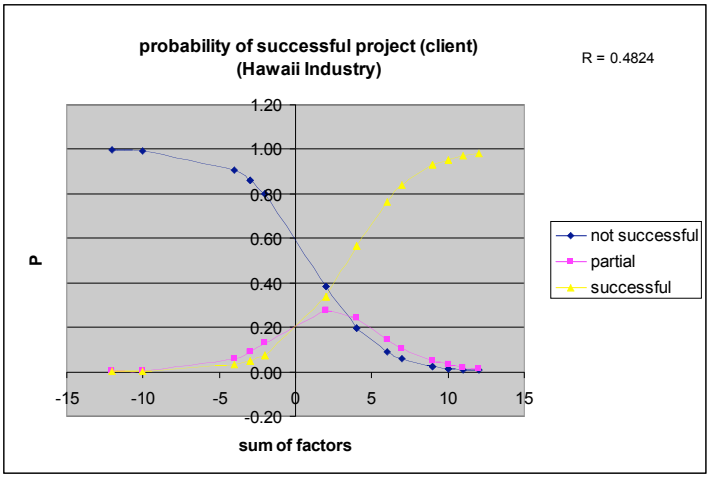

Figure 4: Hawaii industry client success prediction

In summary, results from the university study show that:

-if ten or more of the factors are satisfied, then the software is adopted

-if seven or fewer of the factors are satisfied, then the software is not adopted

-there is no subsumption of factors, that is, *any* ten will do for adoption and lack of satisfaction of *any* seven results in non-adoption

The industry results show considerably more play, with no clear threshold. Although the industry study is just a pilot study, and the application of the identified factors merits considerable additional study, our initial hypothesis regarding the difference is that:

-CS577 projects have a hard deadline – the end of the academic year; if a CS577 software product is not adopted at that point, there is little or no provision – no budget and no available developers – to tweak the software or the environment, however close to ready, in order to push the product over the nearly-accessible adoption threshold.

-Industry projects can usually, though not always, be given a schedule or budget extension to turn an expensive failure into an only slightly more costly success.

References

[Standish 2004] Standish Group, Chaos Chronicles, West Yarmouth, MA, 2004

[Beck 2004] Beck, Kent, Extreme Programming Explained, Addison-Wesley, 2004

[Crispin 2002] Crispin, Lisa, Testing Extreme Programming, Addison-Wesley, 2002

[Jeffries 2000] Jeffries, Ron, Extreme Programming Installed, Addison-Wesley, 2000

[Boehm et al 2000] Boehm, B., Abts, C., Brown, A.W., Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, D. and Steece, B., Software Cost Estimation with COCOMO II, Prentice Hall, 2000.

[Boehm,Turner 2004] Boehm, B., and R. Turner. Balancing Agility and Discipline: A Guide for the Perplexed

[Paulk 1995] Paulk, Mark C., Charles V. Weber, Bill Curtis, Mary Beth Chrissis, The Capability Maturity Model: Guidelines for Improving the Software Process, Addison-Wesley, 1995.

[Boem et al 1999] Boehm, Egyed, Port, Shah, Kwan, Madachy, "A Stakeholder Win-Win Approach to Software Engineering Education", *Annals of Software Engineering*, April 1999, http://sunset.usc.edu/TechRpts/Papers/usccse98-511/usccse98-511.pdf