

# Cost-Benefit Modelling for COTS

Alistair Sutcliffe

*Centre for HCI Design, Department of Computation  
University of Manchester Institute of Science & Technology (UMIST)  
PO Box 88, Manchester, M60 1QD, UK  
ags@co.umist.ac.uk*

## Abstract

*COTS procurement processes do not consider the costs imposed on users when they have to customise products. This paper presents a framework for investigating the costs and perceived benefits for users in accepting COTS products and argues that cost-benefit analysis from the user perspective should be an important part of the product procurement process. The framework describes different types of user costs and contrasts these with user motivations to use the product, based on the fit with their requirements and other perceived rewards. A set of principles summarise how products and the procurement process may be organised to maximise user customisation costs-reward trade-offs.*

## 1. Introduction

COTS products inevitably impose a customisation load on users because the principle of “one size fits all” obliges them to configure the product to suit their needs; the alternative is to accept the compromise of a less than ideal fit to their requirements.

A method for COTS-based RE was described in our previous work [1, 2] which argued for a constraint satisfaction approach for matching users’ requirements to product properties using House of Quality [3] techniques. Because the goodness-of-fit for any one user will be less than ideal, most COTS products are adaptable to enable users to change the functionality and user interface to suit their purposes. However, the range of adaptability provided by a general product will depend on the software developer’s view of the range of potential users. This paper doesn’t enter into that debate; rather, it concentrates on proposing a framework for anticipating the user costs of accepting a COTS product and how these costs might be reduced by RE processes.

Customisation costs should become one of the variables taken into account when selecting products, because a product which appears to have a good fit with user requirements may well be rejected if it requires the user to undertake considerable customisation effort to realise those requirements. Methods for cost estimation

for COTS have been developed from the COCOMO model [4], while cost trade-offs for maintenance or replacement in COTS systems have been modelled by Abts [5] who demonstrated the effect of module size on potential maintenance problems. This paper extends the cost modelling concepts by introducing cost of customisation from an end user’s viewpoint.

## 2. A Framework for Estimating Customisation Costs

The aim for all design is to achieve an optimal fit between the product and the requirements of the customer population. COTS products inevitably impose some burden on customers in configuring the product to their needs. Researchers in Human Computer Interaction have proposed frameworks for modelling the costs and benefits of software from a user’s perspective [6, 7], which point out that effective use is a balance between achieving the user’s goal and the effort required to use a product. Technology acceptance models also point out the trade-off between utility, ease of use and user satisfaction. Leaving aside the problems of market development and customer education, generally the better the fit between users’ needs and application functionality, the greater the users’ satisfaction. Product fit can be summarised in the law of user satisfaction:

*The user satisfaction supplied by a general product will be inversely proportional to its complexity.*

The temptation in COTS development is to make products complex so they can be sold into many different market sectors. Another driving force is responding to a large customer base who request many different modifications. Consequently many applications, and especially Microsoft products, suffer from functional bloat because the requirements process is driven by bug fixes and modification requests from a minority of power users rather than any systematic analysis of what the majority of people want. Applications also suffer from their legacy. Even if Microsoft wanted to simplify Office products, as they have with cut-down word processors, most people still identify with the product that is most familiar, even if

it is worse than the competitors'. The implications are that complex general products tend to make us frustrated because we cannot easily get them to do what we want. Furthermore, the more different types of user there are in a population (gender, age, different cultures) the harder it is for a general product to suit all. The reaction of designers is to strive for adaptable products, but that just increases complexity either overtly by giving users extra work filling in extensive profiles, or covertly by intelligent adaptive interfaces which rarely guess exactly what we want. The second law summarises the effort users will be willing to devote to improving the fit between their requirements and the product:

*The effort a user will devote to customising a software product is inversely proportional to its complexity.*

Complexity may be measured by counts of functional requirements or function points in an implemented system. This implication is that the more complex a product is, the more effort we have to devote to customising palettes, user profiles, setting parameters, etc. Adaptation costs can be estimated as follows:

- $C^{requ}$  = analysing the requirements for personalisation/adaptation
- $C^{acquis}$  = acquiring the necessary parameters for adaptation
- $C^{load}$  = time taken to load the parameters into the system
- $C^{test}$  = time taken to test the effect of different parameters

The total adaptation cost is therefore:

$$C^{adapt} = C^{requ} + C^{acquis} + C^{load} + C^{test} \cdot (Np \cdot Ug)$$

where  $Np$  is the number of parameters (or functions) to be customised, multiplied by  $Ug$ , the number of different users of the target product. These parameters will interact since many users will share a core set of functions, so the variables need to be based on an estimate of the diversity in the user population.

The key to encouraging users to customise products is their motivation. Motivation is a complex subject, but it can be simplified into a set of long-term user goals. Motivations are related to how we perceive the potential benefits of using the product. We can formulate this relationship as a third law:

*User effort expended in customising software will be proportional to the perceived benefit of a product in achieving a job of work or entertainment.*

This implies that part of the requirements problem is not only to specify functionality but also to explain it in an attractive manner so people want to use the function. People will devote considerable effort to learning how to use a product even if it is poorly designed, so long as they are motivated. Classical models of motivation [8] point towards several layers of motivation directed towards

different rewards, ranging from basic needs (hunger, sex, possessions) to higher order self esteem and altruism. The overall trade-off is between users' motivation, their costs and the satisfaction or frustration from the experience of use as perceived benefits are converted into actual benefits, or possibly adverse reaction to the product, as follows:

$$S^{accept} f + (B^{perc} + B^{act}) - (C^{user} + Neg)$$

where  $C^{user}$  = costs are taken from the previous formula  
 $B^{perc}$  = perceived benefits  
 $B^{act}$  = actual realised benefits  
 $Neg$  = negative experience, and  
 $S^{accept}$  expresses either a positive or negative rating of the application's suitability by the user population.

Perceived benefits can be estimated using the motivation model. In work-related products, support of the user's task will be the strongest motivation; however, other motivations such as ability to control systems (power) may play an important role. In entertainment, and in COTS products where use is discretionary, i.e. the home and general public market, motivations may include curiosity, fun, self-esteem from possession, and self-actualisation, the ability to explore and do new things. Motivation can be measured by asking users to rate a product using questionnaires or estimated by expert judgement.

Teenagers already spend considerable time learning and adapting games software that rewards them with excitement; however, most business software only rewards us by getting a job done. If the effort expended in customising software is not motivated by a direct reward, we tend to resent this imposition on our time. Our motivation will therefore depend critically on perceived utility and then the actual utility payoff. For work-related applications we are likely to spend time customising and configuring software only if we are confident that it will empower our work, save time on the job and raise productivity. Unfortunately we rarely have any confidence that customising a product will help.

For personal productivity tools, the motivation may be higher than for more general purpose business software which is regarded as part of the job. Technology acceptance models [9] point out that decisions to adopt technology depend on utility, usability, and possibly fun. Given that fun is going to be limited in many work domains, utility and ease of use will be critical for competitive advantage. This will add a new dimension to non-functional requirements, since many applications in the future will not only have to be usable but will also need to be attractive. Requirements for aesthetic design, visual style and image will need to be considered.

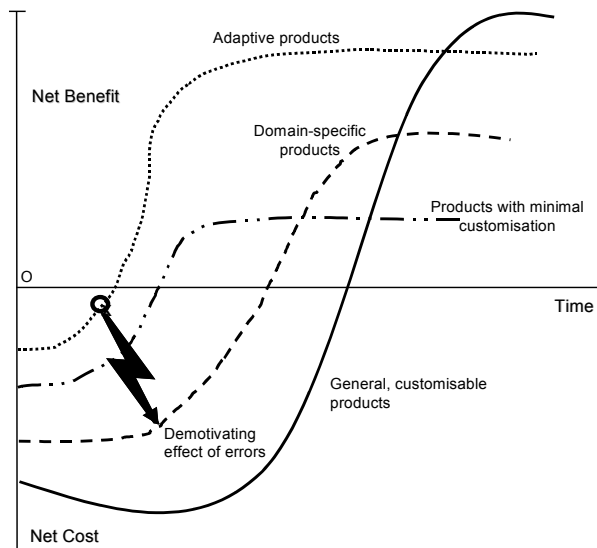
Adaptation causes users work. Software should therefore spare us the burden. Intelligent agents can automatically detect our requirements and provide

appropriate services. Unfortunately adaptation is a two-edged sword. It is fine so long as the adaptation is accurate and fits with our requirements, but when the machine makes mistakes and adapts in the wrong direction it inflicts a worse penalty. First we have to understand what has happened, because the new dialogue, feature or function is unexpected. Then we have to adapt our behaviour to work around the system's misguided initiative. If this were not bad enough, we also feel resentment at the "damned machine" taking initiative and getting it wrong. This soon gives an adverse image of the computer and lowers our motivation to use it effectively. Anyone who has experienced the wretched Microsoft paper clip "office assistant" will know what I mean. This leads to the fourth law:

*The acceptability of adaptation is inversely proportional to system errors in adaptation.*

Because inappropriate adaptation inflicts this triple penalty on our motivation (cost of diagnosing mistakes, cost of working around, and negative emotions about systems usurping human roles), adaptation cannot afford to be wrong. Unfortunately, adaptation is one of the most difficult problems for machines to model. Even if the system could gather perfect knowledge of its environment and had perfect interpretation, other (human) agents may change their minds about their requirements. Such change cannot be anticipated short of clairvoyance. Some system mistakes are therefore inevitable in adaptation.

The cost-benefit profiles for different COTS-customisation approaches is illustrated in figure 1.



**Figure 1. Cost-benefit profiles for COTS customisation.**

General purpose and more complex products have a longer learning curve, so the cost-benefit balance is negative for a long time period. Users have to be well motivated initially and their motivation maintained during the training period. Domain-specific products should have a more rapid learning time since the complexities of a large number of customisation parameters can be avoided; furthermore, once user competence is attained the rewards accrue rapidly. However, domain-specific products are by definition limited to one domain, so there may be a plateau effect on reward. This may not be important for users with a single domain that does not evolve; however, most applications face changing requirements, and domain-specific applications can become limited in a short time if they are updated by new releases. The level of user reward is a function of the ability to change the product to the user's wishes and this is inevitably limited to functions already programmed into the product. The level of effort depends on complexity; as more customisable features are provided, complexity increases, so customisation present a dilemma: more effort for more reward in the long term, versus less effort and quicker short-term gains. Most users don't use customisation facilities in most products, so the effort-reward trade-off does not appear to have been solved for this approach.

Finally, adaptable products lower costs considerably so rewards are perceived quickly; however, this will only be realised in the absence of error. Early errors are critical. Users' motivation can be destroyed by annoying errors in the early stages of reuse, but if rewards are achieved without mistakes, then the user motivation may enable later errors to be tolerated. This suggests a gradual unfolding of adaptive products or simple applications to build up user motivation.

Balancing costs and benefits will require careful estimates and monitoring of users during the acquisition and acceptance phase of products. Initial trial periods should be monitored to investigate users' perceptions of benefits and their experience, as well as the effort expended in customisation. While some cost will be inevitable there are ways in which the cost-benefit trade-off can be optimised to increase the chances of product selection, and we summarise these as a set of guidelines for developers of COTS products and project managers who control the procurement process:

- Increase user motivation by publicity, demonstrations and explanation of the potential benefits of the new product.
- Invest in training; this will reduce the users' learning costs and help them with customisation.
- Monitor user experience in trial periods and deal with negative experiences as quickly as possible.
- Publicise positive experiences and share these in the user community, to increase motivation.

- Share experience of use, tips on how to operate the product and ways to customise it; these reduce user costs.
- Provide customisation templates and ready-made settings for different user groups.

Some of these guidelines can be implemented by a product website to share experience. By building on the shared experience of a user community, the costs of an individual can be reduced; furthermore, the shared experience can increase individual motivation, especially via tips on how to make the most effective use of a product. Of course, shared experience can be negative if early users have poor experiences. Finally, involving users in the RE and procurement process may also reap benefits. If users feel they are involved and own the solution, this will increase their motivation to accept the product and mitigate customisation costs.

### 3. Conclusions

The cost of configuration and customisation has been a neglected area in COTS. The framework proposed in this paper extends current cost-modelling methods (e.g. COCOTS [4]) and points to the need to consider the customer-user perspective of effort and reward. The framework is a first step in providing a systematic approach to estimating costs and benefits from the user's (rather than the procurer's) perspective, although further work is necessary to calibrate the models. Our analysis indicates the importance of connecting user motivation to the perceived reward of using COTS products. User motivation requires considerable research since it will vary by the domain, and by how it is influenced through promotion, training, or functionality embedded in the tool (e.g. wizards, tutors, reuse faculties). The balance between cost and benefit suggests a graded exposure to complexity as recommended in Carroll's minimal manual and training wheels approach [10] that exposes users to simple examples and limited functionality, first to establish confidence and reduce errors. Early reinforcement of motivation will enable users to climb over the hump of effort into benefit. In our future work we will investigate how motivation theory, cost modelling drawn from the HCI perspective and marketing models can influence not only the design of COTS software but also presentation of COTS as an integrated package with training, promotion and user support.

### Acknowledgements

This work was partially support by the EU 5th framework programme Network of Excellence EUD (End-user Development) Net.

### References

- [1] A.G. Sutcliffe, "A Technique Combination Approach to Requirements Engineering," *ISRE '97: 3rd IEEE International Symposium on Requirements Engineering*, pp. 65-74, Los Alamitos, CA: IEEE Computer Society Press, 1997.
- [2] A.G. Sutcliffe, *User-Centred Requirements Engineering*, London: Springer-Verlag, 2002.
- [3] J. Hauser and D. Clausing, "The House of Quality," *Harvard Business Review*, 1988, pp. 63-73.
- [4] C. Abts, B. Boehm and B. Clark, "COCOTS: A COTS software integration cost model", *Proceedings of ESCOM-SCOPE 2000 Conference, Munich, Germany, 2000*.
- [5] C. Abts, "COTS-based systems (CBS) functional density: A heuristics for better CBS design". *Proceedings of International Conferences on COTS-based Software Systems, ICCBSS*, pp. 1-9, Berlin: Springer Verlag, 2002.
- [6] P. Timmer and J. Long, "Separating User Knowledge of Domain and Device: A Framework," *Proceedings of the HCI 97 Conference*, pp. 379- 396, Berlin: Springer-Verlag, 1997.
- [7] J. Dowell and J. Long, "A Conception of the Cognitive Engineering Design Problem," *Ergonomics*, 1998, pp. 126-139.
- [8] A.H. Maslow, R. Frager, C. McReynolds, R. Cox and J. Fadiman, *Motivation and Personality*, New York: Addison Wesley-Longman, 1987.
- [9] F.D. Davis, "User Acceptance of Information Technology: System Characteristics, User Perceptions and Behavioral Impacts," *International Journal of Man-Machine Studies*, 1993, pp. 475-487.
- [10] J.M. Carroll, *The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill*, Cambridge MA: MIT Press, 1990.