Abstract

This research concerns the configuration of packaged software. Specifically, it concentrates on the technical, organizational and social aspects of post-purchase configuration. Accordingly, adoption of a Social construction of technology (SCOT) approach to study configuration process of packaged software is proposed. In doing so configurable packaged software has been defined based on the previous literature. After a review of the socio-technical approach of SCOT, this theoretical lens has been defined and its use in the context of IS has been discussed. The potential insights from such a study are argued to benefit both use and development of packaged software.

1. Introduction

Although the origin of packaged software dates back to the late 1960s, the first serious Information Systems (IS) studies began in late 1980s when Lucas [28] wrote his seminal paper on this topic. Packaged software resulted from a 1968 initiative of the United States Department of Justice that required IBM to unbundle software from hardware [9, 20, 37]. This has changed the nature of software acquisition by organizations dramatically over the past few decades. Today, organizations do not necessarily need to have highly skilled software developers in order to benefit from the latest software developments. Ability to purchase packaged software is believed to save organizations time, money and other resources [22].

Despite the increasing relevance of packaged software, Information Systems (IS) researchers have concentrated mostly on custom developed software [26]. Research on custom development is important because many large (especially governmental) organizations, which need specialized software, still custom develop their software applications. However, a large number of organizations do not address their software needs through custom development [26, 48].

Moreover, the topic of packaged software deserves to be investigated separately due to its uniqueness in terms of development, and use [26]. From the development point of view issues such as not involving the users, building workable software for different organizational contexts, and offering products through third-party organizations to end-users make packaged software significantly different from custom-built software [37]. On the other hand, when it comes to using packaged software, organizations need to either adapt their organizational processes to the packaged software or configure the software to fit organizational processes. This is not necessarily the case in use of custom developed software because this kind of application is developed based on the organizational processes [48]. The only form of packaged software that has received generous attention from the IS research community is Enterprise Resource Planning (ERP) systems. Since the 1990s, there has been a tendency in the IS research community to exemplify packaged software by ERP applications [26]. While the findings of this research stream are valuable and insightful, understanding more ‘generic’ aspects of packaged software and challenges that organizations face when adopting third party software have been under investigated. Therefore, IS literature lacks a clear understanding of the scale and scope of packaged software (Ibid).

Recently, some researchers in the IS community have begun to bring the importance and relevance of studying packaged software issues to the forefront (e.g. [13, 26]). Nevertheless, the importance of packaged software in IS research appears to ebb and flow. Lucas et al. [28] strongly advocated the
significance of this topic. In 1997, Carmel’s article [9] examined the factors that would sustain the hegemony of the U.S. in packaged software market. In 2007, a special issue of EJIS examined the topic of packaged software. Light and Sawyer [26], in the commentary that accompanied this issue, called for attention to theorizing about packaged software and its location in IS research. Other researchers have also shown interest in this topic around each of these timelines (e.g. [1, 21, 22, 13]). However, the body of literature on software packages remains scarce and fragmented.

Packaged software has received more attention from computer science and engineering communities, both of which favor technical over social aspects. However, even in these fields, for most part, the topics of selection and evaluation of software have been the focus [29, 41]. One assumption of research centered on selection and evaluation is that once an organization selects the most fitting software for their organizational needs, they will be able to reap the benefits that the software is designed to offer [30]. What these studies neglect to mention is that even though pre-purchase evaluations of different software options is useful for finding the best fit, the actual fit or misfit of the software would not be realized until it is being configured [48]. This is due to the fact that organizational practices are context dependent (Ibid). Even if an organization considers that they have found the best fitting software, due to organizational and social issues, the benefits of the software have the potential of not being fully (or at all) realized (Ibid). Therefore, discussion of the issues around post-purchase configuration of packaged software is rare even though extremely critical.

The goal of this study is to understand the how and why of the process of configuration of packaged software and to aim for theoretical generalizations about this process. Therefore, we propose an interpretive study that investigates this process through the lens of Social Construction of Technology (SCOT). Such study would investigate the process of packaged software configuration, in which several groups of people are involved. Organizational studies have shown that different individuals/groups interpret and understand a situation differently and they act according to their understanding. Previous research for most part has focused on investigating perceptions of different parties (mostly individuals) about information systems based on some pre-defined attributes of the technology (e.g. ease of use) [14]. Some studies treat understanding and perceptions of individuals/groups towards a technology as contextual and situation-specific. In these studies also there has been a tendency to investigate the contextual differences of the individuals or groups based on some predefined domains (e.g. technology-in-use) [34]. In these studies “stakeholders” are classified based on their functional roles (e.g. users, managers, etc.) in the organization. SCOT offers an alternative approach, the focus of which is interpretation. Accordingly, interpretations and not the functional roles define the classification of various groups. The approaches of various groups towards a technology based on their interpretation and understanding of the technology is then investigated. This approach to the configuration process can shed some light on our understanding of the technical, social, and organizational dynamics of this process.

In the following section we clarify the definition of configurable packaged software that this study concerns. In the sections that follow, we propose the SCOT approach to studying configuration process of packaged software and discuss potential insights from such a study.

2. Literature review

A review of prior literature reveals that there is no common understanding as to what constitutes packaged software. Several attributes are commonly assigned to this type of software by different researchers. These attributes include the software being already-built [33, 43, 44] without any control from the customers over its development [2, 43, 44] distributed in large number of copies, licensed, leased, or bought [33, 8].

Some of the less commonly mentioned attributes of packaged software in the literature include the software being offered by third-party vendors, which retain the intellectual property rights over the software [8], the software needing minimal change and no source code modifications, and customers not having access to the source code [43, 44].

In this study we focus on the type of packaged software that is purpose-specific. In classifying different forms of packaged software Morisio and Torchiano [31] consider the attribute of role to distinguish various packaged software based on their intrinsic function. Role can have two dimensions: horizontal and vertical. A broad vertical role denotes that the software covers a large number of functional areas involved in the same business process. A broad horizontal role entails that the software covers large number of functions. For example, ERP systems encompass broad horizontal roles. Therefore, horizontal role is what distinguishes ERP systems
from the packaged software with smaller scope that is of the focus in this study.

It is argued that almost all software applications require some degree of configuration before they become workable [42]. Configuration can vary from simply setting up the databases and the parameters to modifying organizational business processes [28, 41, 44].

Based on the review of literature, the type of packaged software that is of concern in this study is

- **a standalone, purpose-specific application software with narrow horizontal role.** This type of software is already-built with little or no involvement of its final users. This software can be sold / licensed / leased / borrowed / or acquired free of charge in large number of copies. This type of software requires configuration in order to become workable. Configuration process involves various relevant groups in the organization. This process can vary from simply setting up the parameters to tailoring the software without modification of the source code.

Configuration of packaged software is seldom a simple ‘plug-and-play’ process. This process can become extremely complex and time-consuming [16]. ‘Plain vanilla’ adoption is usually not feasible [e.g. 24, 25, 45, 48] and the process of forming a workable software for a specific organizational context can turn into a challenging process.

Accordingly most organizations need to either adapt their processes to the software or modify the software to fit their practices [40, 38, 45]. Therefore, organizations usually assign groups from different relevant functional areas to configure and implement the software to fit the organizational context. Each of these groups could have different perceptions and understanding of the software and its capabilities. In other words, the software and practices it offers could be interpreted in different ways. These interpretations are key in shaping the direction of the process of configuration: adapting the organizational process to the software, or configuring the software to fit the processes.

The widely known term “configuration” regarding packaged software is related to configuration management (CM). CM is a technical term which refers to a discipline for version controlling software systems [10]. Configuration as a process that involves not only technical but also social and organizational adaptations has been mentioned but not extensively researched in IS or computer engineering fields. Sommerville’s work [42] is one of the rare works that we were able to identify which focuses on the definition of configuration of packaged software as a technical as well as social, and organizational process.

Sommerville [42] identifies three main problems attributed to configuration of packaged software: Understanding the configuration options; Understanding the configuration semantics; Understanding how a system is configured. He argues that “most configurable systems offer a range of different configuration options with, sometimes, subtle and difficult to understand interactions between these options. Sometimes, these options are obscure and poorly documented and there is rarely information available about how different options may interact” [42, p. 7]. Developers thus usually infer these options based on their limited understanding.

Even though Sommerville repeatedly emphasizes the importance of social and organizational challenges of configuration process, his study does not tap into these issues deeply. He mainly focuses on developers (i.e. mostly programmers) in the organizations. In practice, however, various relevant groups from different functional backgrounds are involved in the process, which can add to the challenge. The purpose of [42] is to bring the importance and criticality of the challenges of configuration to the fore. This study calls for more attention from the research community to the topic of configuration of packaged software.

Delving into the dynamics of post-purchase activities of packaged software, therefore, would be significant and timely. In order to gain an understanding of these dynamics, examining interactions of different groups who are involved in the process would be insightful. Studying how these interactions influence the way the software is configured and how in turn the options and limitations offered by the software influence the interactions will add to our understanding of software implementation and configuration.

An in-depth interpretive study would be valuable in understanding the processes of software configuration which involves diverse groups [46, 49]. Social construction of technology (SCOT) lends an appropriate approach to investigating social and organizational aspects of configuration process. In next section, a discussion of SCOT as a theoretical lens that can guide such a study is provided.

### 3. Theoretical lens
Social interaction is at the core of the phenomenon under investigation in this study (i.e. configuration of packaged software). Therefore, choosing a theoretical lens, which is based on social interactions to study this phenomenon is warranted. The relevance and importance of SCOT approach in studying development and implementation of information systems has been repeatedly acknowledged in IS research (e.g. [34, 27, 12]).

SCOT is a theory of socio-technical change [32], which takes an anthropological approach to studying technological developments [5]. Development of SCOT was as a result of dissatisfaction with linear models of technological development. According to linear models, technological developments follow rational, orderly, and pre-specified stages (e.g. research, development, pilot production, and implementation) [17]. To address this issue Bijker and colleagues, adopted a similar approach as Empirical Programme of Relativism (EPOR) to study technological developments. EPOR is a social constructivist approach to development of knowledge in natural sciences [35].

SCOT’s emphasis on thick description [18] of the technological artifact and its surrounding context makes it a well-rounded theoretical lens. One of the main premises of SCOT is that technology is interpreted differently by various social groups that are relevant to the technology. Each group constitutes of individuals who are, in a way, concerned with the technology. Members of each group share the same meanings about the technological artifact, which is different from the meaning(s) other groups assign to it [4, 5].

Accordingly, the classification of different relevant social groups is based on their interpretations of the technology. In other words, individuals who interpret the technology similarly and thus approach its problems in a similar fashion are considered to be members of the same relevant group [3, 5]. This is dramatically different than the traditional way of classifying groups based on their functional areas (e.g. users, developers, and managers) that is prevalent in IS literature. An advantage of this new way of classification is that people are grouped based on their points of view rather than being attributed with some points of view based on their job descriptions. Due to different interpretations assigned to a technological artifact, the technology is said to have interpretive flexibility (Ibid). This means that different groups of people approach the technology differently. These groups of people contribute to gradual shaping of the technology until it takes a workable form. In the SCOT approach the focus is on controversies and their resolutions. By identifying the problems that various relevant groups have with technology, the researcher more closely examines how those controversies are approached, resolved and eventually the technology is interpreted as workable by various relevant groups [3, 4, 5].

The way a group approaches the technology and its related issues is called the technological frame of reference (TFR) of the group. Technological frame of a group guides interactions of the groups with one another and with the technology. Technological frames of reference are built when interactions around a technology begin. Different groups contribute to gradual shaping of the technology until it reaches stabilization and takes a working form. At this point the technology is said to have reached closure. There are different ways to reach closure. For example, in some organizations power could be used as a mechanism to reach closure and to resolve controversies around the technology [3, 4, 5]. Closure does not mean that the technological artifact could not change in form further. What it implies is that at the time of closure the controversies around the technology are resolved and its interpretive flexibility has diminished.

In the past, Socio-cognitive approaches have been adopted to study development of information systems based on interpretations and expectations about the technology (e.g. [34, 11, 12]). These studies are an extension of cognitive studies, which investigate how individuals make sense of technologies based on their frames of reference (e.g. [6, 7]).

However, social interaction is not the focus in socio-cognitive approaches. In some studies, ideas from social theories such as SCOT [3] and structuration [19] have been borrowed to consider a social context for technological frames of reference [34]. Essentially though, in these studies technological frame is an individual-level socio-cognitive structure which might be shared with other individuals. In SCOT, on the other hand, technological frame is defined as a social concept that structures the interactions within and between the groups [5].

Even though the relevance and importance of SCOT in IS research has been acknowledged, it has not been extensively applied in this field. Application of the elements of SCOT seems to be partial or merely at a definition level rather than the theoretical view being applied as a whole [23]. In addition, in socio-cognitive studies, IT artifact is considered as a black-box. SCOT, on the other hand, examines how the artifact is shaped based on the approaches taken by various groups which are in turn influenced by various interpretations of the artifact. Therefore,
SCOT offers a holistic approach, which looks into technical as well as social and organizational dynamics of the process of technological development.

Criticisms of SCOT have been mostly directed to the influence of broader societal structural relationships (e.g. politics, gender) on technological developments [47, 15]. SCOT has also been criticized for its relativism [47, 36]. It has been argued that relativist and subjective approach of SCOT might lead to an oversimplified view of technology and the society.

SCOT does not deny the influence of structural issues such as politics. These issues could be taken into account when identifying social relevant groups and their interpretations of the technology. What needs to be taken into consideration is that SCOT is meant to be a theoretical lens through which technological developments can be studied- studying technological development through interpretations and meaning.

All in all, as the developers of this approach have acknowledged, SCOT is a not a deterministic theory of technological development. SCOT is a theory to be used as a sensitizing lens to study development of a technological artifact paying attention to both its social and technological aspects. These attributes of SCOT make it an appropriate theory for the purpose of this study.

A discussion of the potential insights that could be gained from a SCOT approach to studying configuration process of packaged software is provided in the following section.

4. Configuration process through the lens of SCOT

SCOT is highly relevant in the context of configuration of packaged software. Various relevant groups in an organizational are normally involved in the process of configuration. Perceptions and expectations of these different groups of the packaged software and its functionalities can be critical. Various understandings of the software by the relevant groups could be reflected in the suggestions they provide for how to configure the software to a workable form. The decisions that are made during the process can be highly influenced by different interpretations. Rich functionalities of a software can make understanding of what it can offer and how it can be configured challenging. This challenge could be amplified if different relevant groups understand the software differently.

Therefore, the nature of the process of configuration of packaged software is in line with the SCOT approach. Accordingly, based on SCOT different people involved in this process can be grouped based on the shared meaning they assign to the software. This kind of approach to the process of configuration could allow us to examine different perceptions about the software, its functionalities, and configuration options.

Taking a SCOT approach, a researcher would identify problems of various relevant groups regarding the software. Identification of these problems would allow the researcher to closely examine how they are approached and addressed. The researcher would be able to study the process of configuration as a series of technical, social and organizational activities.

Furthermore, the emphasis of SCOT on the technological artifact (the packaged software in this case) and providing a thick description of its evolvement and the social and organizational issues around it make SCOT an insightful lens. Figure 1, schematically demonstrates the relationship of different elements of SCOT in the context of configuration process of packaged software.

As a preliminary step to substantiate the application of SCOT in the context of studying packaged software, we searched for studies with similar objectives. Our goal was to look at the studies that took different approaches than ours. We then used SCOT approach to investigate how SCOT would have explained the findings of (or contradictions found in) the studies. Yeow and Sia [48], for example, examined the incongruence of socio-cognitive [34] technological frames of different groups in the process of implementation of a packaged software. The goal of the study was to investigate how the ‘best practices’ of the packaged software were negotiated by different groups and how these groups reached agreements.

The researchers classified the participants of the project based on their functional areas (e.g. end-users, Financial policy department (FPD)). Their findings from a pair-wise comparison of various technological frames demonstrated that several frames overlapped on some elements. For example, they found that both end-users and FPD groups’ frames emphasized on audit and control issues. This finding is similar to the premise of SCOT that various groups take different approaches towards the technology. However, based on a SCOT approach the classification of the participants of the project would have occurred after identifying their perceptions about the software and its issues. Examining Yeow and Sia’s [48] work assured our study in two ways.
First, it showed that implementation and configuration of packaged software can become complex, studying of which is warranted. Second, SCOT approach demonstrated to be relevant to such studies. Therefore, the insights gained from an SCOT approach to the process of configuration of packaged software would allow us to further understand this process. These insights could also be incorporated by packaged software developers in order to build software that is more understandable by end-users. This contribution is particularly important since the nature of packaged software development does not allow for direct involvement of end-users. As such, the implications of such a study are discussed in the next section.

![Figure 1. SCOT approach to configuration of packaged software](image)

5. Future research

The goal of this study is to introduce the socio-technical SCOT approach and discuss its potential application to studying a phenomenon that has been under-researched in IS literature – namely packaged software. To further support our proposed conceptual approach to studying packaged software configuration process, in-depth interpretive studies would seem appropriate. The empirical part of the study presented in this article consists of a work-in-progress case study. The data collection for this case study began in Spring 2007 by contacting a public organization that is in the process of implementation of a work management system. For the past twelve months, one of the researchers has been participating as a neutral observer in twenty six sessions conducted for configuration of the software. These sessions are run by employees from different functional areas that are involved in the configuration process. The researcher has also obtained and partially reviewed documentation material related to the project. The next step is to conduct in-depth interviews with relevant individuals and/or groups.

The process of data collection has been guided by the theoretical lens of SCOT. This lens will also be used in analyzing the data from different sources using a hermeneutic approach. Using various data sources - observation, interviews, and documentation - will enrich our understanding, and help to validate the findings. The observations so far have shown that indeed interpretations of the software are diverse and the process of configuration is approached differently based on these diverse interpretations. However, this is a research in progress and no insights are proposed until the data collection and analysis are completed.

The findings of the proposed study will contribute to both theory and practice. This study will be a contribution towards building a cumulative body of theoretical knowledge on the concept of packaged
software. Another contribution will be the application of an insightful theoretical lens such as SCOT in the context of IS studies. Practitioners interested either in acquisition or development side of packaged software products would also benefit from potential contributions of this research.

On the acquisition side, the insights gained will be beneficial in familiarizing the organizations with the dynamics and issues related to implementation and configuration of packaged software. This study will also benefit the IS development side because it will help software development companies familiarize with the type of challenges their customers face when implementing their products. These understandings then could be incorporated in the development process by software companies. Consulting companies would also gain from such a study. The insights gained could be used to tailor the services (e.g., training and documentation) they offer to their customers for implementation of packaged software.

6. References


di


