Rationalism and improvisation: An investigation of requirements engineering in a COTS selection environment

RATIONALISM AND IMPROVISATION: AN INVESTIGATION OF REQUIREMENTS ENGINEERING IN A COTS SELECTION ENVIRONMENT

Kieran Conboy
Computer Science and Information Systems Dept.
University of Limerick

Michael Lang
Dept. of Accountancy & Finance
NUI, Galway

Telephone: +353 1 6462708
Email: kieran.conboy@accenture.com

Abstract

The use of Commercial-Off-The-Shelf (COTS) software applications has become much more prevalent with the emergence of the Web, and more recently with the explosion of hypermedia. However, using COTS applications does not mean the life cycle can be bypassed. The requirements engineering phase of regular development manifests itself in the form of product selection. Herein are reported the findings of a case study of a Web Information System (WIS) development project which incorporates such a product selection approach.

Within the field of Information Systems (IS) research, much has been made of the gap between academic research and practice. Nowhere is this more evident than in the area of requirements engineering. In practice, requirements engineering methodologies are not executed in the structured, methodical way advocated by researchers. This is not surprising, as the underlying philosophy of most of these methodologies is that systems development is a rational process, whereas in actuality it is more accurately portrayed as creative, somewhat improvised behaviour. It is therefore important to determine if the key issues suggested by the normative view of requirements engineering corresponds with the approaches being used in the real world.

This paper examines requirements engineering practice from a design theory perspective, focusing specifically on the contrast between "rationalism" and "improvisation" and the battle between the need for structure, co-ordination, co-operation, and the restrictions that they impose. Both philosophies have pitfalls inherently associated with them. This paper describes these pitfalls and discusses how these pitfalls were encountered in the case study. The findings of this paper suggest that, although a methodology may be very specific and firmly based in the "rational" paradigm, the rationale may be too simple for the problem to be solved, thereby forcing the users to improvise by privately making decisions. Furthermore, users may still improvise even if they are not forced to. In conclusion, this paper asserts that even where methodologies are developed and revised based on actual experiences in practice, as opposed to academic theory, a wholly methodical approach is difficult to develop and impossible to follow.

1. Introduction

The use of COTS (Commercial-Off-The-Shelf) software applications has become much more prevalent with the emergence of the Web, and more recently with the explosion of hypermedia [3]. The increasing time pressures placed on the life cycle has made choosing a shrink-wrapped solution a tempting choice. "Any approach that carves days or weeks from a development schedule is worth considering" [3]. Also, because many customers do not know what they want, it is often desirable to choose an established product that has been proven in the industry. However, using COTS applications does not mean the life cycle can be bypassed. The requirements engineering phase of regular development manifests itself in the form of product selection. This paper concentrates on the requirements engineering aspect of a COTS selection process in a WIS development environment, as opposed to the entire life cycle.

In practice, requirements engineering methods are not executed in the structured, methodical way advocated by researchers [10]. This is because the underlying philosophy of most of these approaches is that systems development is a rational process, whereas in reality it is more accurately portrayed as "situated action" [18], "improvisation" [2], or "amethodical" behaviour [1]. This gap between academic research and industry is not limited to requirements engineering, but is a phenomenon that is clearly evident across the whole field of ISD. By examining the occurrence of methodical and amethodical pitfalls in this case study, an attempt is made to identify why this gap exists.

2. Systems Development Design Theory

Design theory in general, and in particular within the context of ISD, revolves around two contrasting viewpoints. The act of designing either fits within the logical and rational framework traditionally associated with the "scientific" approach to problem solving, or else is ad hoc and unstructured in nature. This debate is at the heart of IS philosophy where there is a battle between rationalism and improvisation: a battle between the need for structure and the restrictions it imposes.

2.1. The Concept of Rationalism and the "Methodical" Approach

Decartes, the father of Rationalism, set out the four principles which define his philosophy: (1) accept nothing as true which is not clear and distinct; (2) analyse a problem into its parts and discuss it part by part; (3) arrange thoughts from simple to complex as the order of study; and (4), enumerations must be full and complete.

Most of the well-known systems development methodologies (SDMs), – such as SSADM, Information Engineering, and the "Waterfall" SDLC, – are firmly based on the principles of rationalism and have been developed by academics and academic research. Empirical studies, however, reveal that, in reality, SDM usage in industry is rather limited [9,6], with some research estimating this usage as being as low as 6% [5]. A number of major philosophical and pragmatic problems with the use of formalised SDMs in practice have become evident, outlined in the following paragraphs.

Methodology Visibility: Methodologies are tools to get the job done, and ought to be invisible to the task. The general trend within ISD has been an increase in the visibility of tools and methodologies, a problem referred to as the methodology visibility problem. Many users of methodologies became weighed down with the intricate documentation required, instead of focusing on the actual problem at hand [19].

Blindness: The Cartesian laws demand that the problem being solved is well defined and fixed before design starts. There is also a tendency to separate the "ends" and the "means", referring to the finished product, and the method used to get there [16]. In reality, the design "ends" are often undefined and likely to change, and any design "means" that assume otherwise is likely to fail [12]. This results in what Schön coins "blindness", describing the premature selection of a preferred course of action at the exclusion of many possible interpretations.

Accountability: In a time of tight budgets and short deadlines, developers follow a specified method, or at least need to be overtly seen to be following a specified method. This can cause what Parnas & Clements [13] refer to as "faking the process", tied to the concept of accountability, where developers follow a method purely as a politically astute safety mechanism that affords an excuse against failure [7].

Theory Above Practice: Robinson [15] highlights this shortcoming of SDMs, asserting that the relationship between theory and practice has traditionally been "one of theory standing logically above practice, informing

and dictating practice". Few academics have a sufficiently thorough understanding of the context of real-world systems development. Furthermore, few of the methods they devise are adequately tested in live situations [7]. Nevertheless, academic researchers persist in developing new methods that, not surprisingly given the two previous points, are often arcane, impractical, and unworkable.

Definition of Rationality: Stolterman & Russo [17] distinguish between public and private rationality. If a method is viewed as a way to put specific concepts of rationality into practice, it is obvious that the developer will need to understand the rationale, known as "public rationale". However, "private rationality" is the term given to the unique judgments and interpretations that a developer will make. These types of rationality may, and probably will, differ.

2.2. The Concept of Improvisation and the "Amethodical" Approach

Improvisation is the polar-opposite philosophy to the theory of rationalism, and makes the assertion that striving for predictability and control is a paradox in design, since the most valued and desirable characteristic of a design process is creativity. There is a strong view that software development is a very complex problem-solving activity that requires the ultimate in creativity [8], and that ISD therefore resists method and is essentially amethodical [11,1,18]. Ciborra [2] supports this view, referring to an implicit process of improvisation that typifies ISD in practice:

> "*Improvisation is simultaneously rational and unpredictable; planned but emergent; purposeful but opaque; effective but irreflexive; discernible after the fact, but spontaneous in its manifestation*".

Improvisation encourages creativity and guides the hidden rationality of skilled individuals [4]. Of course, absolute improvisation is a potential license for anarchy, but if founded upon competent decision-making or what Ciborra calls "smart improvisation", it can contribute contributes to individual and organisational effectiveness. Although this theory of "improvisation" is more intuitively satisfactory than that of rationalism, it is not without its difficulties, which are presented in the following paragraphs.

Transport of Knowledge: While formalised methodologies can be overly visible, the opposite is the case with more unstructured, amethodical approaches, where transport of knowledge becomes a problem. The improvisation approach depends on continuity, where large volumes of hard-learned design lessons must be passed from one generation to the next if development processes are to become efficient. Whereas inexperienced developers can learn easily by following rational, prescriptive methods, it is more difficult to pass on knowledge if purely improvised approaches are being relied upon.

Shift of Control: Improvisation can also have a negative impact on project management. By shifting the control towards the developers, it becomes more difficult to tell exactly where a project currently stands. Predicting where an implementation is going and how long it will take to get there becomes harder to determine [12].

Bias: A third issue is that whereas rational approaches attempt to eliminate biased judgment or opinion, improvisation can actually encourage bias, by allowing developers to "embrace their biases to the point that alternative views are occluded" or to "inflate the importance of their own point of view at the expense of others" [12].

3. Case Study Findings

This case study is based on a WIS development project conducted by a multinational IT consulting organisation on behalf of a major European energy supply company. In the interests of confidentiality, the pseudonyms ABC Consulting and EnergyCo shall be used to refer to the consulting organisation and the energy supply company respectively.

The purpose of the project being studied was to develop a design plan for a Web-based system to provide EnergyCo with a risk management capability to result in more efficient purchasing of resources, and to adequately prepare for competition in an industry in the process of deregulation. With the constant fluctuation of these resources, namely gas, peat, oil and electricity from outside providers, the system is designed to provide real-time supply and demand information organisation-wide.

This project was selected for a case study because it exhibited many of the classical characteristics of WIS development environments, as regards the pressures of accelerated "Web Time", complexity due to the need to integrate data feeds from multiple legacy systems, and a diversity of professional disciplines in the development team. The project involved a 5-person development team from ABC Consulting, comprising of a project

manager, 2 requirements engineers, and 2 analysts with experience in risk management and energy trading systems. EnergyCo employees who formed an integral part of the team included the 5 heads of the Purchasing, Sales, Risk Management, Production and Finance departments, as well as a large number of subordinates who provided input at various stages of the project.

The methodology in this case was strictly adhered to, and the project manager never felt unsure about how a certain procedure was to be carried out. Therefore, it would be expected that amethodical pitfalls would be non-existent. Although this methodology was very specific and firmly based in the "rational" paradigm, the users often resorted to improvisation and privately made decisions on how certain aspects should or should not be implemented. As such, elements of both the methodical (rational) and amethodical (improvised) approaches were experienced. In this section of the paper, we discuss how the pitfalls of each approach, as described in Section 2, were experienced.

3.1. Pitfalls of a Methodical (Rational) Approach

3.1.1. Methodology Visibility

A first impression of the requirements selection process carried out at in this study would conclude that methodology visibility was very high. 400 page requirements documents were subjected to a seemingly endless reviewing and re-reviewing process for a period of 3 weeks.

Visibility was somewhat pronounced due to the fact that much of the decision-making process involved users who were senior management officials. Busy schedules meant that requests to constantly attend review and status meetings and develop or review documents did result in the need for certain tasks to be called into question from time to time.

However, both the development team and the users involved agreed that every step and procedure throughout the selection process was warranted and valuable, and that the level of intricacy and attention to detail was appropriate. Due to the fact that the organisation was undergoing a major change due to deregulation, many were unclear as to what their future role would entail. By forcing every individual to stringently document every requirement, both of the proposed system and the people using it, every employee had a clear picture of their own requirements and what was expected of them by others. The documents used for the requirements gathering phase were also utilised at later stages in the project, such as in the change management and system training phases. Also, the project manager justified the stringent process as being "vital to maintaining task ownership", where a single user is always responsible for the validity and accuracy of a document, even if it was written by a number of people within the development team. However, the rationale behind task ownership has a direct link to accountability, another pitfall of an amethodical approach and discussed in the next section.

3.1.2. Accountability

Accountability had a massive influence on the selection process, apart from the focus on task ownership discussed above. ABC Consulting had previously conducted the exact same selection process for a UK based client that had faced a very similar deregulation situation at the time. The results of that selection process found that one of the vendors in question was not only a clear winner, but was the only one of the candidates that could have provided an appropriate solution. The only changes to the vendors' products since then were negligible. The development team were aware of this, but the decision was made to proceed with the selection process again, and did not even inform the client that the previous one had occurred.

There were two reasons given for this decision. Firstly, a high level of accountability is always demanded when working for a government body or a state-owned company. Secondly, the ABC Consulting's central methodology discussed in the introduction to this study, had to be followed and prior knowledge of product features is not accounted for in this methodology.

3.1.3. Theory above Practice

There were no gaps between the methodology and the way it was applied to this project. That is to say, everything that the methodology said to do was done. As earlier stated, ABC Consulting methodologies are always followed to the letter of the law. However, it was found during the course of the project that a methodology that was originally considered very detailed and prescriptive did contain gaps in the form of situations that were not adequately handled.

Firstly, an underlying assumption of these methodologies was that the users involved in the selection process had a working knowledge of their environment. A scenario where business processes were undergoing radical change and the proposed system was not replacing an existing one crossed the boundaries of this assumption.

Secondly, the issues that arose during the requirement ratings stage, such as the distinction between a "very poor" requirements offering, an offering which is not available now but may become available in the next product release, and a complete failure to fulfil a requirement were not handled.

In contrast, however, a methodology to handle Requirements Prioritisation, detailed on the central Intranet site, catered for issues such as these, but, because the COTS Selection methodology did not incorporate these steps, they were not used. According to the project manager, whose role is to ensure the methodology is followed, the effort placed on the construction and up-keep of the COTS Selection methodology would probably reflect the small proportion of all projects that actually involves a product selection process. He reflects on the frustration of having to meticulously follow a methodology that is known to be flawed, but not important enough to merit a change:

> "Sometimes it is difficult to encourage a client to participate in mundane activities when they cannot see the point. It is much more difficult to convince them when we don't see the point either"

3.1.4. Definition of Rationality

The only evidence of a conflict between project methodology and a team member's opinion of what that methodology entailed is displayed through the inconsistencies regarding requirements ratings. Because there were no guidelines to differentiate between a "very poor" requirements offering, an offering that is not available now but may become available in the next product release, and a complete failure to fulfil a requirement, each user adopted a method that made sense to them. A prime example of this occurred when a vendor could not fulfil a requirement, but made a commitment to fulfil it if the product was chosen. Some users gave the vendor a high mark, because they trusted the word of the vendor. Others awarded a lower mark because they felt they had to make a distinction between a vendor who may fulfil a requirement in the future and a vendor that can fulfil it now. Others awarded a zero mark because they felt they could not award a mark to something they could not see.

This did not cause any major time delays but as the problem was not discovered until after the first vendor presentation, the quality of the selection process was jeopardised. However, it has to be said that this is not a proper example of public versus private rationality described in the literature. Firstly, it was the members of the development team who represented the client that displayed these inconsistencies. Secondly, as a consistent approach was not clearly stated in the methodology this was a case where users were forced to rely on their own interpretations, and not one where actual misinterpretations occurred.

3.2. Pitfalls of an Amethodical (Improvised) Approach

3.2.1. Shift of Control

Another observation of this study was that different sections and members of the team did not know what was happening outside their own area of responsibility. For example, different members were responsible for the elicitation and documentation of requirements from users in different departments. The situation described below explains the problems caused by a lack of communication between those team members.

According to the project manager, the methodology used was "structured but simple". He stated that the methodology must cater for eliciting requirements from users who are not familiar with the system or a selection process. Therefore, while the methodology did not result in a "free for all", emphasis was placed on aiding the users to get the most complete list of requirements rather than on creating the "ultimate" requirements gathering process which was 100% statistically and quantitatively infallible. Emphasis on aiding the user came in the form of advising on areas to be considered such as usability that might not have been dealt with otherwise. A downside of the methodology was a distinct lack of control over the construction of requirements.

The result of this approach was a very unbalanced list of requirements. For example, a list that fell under the heading of Risk Management contained 455 requirements. In comparison, the Purchasing Department came up with a list of 82 requirements. This is despite the fact that the general consensus within the company is that the system will play a much more vital role within the Purchasing Department than within the Risk Management Department. Table 1 provides a list of figures to support this.

The reason for such an imbalance was the fact the Risk Management Department did not previously exist. Due to a lack of practical experience, the Risk Management users meticulously documented every process and requirement, along with encouragement from the requirements engineering team. Those working in previously existing departments, such as Purchasing, were less stringent, and instead just focused on documenting requirements arising from process changes that would occur after implementation. In hindsight, the project manager realised that this was an issue that was not adequately dealt with.

Table 1: Comparison of Purchasing and Risk Management Depts.

	Purchasing Dept.	**Risk Management Dept.**
No. of Employees	70+	4
Added Value of System (Analysis of Added Value of each Department after System Implementation) *Source: Business & Project Plan Agreement document.*	45%	10%
Percentage of Project Time Allotted to Implement System Features for each Dept. *Source: Business & Project Plan Agreement document.*	55%	5%
Management Priority *Source: Client Invitation to Tender document.*	1st	3rd

A second problem was caused by the fact that the only distinction between requirements was critical from non-critical. It was decided that any product that could not satisfy a critical requirement was automatically dismissed. A major part of the project team's methodology was to rigorously review any requirement that was classified as critical. The initial draft of requirements had 35% listed as critical. The requirements engineer entered each set of discussions with the client with the primary objective of reducing the number of critical requirements. All "should have" requirements were changed leaving just "show-stoppers" as critical. The fact that the requirements engineer was not an expert in the field was a hindrance, and as a result 24% of requirements still remained critical after the final draft. The project manager indicated that it was unrealistic to dismiss a vendor if a critical requirement was not satisfied, when nearly 1 in 4 requirements were considered critical. As a result, concern was raised when vendors could not satisfy a critical requirement, but instead of dismissing them, the development team and users discussed alternative solutions to fulfilling that requirement.

3.2.2. Transport of Knowledge

One of the facts noted in Table 1 is that the users' rationale behind the ratings was not recorded for future reference. This was highlighted as a problem after analysis of ratings awarded to the first vendor was conducted. This is a follow-on effect of the variant marking policies adopted by the users, a problem discussed already under the "Definition of Rationality" pitfall. When it was found that the users were applying different rationale to their marking systems, it was difficult to identify the rationale of each user and to determine exactly what went wrong, when each user's script comprised of a list on numbers without any explanations.

3.2.3. Bias

There was no evidence of bias among the team members' work. This was remarkable given the fact that the whole development team were involved in an identical process in a similar environment previous to this project. Also, the fact that the majority of the team had been working in the Energy Trading and Risk Management area for a number of years, and that the number of these systems in existence can be counted on one hand, suggested that the team would find it difficult to maintain an objective view, given the in-depth knowledge of each system's strengths and weaknesses. The project manager highlighted the restraint required to ensure a valid selection process. A prime example of this was when a vendor's demonstration of its real-time capability looked impressive, all of the development team knew that the system still relied on batch feeds from national power-stations, a fact not made known to the audience. Issues such as these were raised as "outstanding issues" to be

covered at weekly meetings, and when advice was given by the development team, it was intended to be a general statement, and that no single vendor was victimised. This strict approach became more relaxed as the process neared conclusion. As a result, there was no status meeting after the final presentation, as it was already decided that the second vendor presentation had clinched the sale.

4. Conclusions

4.1. Pitfalls of Methodical Approaches

Because blindness only becomes a problem where the objectives change during a project, which did not happen in this case study because the objective was simply to "select a vendor product", it can be said not to apply in this case study. That aside, every other foreseeable pitfall of the methodical approach was experienced.

It is easy to understand why methodologies developed from theory would succumb to these pitfalls. Many researchers develop approaches without the hindsight of practical implementation, and so would be unable to evaluate such factors as the impact of human fallacies. The revelation in this study is that an in-house methodology, created and updated by practical experience alone, not directly influenced by any academic involvement, still fails to overcome these problems. The reason, according to the project manager, is that these problems occur in every project and that they are impossible to overcome.

Two conclusions can be drawn from this study of methodical pitfalls:

- Even an in-house methodology, developed from practice and subjected to repeated validation and revision from numerous projects, did not eliminate methodical pitfalls.

- Managers consider these pitfalls to be a quite natural feature in the implementation of a structured, methodical approach, and efforts to eliminate them are futile. Therefore, no effort was made.

4.2. Pitfalls of Amethodical Approaches

It is remarkable that even though the approach taken in the case study was of a highly methodical nature, two of the three pitfalls of amethodical approaches as earlier outlined were experienced.

There is evidence to suggest that the amethodical pitfalls are encountered for the same reasons methodical ones are. They are simply too difficult to eradicate entirely. When discussing these pitfalls and the reason for their occurrence, the manager again dismissed them as problems that occur as part of every project.

There is evidence to suggest that even though every attempt was made by ABC Consulting to achieve a purely methodical approach, they were trying to achieve an impossible goal. Academic research indicates that most approaches in industry lie somewhere between being purely methodical and purely amethodical in nature. There is a consensus that the advantages and disadvantages of each have driven project managers and method developers to adopt an approach that finds some middle ground between the two. In this case however, there was no conscious decision to find that middle ground. Instead, every attempt was made to adopt a purely methodical approach to such an extent that the project manager considered the approach taken in this project to be purely methodical and structured in nature. However, there are examples to show that each amethodical pitfall was encountered simply because a purely methodical approach was not achieved.

In hindsight, it is clear that amethodical tendencies arose where insufficient controls were in place. However, it is very difficult, perhaps impossible, to predict such problems and to then be able to resolve them. The conclusion to be drawn from this is that in a turbulent development environment with so many variables and the unpredictable nature of human actions, it is impossible to develop a completely methodical approach that can control every situation.

5. References

[1] Baskerville, R., Travis, J. & Truex, D. (1992) Systems without Method: The Impact of New Technologies on Information Systems Development Projects. In Kendall, K. E. et al. (eds), *IFIP Transactions A8, The Impact of Computer Supported Technologies on Information Systems Development,* pp. 241-269. Elsevier Science Publishers (North-Holland).

[2] Ciborra, C. U. (1999) A Theory of Information Systems Based on Improvisation. In Currie, W. L. & Galliers, B. (eds), *Rethinking Management Information Systems,* pp. 136-155. Oxford University Press.

[3] Clapp, J.; Taub, A. (1998) A Management Guide to Software Maintenance in COTS-Based Systems. In: *Electronic Systems Center*, MP 98B0000069 November 1998.

[4] Cusumano, M. A. & Yoffie, D. B. (1998) *Competing on Internet Time / Lessons from Netscape and Its Battle with Microsoft.* New York. The Free Press.

[5] Fitzgerald, B. (1998) An Empirical Investigation into the Adoption of Systems Development Methodologies. In: *Information and Management*, Vol. 34, pp. 317-328.

[6] Fitzgerald, B. (1997) The Use of Systems Development Methodologies in Practice: A Field Study. *Information Systems Journal.* **7**(3), 201-212.

[7] Fitzgerald, G. (1991) Validating New Information Systems Techniques: A Retrospective Analysis. In Nissen, H.-E. et al. (eds), *Information Systems Research: Contemporary Approaches and Emergent Traditions,* pp. 657-672. Elsevier Science Publishers B.V. (North-Holland).

[8] Glass, R. L. (1995) *Software Creativity.* Englewood Cliffs, NJ. Prentice-Hall.

[9] Hardy, C. J., Thompson, J. B. & Edwards, H. M. (1995) The use, limitations and customization of structured systems development methods in the United Kingdom. *Information and Software Technology.* **37**(9), 467-477.

[10] Hooks, I. & Fellows, L. (1998) A Case for Priority - Classifying Requirements. In *Proceedings of International Council on Systems Engineering 8th Annual Symposium*, Vancouver, British Columbia, Canada, July 26-30 1998.

[11] Introna, L. D. & Whitley, E. A. (1997) Against method-ism: exploring the limits of method. *Information Technology & People.* **10**(1), 31-45.

[12] McPhee, K. (1997) *Design theory and software design*. Technical Report TR-96-26. Department of Computing Science, University of Alberta, Edmonton, Canada. May.

[13] Parnas, D. L. & Clements, P. C. (1986) A Rational Design Process: How and Why to Fake it. *IEEE Transactions on Software Engineering.* **12**(2), 251-257.

[14] Paul, R. J. (1994) Why Users Cannot 'Get What They Want'. *International Journal of Manufacturing Systems Design.* **1**(4), 389-394.

[15] Robinson, H. (2001) Reflecting on Research and Practice. *IEEE Software.* **18**(1), 110-112.

[16] Schön, D. A. (1984) *The reflective practitioner : how professionals think in action.* New York. Basic Books.

[17] Stolterman, E. & Russo, N. (1997) The Paradox of Information Systems Methods: Public and Private Rationality. In *Proceedings of the 5th British Computer Society Conference on Information System Methodologies*, Lancaster, England.

[18] Suchman, L. A. (1987) *Plans and Situated Actions*. Cambridge University Press.

[19] Winograd, T; Flores, F. (1986) *Understanding Computers and Cognition.* Ablex, 1986.