Going beyond ‘misfit’ as a reason for ERP package customisation
Ben Light

Abstract

In an attempt to deal with the potential problems presented by existing information systems, a shift towards the implementation of ERP packages has been witnessed. The common view, particularly the one espoused by vendors, is that ERP packages are most successfully implemented when the standard model is adopted. Yet, despite this, customisation activity still occurs reportedly due to misalignment between the functionality of the package and the requirements of those in the implementing organisation. However, it is recognised that systems development and organisational decision-making are activities influenced by the perspectives of the various groups and individuals involved in the process. Thus, as customisation can be seen as part of systems development, and has to be decided upon, it should be thought about in the same way. In this study, two ERP projects are used to examine different reasons why customisation might take place. These reasons are then built upon through reference to the ERP and more general packaged software literature. The study suggests that whilst a common reason for customising ERP packages might be concerned with functionality misfits, it is important to look further into why these may occur, as there are clearly other reasons for customisation stemming from the multiplicity of social groups involved in the process.

1. Introduction

The purpose of this paper is to explore why enterprise resource planning (ERP) packages might be customised. Packages are now widely adopted on the understanding that they make a significant contribution to resolving the problems of existing information systems. ERP packages supposedly offer benefits such as: improved software structure [1,2]; a readily available skills base [3–5]; reduced levels of system entropy, through the use of a pre-built, tried and tested, product [1,2,4]; a common IT and organisational blueprint in order to deal with globalisation and/or previous merger and acquisition activity [6] and the opportunity to adopt perceived best practice business models embedded within the package through reengineering activity [7,8]. Yet,
even in the light of these ‘benefits’, customisation still occurs. A number of studies highlight that, for those in some organisations, the ‘inscribed’ [9] standard model is not satisfactory and therefore customisation may occur [10–15]. Thus, the problematic conditions that the ERP package was originally intended to resolve might be recreated. Why then, would they want to take this action?

By way of introduction, it is suggested that ERP customisation can be thought of in terms of systems development and organisational decision-making. Custom development and organisational decision-making are known as less than rational activities and thus, customisation has to be conceptualised as an activity that is ‘flexibly interpreted’ [16]. Two ERP projects, where customisations have been performed, are then used to illustrate some of the reasons why customisations might be performed. These reasons are then built upon through reference to the ERP and the more general packaged software literature. In summary, it is argued that whilst a common reason for customising ERP packages might be concerned with functionality misfits [12], it is necessary to examine more deeply why these might occur, and indeed, if other reasons exist.

2. ERP package customisation and ‘traditional’ systems development ideas

Early systems development efforts were relatively unstructured. Lehman [17] for instance, argues that for 1950’s programmers, the ecstasy of instructing a machine to undertake computations at speed overshadowed the rather dull need for a guiding theory and discipline. However, as those in organisations increased their reliance information systems—structured programming, analysis and design methods took on more importance. Some state the reported reasoning was that a perfect system would be produced if a logical and structured procedure were followed [18,19]. Underpinning this of course is the belief that a ‘perfect’ system can be produced.

Similarly, as ideas on systems development evolved some researchers have suggested that it was felt that the only way of achieving the perfect system was to involve end users [18]. Consequently much effort has, and continues to be spent upon rationalising the process of custom systems development and involving end users, in an attempt to build the perfect system. However, despite the process of systems development being historically characterised as inherently rational it has also been argued this might not be the case [18–20]. For example, developers may have limited experience or knowledge and their own agenda [21–23]. Moreover, in terms of end users, they are often assumed to have a shared commitment to any given systems development when, as with developers, this is often not the case [24]. End users will have varying personal or group agendas, levels of interest and degrees of power in systems development efforts [25–27]. Clearly then, there are multiple realities and hitherto potential interpretations of rationales in existence. Custom system development is not a rational and linear process. Indeed, it has been characterised, quite some time ago, as whilst holding the potential to be explained by rational motives, also being inherently influenced by politics and power [25,28]. Even though packaged software may display differences to custom development there are still developers, users, the (flexible) use of methodologies, and of course the aim is usually the creation of an information system. Thus, if ERP package customisation is considered part of systems development, it is difficult to argue that it is singularly rationally motivated. Customisation also has to be decided upon, and this is the focus of the next section.

3. ERP customisation as a process of decision-making

Decision-making, the process of choosing between alternatives, is based on the idea that if there are no alternatives, there is no decision to make. Decision-making theories fall broadly into two camps, normative and descriptive [29]. Normative models emphasise rationality and assert how decisions should be made whereas descriptive models set out to explain how decisions are made [30]. Normative models assume that decision makers: have a sound basis of knowledge available to them, so they can know the alternatives and successfully predict a variety of outcomes; have excellent judgement so they can rank the outcomes in terms of value and are rational in the way they make decisions [31]. However, there is a
great deal of research, which questions rational models of decision-making [32]. For example, it is rarely the case that decision makers have complete information about alternatives and the use of intuition is more prevalent that may be reported [33–35]. Moreover, preferences are seldom rigid and may change in the light of experience [36]. Also, rational models ignore the contexts within which decisions are made, in particular group and organisational influences [37].

In other areas of information systems where decision-making is required, such as systems thinking [38], systems development [39], strategic information systems [40] and strategy planning [41], normative approaches are shown to be flawed. This raises further questions regarding the appropriateness of viewing the customisation of ERP packages in a rationalistic fashion. Moreover, a popular conceptualisation of decision-making in the descriptive tradition, the ‘garbage can model’ [42] suggests: there is typically more than one problem (others may bring different problems to the table); solutions are rarely created in the decision-making process (people come to the table with ‘the’ solution, and the one that may suit them); Who is involved and not involved in the decision-making process may affect the outcomes and the extent to which those who are involved, are involved, may also affect outcomes. cf. [31] and [43]. The garbage can model implies multiple realities and therefore decision-making becomes less straightforward than rational models suggest. The process is clearly subject to flexible interpretation by the presence of various relevant social groups [16]. Therefore, conceptualising the decision to customise ERP packages as a simple economic matter of addressing function misfits becomes increasingly problematical.

4. Research method

The approach of this study was to compile case studies of the customisation of ERP packages. The aim was to identify different reasons for the customisation of ERP packages. Thus, a qualitative case study research strategy was employed as the subject of the study poses context, content and process questions [44,45]. Context in terms of why customisations took place, content in terms of the nature of the customisations and process in terms of how this was done. Paying attention to these three areas arguably gave a richer understanding of the reasoning for customisations. This is because why something was done in the two ERP projects was inextricably linked with questions of what could be done and how. The two organisations used in the study have been given pseudonyms—Cable and Home and form part of a wider study of the customisation of ERP packages. The data were collected via formal and informal interviews with various people involved in the ERP projects, review of the ERP packages whilst operating in the organisations and project documentation. The formal interviews were undertaken at each implementation milestone (approximately one per month), and the information reporting of customisation activity supplemented this. The data from both ERP project sites were collected over 1 year.

In addition to the case studies, secondary research was conducted to identify further reasons for ERP package customisation. Whilst a comprehensive range of reasons were identified by the primary research, the author was aware of other insights from work already published in the area, although not specifically with this study’s research objective in mind. Thus, the literatures on packaged software and ERP were drawn upon to construct further potential reasons for customisation and supplement the primary research. Nevertheless, it is important to note that primary and secondary research combined do not represent ‘the’ list of all potential reasons for ERP customisation. The point is merely to demonstrate that there are other reasons for this taking place, other than achieving function-fit.

5. Illustrative customisations at Cable and Home

This section presents interpretations of the data and findings of the primary research. For each organisation, the reported drivers for adopting an ERP package, details of the customisations and the reasons for the customisations are provided. At the end of each, a summary table of the reasons for customisation is provided – see Tables 3 and 4. Both project teams in the organisations undertook an extensive package evaluation exercise to ‘ensure’ they
implemented software that matched their requirements. In each case, the package implemented was that which the companies perceived as the one that mostly closely matched their requirements.

5.1. Cable

The Cable organisation comprises sites throughout Europe, America and the Asia Pacific region. The organisation grew by acquisition and as a result, the many business units functioned in individual ways. There was proliferation of systems (79 in total) and consequently, there was inconsistent terminology, coding structures, business processes and information systems practices. The senior management of Cable therefore decided that they wanted to implement an ERP package to enable them to migrate the existing systems to a common platform.

5.2. Illustrative customisation points

5.2.1. Metal pricing

One of the materials used within cable production is copper. Copper prices vary daily and there is a policy aimed at eliminating the risk involved in buying and stocking products whose value fluctuates. Consequently, products are sold based on the amount of copper they contain, valued at the price of its purchase. To manage this, data regarding the copper weight contained within each finished cable and the price of that copper when it was purchased is needed. In addition, as key customers could eliminate their risk by reserving a quantity of copper (a book) at a specific price for use in the manufacture of their orders, data about the contract price and the rate of consumption of copper books was needed too. The ERP package did not include this functionality and thus a customisation was performed. This functionality was eventually incorporated into the standard functionality of the package by the vendor, as they realised that the principles it embedded had wider applicability. It is interesting to note that another company wanted to try and change certain aspects of the functionality at a later date but the ERP vendor did not allow this. A member of the information systems staff at Cable pointed out that if the change had been made this would have led to difficulties when they upgraded. The package would have needed to be re-customised in order to re-align the software with their ways of work. Here then, customisation was performed because the required functionality was not present in the package as it stood at the time of purchase. Moreover, the vendor’s incorporation of the customisation into the standard product highlights that customisation can fuel product development activities.

5.2.2. Material requirements planning (MRP)

The ERP package did not cater for the needs of a cable company where manufacturing operations require details of product component length. Standard MRP takes top-level demand, totals this demand if required and generates the required production and purchase advices based on the items bill of material (BOM) quantities. Individual sub-components quantities are totalled; existing stock and work-in-progress quantities are netted off against these total quantities to create single orders within the specified time fence. This scenario does not work in the length environment. Length items in the BOM must not be time netted and component order needs to be allocated to the ‘parent order’. For example, if the BOM for 100 m of Cable X includes 100 m of Core Y, 100 m of Core Z and 0.5 kg of Sheath A, can a customer order for 500 m of Cable X be satisfied? The answer is no. A ‘standard’ MRP calculation (the one embedded in the ERP package) would suggest yes, as the calculation would be reported as shown in Table 1 but actual stock levels are as detailed in Table 2.

The ERP vendor developed this customisation and it was then re-integrated into the standard product for so that other customers could use it as necessary. As with metal pricing, the customisation was performed because functionality was missing from the standard

<table>
<thead>
<tr>
<th>Table 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stock levels</strong></td>
</tr>
<tr>
<td>Cable X</td>
</tr>
<tr>
<td>Core Y</td>
</tr>
<tr>
<td>Core Z</td>
</tr>
<tr>
<td>Sheath A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Breakdown of stock levels</strong></td>
</tr>
<tr>
<td>Cable X</td>
</tr>
<tr>
<td>Core Y</td>
</tr>
<tr>
<td>Core Z</td>
</tr>
<tr>
<td>Sheath A</td>
</tr>
</tbody>
</table>
package and, potentially, it fuelled the vendor’s processes of product development.

5.2.3. Corporate document set

Vendors expect external documents to be changed and therefore ERP packages include an existing set of documents which carry the information required in a very basic manner. A number of these documents were customised by information systems staff and external consultants by adding data fields and formatting them into the company ‘house style’. In this case, customisations were performed to make the system look more appealing to various users (including internal staff and customers).

5.2.4. Shop floor control

It became clear during the initial consultations with those on the shop floor that the ERP package’s progress reporting screens were too cluttered and complex. A trial of the new shop floor procedures was undertaken using five people from one work centre—17% of the data entered was inaccurate. In one instance, if the error had been in a real environment the resulting re-work costs would have been nearly £ 4000. A member of the project team said that with time and training this error rate could have been reduced, but it would have been costly. Instead, an ERP package application program interface tool was used to simplify production progress reporting screens. A new screen was developed where minimal data was displayed and entered. The process automation then drove the standard ERP package production progress-reporting screen in the background, using this data. Trials of this new screen reduced the error rate to 8% immediately and virtually eliminated it within 2 weeks. Thus, this customisation was performed to increase the efficiency of the operation of the package. It was also reported as facilitating user acceptance of the package. Ironically, this combination of efficiency and user acceptance led to further efficiency gains, which meant that the user base could be reduced.

5.2.5. Checking facilities

One business unit within Cable operated in a high volume, high cost market with long lead times where many of the customers were deemed high risk. The existing credit facilities within the ERP package automatically checked a customer’s credit rating when a sales order was entered into the system. However, by the time the goods were ready for despatch, this credit rating may have changed. The system did not automatically check for this. Solution one was to write a report showing a customer’s credit rating and introduce business procedures whereby the user checked this report before despatch. Solution two involved adding a second credit check to ERP package so that the customer’s credit rating was automatically checked prior to the despatch of goods and despatch barred for those customers failing. It was decided that the high risk and high cost of sending goods where payment would not then be received meant that an automatic procedure was the preferred option. Customisation was performed in this instance as the ‘best practices’ embedded in the package were perceived by those at Cable as lacking. It was felt that their approach to credit checking was better.

5.2.6. Terminology and layout

In the early stages of implementation a number of requests for the customisation of the application’s field labels were made. For example, change ‘Ref A’ on the sales order header to read ‘Customer’s Purchase Order Number’. These requests contradicted one of Cable’s project management team objectives of creating common terminology across the group, based upon the ERP package terminology. Thus, all requests were refused resulting in a considerable amount of friction between the project team and the lead consultants from the central competency centre. However, as the project progressed and the staff of Cable became more familiar with, and accepting of, ERP terminology requests for changes dropped significantly. While the change was difficult at the time, most staff came to accept that to have customised heavily in this area would have added little value. The change from ‘Ref A’ to ‘Customer’s Purchase Order Number’ did however eventually take place. The change was considered of benefit to the generic business procedures but it was felt that if the change had been made early on in the project it would have opened the floodgates for other requests. This illustrates how customisations might be used as control mechanisms. The customisation that was performed was always felt

---

1 The term ‘shop floor’ is a colloquialism—it refers to the manufacturing function of the business.
Table 3
Summary reasons for customisation at Cable

<table>
<thead>
<tr>
<th>Reason</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metal pricing</td>
<td>The functionality was not present in the package as it stood at the time of purchase.</td>
</tr>
<tr>
<td></td>
<td>To fuel the process of product development.</td>
</tr>
<tr>
<td>Material requirements planning</td>
<td>The functionality was not present in the package as it stood at the time of purchase.</td>
</tr>
<tr>
<td></td>
<td>To fuel the process of product development.</td>
</tr>
<tr>
<td>Corporate document set</td>
<td>To make the system look more appealing and to various users (including internal staff and customers).</td>
</tr>
<tr>
<td>Shop floor control</td>
<td>To increase the efficiency of the operation of the package. To facilitate user acceptance of the package.</td>
</tr>
<tr>
<td>Checking facilities</td>
<td>As the ‘best practices’ embedded in the package were perceived by those at Cable as lacking.</td>
</tr>
<tr>
<td>Terminology and layout</td>
<td>As a control mechanism. The project team ‘gave away’ power to retain the majority of power.</td>
</tr>
</tbody>
</table>

necessary by the project team, but they decided not to allow this to happen in order to ‘stem the flow’ of requests. Moreover, the eventual ‘about turn’ of the project team helped foster relations between them and those in the affected business units as, those in the units felt they had been listened to. This is quite pluralistic in that the project team ‘gave away’ some power in order to retain the majority of power.

5.3. Case 2: Home

Home comprises 20 sites throughout the UK. As with Cable, the existing information systems were incredibly fragmented. Some core systems were shared across many divisions and other systems had been developed independently. Business procedures also differed across sites and producing a corporate view of the organisation was difficult. The ERP package was expected to integrate systems across the group, provide a common source of data, eliminate duplication of effort in data entry and maintenance, replace manual systems without incurring long development timescales, introduce common coding structures, introduce common procedures and provide a corporate view of business information.

5.4. Illustrative customisation points

5.4.1. Pricing

Three elements are taken into account when constructing a price: product, customer and location:

- Each of Home’s 10,000 products has a base price.
- Customers are categorised into eight main customer types depending on volume of sales and perceived value to the organisation.
- The country is divided into 124 regions which are further divided by the first four characters of the postcode. The detailed geographical breakdown of the country is used to better target pricing based around the distance travelled to deliver the product. As transport accounts for 30% of overall cost of sales, as much of that cost is included in the sale price as possible.

The combinations of each of these cost elements led to the generation and maintenance of 26 million different prices. Changes to this pricing structure would have necessitated negotiations with customers and it was considered by the project team to add an unacceptable level of complexity to project. It was decided to customise the ERP package to reflect this pricing structure and to address the renegotiation of the pricing structure post go-live. The customisation was quite extensive and took a total of 5 months to complete using external developers. Here customisation was performed primarily because the ‘implicit business model’ inscribed in the package was incompatible with Home’s pricing policy. However, plans were made to change this structure in line with the ERP package post implementation. Therefore, the customisation was really performed here to facilitate a smoother implementation (although it is arguable whether this was achieved).

5.4.2. Despatch procedures

The despatch procedures inscribed into the ERP package were cumbersome involving the entry of the same data in a number of different sessions. Therefore,
keystroke emulation software was used to automate these procedures. This ran in the background ‘looking’ for new sales orders on the system and it allowed users to concentrate on handling any exceptions thrown out by the automated procedure. Thus, this customisation was performed to simplify the amount of time taken to carry out day-to-day operations. Additionally, the reduction in despatch administration enabled units to reduce the number of staff in that area—thus reducing costs.

5.4.3. Key performance indicators (KPIs)

Home’s management team uses KPI reports to enable them to effectively manage the business. Although there were many standard KPI reports in the package, further ones needed adding. These customisations have added value to the ERP package, improving the visibility of those elements that are deemed important by management to the organisation as a whole.

5.4.4. Haulier self-billing

600 external hauliers work with those at Home to deliver products to customers. These hauliers also represent 65% of the total delivery capacity. The decision was taken to customise the ERP package to enable the continuance of Haulier self-billing. Haulier self-billing existed in the previous system so the procedures were well understood. Also, the despatch staff did not want to take on the additional administration task of manually checking each despatch to confirm that invoices supplied by hauliers were correct.

Automating the production of Haulier invoices from ERP package allowed the continued outsourcing of invoice checking to the haulier. The system produces Haulier statements each month for all hauliers who delivered during the month. The statement shows the date, time, trip number and price of each trip made with a total at the bottom indicating the amount that Home intends to pay. The hauliers then check these values against their records. Queries are handled and any adjustments made in the following month. The customisation has been extended over the last year to include a number of enquiries to make the resolution of haulier queries easier. Hence, customisations were performed here to maintain existing ways of work that were perceived of value.

6. Further potential reasons for ERP package customisation

In this section, further potential reasons for ERP package customisation are raised. A summary of these reasons is provided in Table 5 at the end of this section. From this it can be seen that the reasons for customisation stem from the existence of multiple rationales rooted in various relevant social groups.

6.1. The use of packages as a software blueprint

A number of early studies of packaged software indicate that there was an intention to customise products as a short cut to development—and thereby deal with the applications backlog [46,47]. Indeed, in one study, a whole category of analysis is devoted to problems with product customisation [48]. To clarify this, the subject’s problems were that they were unable to customise the software easily, rather than that they had difficulties following the customisation activity. However, from this point on, and more strongly in the 2000s, packaged software is linked with the minimisation of customisation activity [12,49]. Although many organisations choose to implement packages because of the benefits they offer over custom development, it will be interesting to see how the rhetorics around the non-customisation of packaged software play out long term given the potential issues associated with maintenance as discussed in the next section.
Table 5
Summary of further potential reasons for ERP package customisation

<table>
<thead>
<tr>
<th>Potential Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>The use of packages as a software blueprint</strong></td>
</tr>
<tr>
<td>As a shortcut to custom development whereby the licence is purchased and then the software is customised ‘in-house’ or via other means.</td>
</tr>
<tr>
<td><strong>Customisation as a form of maintenance</strong></td>
</tr>
<tr>
<td>To correct development work in the package ‘as purchased’.</td>
</tr>
<tr>
<td>If the vendor does not develop or maintain the product in a timeframe required by the consumer.</td>
</tr>
<tr>
<td>If consumers cannot influence the broader development of the product.</td>
</tr>
<tr>
<td>If a product is, or becomes, less popular, support for the package may be hard to find.</td>
</tr>
<tr>
<td>If the vendor an organisation has purchased from drops out of the market.</td>
</tr>
<tr>
<td>Overbearing charges might push consumers to make their own arrangements for maintenance.</td>
</tr>
<tr>
<td><strong>The problems of ‘expert advice’</strong></td>
</tr>
<tr>
<td>Those seen as experts might not be and so they advocate customisations where they are not necessary.</td>
</tr>
<tr>
<td><strong>Prior selling activities</strong></td>
</tr>
<tr>
<td>Due to the purchase of a product that does not adequately meet a consumer’s needs as a result of various acts of selling.</td>
</tr>
<tr>
<td><strong>Misconceptions surrounding the package</strong></td>
</tr>
<tr>
<td>When the selection process was enacted, it was not clear what the basis for selection was therefore a product is chosen that does not fit well.</td>
</tr>
<tr>
<td><strong>Shifting expectations of functionality fit</strong></td>
</tr>
<tr>
<td>To accommodate unanticipated, shifting requirements which the vendor might not meet.</td>
</tr>
<tr>
<td><strong>Safeguarding</strong></td>
</tr>
<tr>
<td>As a form of resistance on the part of ‘in-house’ or other relevant information systems personnel. Customisation work would justify their existence.</td>
</tr>
<tr>
<td>More broadly, others might perform customisations in organisations as a form of protection from unwanted changes.</td>
</tr>
<tr>
<td>As a form of ‘external safeguarding’ such as the desire to maintain competitive advantages.</td>
</tr>
</tbody>
</table>

6.2. Customisation as a form of maintenance

Although, packaged software is supposed to be ‘better built’ than custom developed software [50], other studies suggest that there is a lack of rigour in the product development processes of the packaged software product industry [51,52]. Therefore, customisations might be necessary as a form of ‘corrective maintenance’ [53] to deal with shoddy development work in the package ‘as purchased’.

Also, when an agreement with a particular software vendor has been entered into, there may be a problem of path dependency. Those in consumer organisations are effectively committing themselves to upgrading software periodically (and mostly at the behest of the vendor) if they hope to avoid major conversion headaches [1,54]. Therefore, the responsiveness of the vendor and the ability of the user base to influence the development trajectory of a product require consideration. This is similar to custom development where various users at the organisational level deal with the development team in various ways with various outcomes [27,55]. The main difference though is that this negotiation is undertaken in a market environment where the user base is more diverse and diffuse. Indeed, it has been argued that the market represents a barrier between the developer and the user [56]. Thus, if consumers cannot influence the broader development of the product, they may also undertake local customisation work. Andersson and Nilsson [57] also suggest that price competition is less when selling services such as support and maintenance to existing customers and that contracts for these can generate considerable repeat income. Essentially what this means is that packaged software vendors will have little scope for operating a cost based strategy in relation to the initial sale of licences but they will be more successful in generating profit once the customer is locked-in through maintenance and upgrade services because switching costs increase dramatically. Indeed, the hidden costs of support, training, tailoring, maintenance, hardware adjustment, forced upgrade and incremental licensing agreements were bemoaned by 33% of respondents in one study [58]. Maintenance is also a form of revenue generation for the vendor. For example, patches may be provided at no cost, beyond any maintenance agreement. However, changes to smooth out operable, but badly fitting or designed functionality often form the basis of new releases for which additional charges are made [59]. Therefore, as customisation can be a source of revenue for vendors, once a customer is ‘locked-in’, the customer might avoid what they see as overbearing charges, by choosing to make their own arrangements for maintenance. Moreover, even if a customer is happy with the maintenance contract in place,
customisation may be required if the vendor drops out of the market. It is argued that due to the relatively low costs of entry into the industry, the financial stability of some vendors is questionable and a cause for real concern [48]. Recently, in an ERP context, the Baan company has seen considerable difficulties leading to its purchase by SSA Global Technologies Inc. Until recently, SSA policy was that it would not support prior versions of the Baan product [60].

Furthermore, it is argued that software vendors can produce new releases faster than consumers can absorb them [59]. However, earlier, it was argued that acts of maintenance, in this case ‘adaptive/perfective’ [53] are at the behest of the ERP package vendor. The assumption is that these acts will be carried out in time for the ‘consumer base’ when in fact; the software vendor might also represent a form of applications backlog. One example is the increase in popularity of customer relationship management (CRM) in the late 1990s. Although there were major CRM package companies, such as Siebel offering products, many ERP packages had already been implemented that did not offer CRM functionality. Consequently a decision had to be made as to whether a user would wait for a CRM product to be built by their ERP vendor, adopt another package, or customise the existing package. It also follows from this that problems may also emerge if a product is, or becomes, less popular which might mean that the support for the package may be hard to find. For example, the lack of SAP consultants in the late 1990s–early 2000s echoed the reported shortage of assembly program language skills in 1994 [61].

6.3. The problems of ‘expert advice’

Although vendors might be seen as experts [50], they might not be so advise customisations where they are not necessary. As the ERP market grew so quickly, this led to a shortage of competent consultants [3,4]. Those in consumer organisations therefore widely complained about consultants with only a few months training who charged up to US$ 2500 a day [62]. This problem further manifested itself in a widespread lack of knowledge about ERP products, particularly where integration work and partner products were concerned [10]. This issue is not new, nor ERP specific. An earlier packaged software study reported difficulties in engaging users in the implementation process as the development team were perceived by the end users as not possessing adequate knowledge of the product in question [63]. Therefore, customisations might be performed because of a lack of knowledge about a product or a products context because of inexperience or incapacity—cf. Howcroft and Light’s [64] study which highlights the capabilities of various package intermediaries.

Consultants may offer standard solutions to problems that are very specific to the organisations that are employing them—they may not want, or be able to, grasp organisational realities. Consultants may also be viewed as holding too much power, influence and knowledge which may ‘walk out of the door’ when they do [65]. In Skok and Legge’s study, a representative of the company reported that the documentation provided by the consulting group was not tailored to their needs. For example, a costing invoice was called a ‘different outlet’ that did not make much sense to their employees. In contrast, the project team at Guilbert looked into the use of intermediaries to facilitate their ERP implementation process. However, in the end they decided that it was preferable to use internal expertise to enable change management and thus, consultants were only used to assist in the technical configuration of the software [66].

6.4. Prior selling activities

Whilst vendors may control the landscape of selection to a fair degree, they clearly have to respond to market demands to some extent. Yet, vendors and other intermediaries, such as consultants, may attempt to circumvent this by engaging in processes of convincing consumer organisations that their products are the ‘best’ and that they should buy into their vision for the package especially where comparable alternatives exist, a typical feature of such work in management consulting [67,68]. In an IS context, Friedman and Cornford [18] call this ‘salesmanship’, the idea that users may be taught what is good for them and to do things in a new way. Thus, it is possible to argue that, due to the competitive nature of the packaged software market, the selection process may be directly/indirectly or implicitly/explicitly influenced by vendors who bring their own agendas to ‘the table’. For example ‘strong ERP vendor marketing’ and ‘the right solution
and message at the right time’ have been cited as key reasons for its adoption [69]. Similar reasoning has been reported with CRM packages [70]. It has been further suggested that demonstrations of packaged software are only effective in showing the ‘bells and whistles’ and do not fully detail the specifics of the package [48]. This is because the vendor views the demonstration as an opportunity to sell the firm’s image as well as the product, to a buying group that may consist of representatives from various departments. Therefore, the salesperson has to tailor their pitch to what Gross and Ginzberg term ‘the lowest common denominator’. The bells and whistles approach is the most usual one. Another, perhaps more subtle form of selling is that undertaken by people in organisations. At T.Co., various members of the project team helped the vendor to sell their product to senior management by ‘grooming them’ for their sales pitch [64]. In this case, Howcroft and Light argue that the process of selling was elevated above the product being purchased. Therefore, it is not surprising perhaps that those in consumer organisations end up having to implement customisations. They may have been ‘sold’ a product that does not readily meet their needs.

6.5. Misconceptions surrounding the package

It would be expected that those in organisations would usually have some idea of their functional requirements before they begin to assess whether a particular package will fit with these. This is because some required functionality might only be partially met and in others, missing entirely [1]. Moreover, the significance of this is increased in the light of the various selling acts that may occur in the process of selection and procurement. For example, the blurring of packaged software capabilities by vendors:

‘with the huge explosion in the CRM market it is not surprising that the ERP vendors are quick to jump on the bandwagon, touting the benefits of an extended product’ [71].

However, software purchasers often do not understand their own requirements and may choose a package on the basis of a sales pitch [1,64]. Furthermore, as the software and the systems architecture will be new to those in an organisation, the business practices that the software supports may also be new and therefore it is suggested that organisations must engage in a process of organisational learning [72]. The idea that people in organisations may have to learn about the packages they have adopted is touched upon in the literature, interestingly though—this is usually during implementation. For example at Dow Corning, team members who requested modifications to the SAP package they were implementing were often unsure if the need for this had arisen from a deficiency in the package or their ability to learn how the package allowed a process to be performed [73]. Adam and O’Doherty [74] also report that in a few of their cases, the packages adopted led to the development of managerial expectations—they learnt what was possible and became more ambitious in terms of their objectives for the system. These issues do not vary widely from those encountered for some time in custom development. For example, Flynn [24] states that often, users may have a vague notion of requirements at the beginning of a project, there may be changes in external environments and initial requirements may have unfeasible implications that are not realised until implementation. Thus, a further reason why customisations might be implemented can be seen. When the selection process was being enacted, it was not clear what the basis for selection was therefore a product that does not fit with requirements might be chosen.

6.6. Shifting expectations of functionality fit

It is argued that a better fit between the packaged software functionality and user requirements leads to successful implementation and usage [75]. At a simple level, this seems logical. However, as illustrated in the last section, those in organisations are often uncertain about their precise requirements and therefore it may prove difficult to evaluate a package upon the basis of these ‘incomplete’ or ‘inadequate’ requirements. Moreover, even where requirements are fairly well articulated, evaluation can still be problematic as packages may well meet the requirements in an unfamiliar or unacceptable way [76]. It has been suggested for example, that most package purchasers find that at least 20% of their requirements are missing from the functionality of packages [77]. This is potentially no small mismatch either if the missing
20% is that which is perceived to be the most crucial to those in the consumer organisation. At the University of Nebraska, the average fit between the implemented SAP packaged was 60% and as low as 30% in some areas yet they bought the package with the expectation of it providing enterprise support [78].

Even though it has been argued that when purchasing a package, functional requirements must be determined and used to evaluate products, this may not happen as expectations are fuelled by a combination of selling acts and organisational learning. Moreover, as a result, expectations of package functionality may also shift when the realities of implementation and usage are experienced. Therefore, customisation might be necessary to accommodate unanticipated, shifting requirements that might not be readily met by the vendor.

6.7. Safeguarding

There are, at least, two strands to the idea that customisation might be undertaken as a form of safeguarding. Very early in the usage of packaged software, it was recognised that it could lead to the release of information systems personnel to work on other projects [79]. Additionally, a recent survey reported that 40% of respondents felt that packaged software would allow for reductions in the in-house development team [58]. This is argued to be the case as the consumer organisation will need to allocate fewer resources to development and maintenance activity because this is outsourced to the vendor [1]. Therefore, customisation might be performed as an act of resistance by ‘in-house’ or other relevant information systems personnel. Customisation work would justify their existence. More broadly, others might perform customisations in organisations as a form of protection. Hanseth and Braa [80] offer an excellent illustration of this in their account of the SAP project at Norsk Hydro. In this project local managers and many employees did not see the need for integration. Local users thus took a key role in the process of customising SAP for their individual sites. The result was a shift from a common system to a heterogeneous infrastructure. A further form of customisation, ‘external safeguarding’ might be concerned with the desire to maintain competitive advantages. As packages are generic products, with so called best practices inscribed within them, it might be perceived as preferable to customise the software so it is different from the mainstream [81,12]. The caveat on this point is to remember that competitive advantages do not only arise from those processes that are supported by technologies in organisations.

7. Conclusions

Despite the suggestion that ERP customisation will have catastrophic consequences, it is still undertaken in some form, with mixed results. There is little point in saying customisation is necessarily ‘good’ or ‘bad’, it is undertaken and it is important to understand why. The primary research reported here was predominantly focussed upon the fairly rational reasons for ERP package customisation. Moreover, the focus tends to be upon those reasons from the perspective of those in the implementing organisation. So, for example, customisations might be performed because functionality is missing, there is a desire to add value to the system, to make the system look more appealing to various users and to increase the efficiency of the operation of the package. However, what this research also begins to point to are other agendas for customisation that might not be expected given what is said of packaged software. Thus, the customisation of ERP packages might be seen by vendors, as fuelling the process of product development. Could therefore, some customisations can be viewed as unsuccessful product development initiatives? Shifting back to those in the organisation, it is also possible to see various relevant social groups. One could be the implementation team, using customisation to facilitate user acceptance or as a pluralistic control mechanism. Another could be management staff looking to maintain existing ‘value adding processes’ by working around inscribed ‘best practices’ or using customisations to increase system acceptability and efficiency. In the secondary research these themes are further amplified, especially the influence of diverse relevant social groups. In summary, customisations might occur as a form of maintenance because those in the consumer organisation see it as necessary or desirable due to the nature of the packaged software industry. Customisation might also be performed where it is not necessary - because of the nature of the industry and even specific product lines. Additionally, customisa-
tion might also happen because of the nature of the consumer organisational environment where there might be misconceptions surrounding the capabilities of a product or shifting expectations of a product over time. Finally, customisation might occur as a form of resistance or protection.

This study attempts to provide insights into why ERP packages might be customised. This work is useful for facilitating an understanding of the customisation of other forms of packaged software too. Given this, further work might empirically explore the reasoning, or lack of reasoning behind the decisions to customise software as alluded to in the secondary research conducted here. More broadly, there is also the possibility that the issues raised here, especially those related to the market oriented nature of systems development, have applicability to ‘traditional’ custom development projects. Custom development projects are usually conceptualised as in-house activities. However, with the commercialisation of in-house information systems functions and indeed, the use of external contractors for custom development, it is also worth considering future research that examines custom development from a market oriented perspective too.

Clearly, other reasons for customisation will exist, so the reasons reported here should not be taken as the definitive list. Indeed, identifying other reasons for customisation could be a further avenue of future research. For example, the primary research in this study is currently being reinterpreted to look at the role of customisations in aiding sociotechnical integration efforts. The aim here was merely to illustrate that there is more to customisation that the need to achieve function-fit.

Acknowledgement

The author would like to thank the researcher involved in the collection and interpretation of the primary data, but who wishes to remain anonymous.

References


