Intensifying to cease

Mehrizi, Mohammad Hosein Rezazade; Modol, Joan Rodon; Nezhad, Milad Zafar

published in
MIS Quarterly: Management Information Systems
2019

DOI (link to publisher)
10.25300/MISQ/2019/13717

document version
Publisher's PDF, also known as Version of record

MIS Quarterly

RESEARCH ARTICLE

INTENSIFYING TO CEASE: UNPACKING THE PROCESS OF INFORMATION SYSTEMS DISCONTINUANCE[1]

Mohammad Hosein Rezazade Mehrizi
School of Business and Economics, Vrije University Amsterdam, De Boelelaan 1105,
1081 HV Amsterdam, NETHERLANDS {m.rezazademehrizi@vu.nl}

Joan Rodon Modol
Universitat Ramon Llull, ESADE Business School,
08172 Sant Cugat del Vallès, SPAIN {joan.rodon@esade.edu}

Milad Zafar Nezhad
Industrial and Systems Engineering, Wayne State University,
Detroit, MI 48202 U.S.A. {m.zafarnezhad@wayne.edu}

Legacy information systems consume a large portion of information technology budgets and often impose serious limitations on organizations' flexibility and innovation. Despite the extensive literature on how organizations adopt and use new IS, we know little about how organizations discontinue their ***legacy*** *IS. Current studies suggest some actions and events to cease mechanisms such as legitimization, learning, and routinization that give continuity to the systems. However, we do not know* ***when*** *these actions emerge in the discontinuance process to gradually reduce the organizational commitments to legacy IS, nor do we know how ceasing one mechanism can facilitate or, conversely, hamper ceasing other mechanisms, especially when these systems involve interdependences between various components. Based on a process analysis of four software companies, we show that, contrary to the current literature, IS discontinuance is not a matter of merely ceasing each and every mechanism of an established IS; rather, it often requires that some mechanisms be temporarily* ***intensified*** *to prevent premature discontinuance and enable subsequent cessation of other mechanisms. Through cross-case analysis, we articulate a set of causal mechanisms that expands our understanding of how the discontinuance process can be differently shaped by organizational commitments and interdependences involved in legacy IS.*

Keywords: Information systems discontinuance, self-reinforcing mechanisms, organizational path, obsolescence, process model

Introduction

> I wish we could start from zero. Our emergent competitors don't have such a huge dependency as we have. They just start by choosing the right technology, but we have to also fight with the monster we have created over the years.

This statement by the CEO of a software company inspired us to ask: *How do organizations discontinue their legacy IS?* IS discontinuance refers to "the cessation of the use of an organizational information system" (Furneaux and Wade 2011, p. 574) that no longer contributes to organizational aims.

[1]Youngjin Yoo was the accepting senior editor for this paper. Chee-Wee Tan served as the associate editor.

The appendices for this paper are located in the "Online Supplements" section of *MIS Quarterly*'s website (https://misq.org).

IS discontinuance is vital when legacy systems "constrain rather than support the ability of the organization to respond to changing environmental conditions" (Alderson and Shah 1999, p. 115) and consume a major part of IT budgets (Slee and Slovin 1997). For instance, a survey of U.S. federal agencies shows that 70% of the $62 billion annual IT budget is devoted to maintaining legacy systems (MeriTalk 2013); yet, about 75% of this cost does not provide any competitive value for organizations (Hainaut et al. 2008). IS discontinuance is particularly challenging in established industries such as financial services where legacy IS comprises multiple interdependent technological objects, organizational routines, and actors to which organizations have developed historical commitments (Polites and Karahanna 2012).

While the adoption and use of new systems have been at the heart of IS literature, IS discontinuance has remained underresearched (Furneaux and Wade 2011)[2]. Research on IS discontinuance focuses on factors, such as dissatisfaction with the legacy IS, that motivate users (Parthasarathy and Bhattacherjee 1998) and managers (Furneaux and Wade 2011) to stop using it. Other studies suggest that to discontinue a system, organizations need to cease the mechanisms that give continuity to it (e.g., via disrupting legacy IS routines) (Polites and Karahanna 2013) and delegitimizing current IS choices (Sahay et al. 2010).

These views provide only a limited explanation of how organizations discontinue their legacy systems. First, prior studies do not examine the *process* through which organizations discontinue their systems. Our position is that we should not see IS discontinuance as an event, but as an iterative, emergent process that consists of multiple phases and interactions. Moreover, it is also important to examine the temporality of actions that gradually reduce organizations' commitments to legacy IS. Sometimes ceasing a mechanism too early can harm the entire IS discontinuance if organizations fail to mindfully manage their legacy commitments.

Second, prior studies examine discontinuance at the level of individuals' intentions but not as an *organizational* phenomenon. In particular, prior studies do not examine how actions taken by various organizational actors interact over time and gradually cease the use of a system. We consider discontinuance as a complex process that involves multiple *interdependences* across the legacy technological elements, organizational routines, heterogeneous actors, and various products and services. We argue that we must consider the mechanisms that give continuity to existing IS not as isolated from one another, but rather as interdependent. Therefore, we need to understand how the discontinuance of one aspect of a legacy IS (e.g., disrupting a specific IS routine) may facilitate or hamper the discontinuance of other aspects.

Drawing upon organizational path literature (Sydow et al. 2009; Vergne and Durand 2011), we frame IS discontinuance as a process that involves ceasing a set of interdependent self-reinforcing mechanisms (SRMs). A SRM refers to a mechanism such as legitimization, learning, and routinization, by which each subsequent turn intensifies the previous turn (Schreyögg and Sydow 2011). Accordingly, we seek to answer the question: *How do the actions and events that cease the self-reinforcing mechanisms of a legacy IS interact over time, leading to its discontinuance?*

We study the discontinuance of established systems that underlie the products and services of four software companies. In so doing, we make three contributions. First, we show that IS discontinuance is not a set of independent attempts to cease the various SRMs, but rather an *iterative process* through which these attempts interact in a temporally complex way. In particular, we articulate and theorize four phases, which constitute the discontinuance process: (1) realization, (2) reversion, (3) handover, and (4) marginalization.

Second, the phases help us show that the IS discontinuance process does not merely require ceasing underlying SRMs, which is often proposed by the current theories (Stache and Sydow 2014; Sydow et al. 2009). Rather, paradoxically, it also involves phases through which some SRMs are *temporarily intensified* to enable the subsequent cessation of other SRMs.

Third, through cross-case insights, we show the process is contingent on the historical commitments and the interdependences between the various components of the legacy systems. We propose theoretical explanations that extend current literature, especially when legacy IS involves extensive commitments and/or interdependences.

In the next section, we conceptualize the IS discontinuance process through organizational path theory. We then report our empirical inquiry of how IS discontinuance unfolded differently in the four cases. Finally, we discuss the theoretical and practical implications of our study.

[2]Technological discontinuance and related concepts, such as decommissioning, sun-setting, retirement, and phasing-out, have been widely addressed in the engineering field (Bartels et al. 2012). These studies usually propose prescriptive models or depict best practices to guide practitioners in discontinuing technological products. This literature usually treats discontinuance as part of the process of managing technological product obsolescence.

Theorizing IS Discontinuance as a Path-Breaking Process

We frame a legacy IS as a set of interdependent technologies, actors, and routines that are developed and aligned through historical decisions and actions (Lyytinen and Newman 2008). Therefore, we need a theoretical perspective that allows us to put at the forefront of our analysis the commitments and interdependences that characterize the legacy IS. We theorize the discontinuance process by drawing upon organizational path literature (Garud et al. 2010; Singh et al. 2015; Sydow et al. 2009; Sydow et al. 2012), which explains how a certain pattern of organizational choices and actions (that constitutes a path) emerges, sustains, and is broken. To explain the emergence and dominance of a path, the theory offers the concept of *self-reinforcing mechanisms* (SRMs): the forces that reproduce a particular pattern of choices and actions over time, and hence sustain it (Schreyögg and Sydow 2011).[3]

Once a specific IS path has emerged and is consolidated, organizations might face strong inertia (Polites and Karahanna 2012). As a result, when organizations are trapped in the existing IS path and at the same time must embark on an alternate IS path (e.g., implement a new technology), they need to break into the current IS path. According to organizational path theory, organizations can break away from the established path by "interrupting the logic and the specific energy of the self reinforcing mechanisms" (Sydow et al. 2009, p. 702). Although organizational actors are influenced by historical choices and actions, they at the same time can "deviate from path-dependent behaviors" (Lyytinen and Newman 2008, p. 606).

Five streams of research provide insights into how an established organizational path can be broken by ceasing a specific SRM (see Table 1).[4] A first stream draws upon human behavior to consider the *indwelling* SRM (Polanyi 1966): The more actors interact with an IS, the less they pay attention to its details,thereby gradually pushing the details of tasks and objects to the background. Users can economize on their limited cognitive and attentional capabilities, particularly in dealing with complex information systems. Certain breakdowns in the flows of actions (Bhattacherjee 2001) can cease indwelling and bring the details of the system back to the center of actors' attention (Simon 1977).

A second stream of research considers the *legitimization* SRM: The more widely an IS is accepted as an appropriate system, the more widely it will be accepted and used by other organizational actors (Sydow et al. 2009). Delegitimization (Oliver 1992; Scott 2001) can be triggered by functional failures and political dissension (Nicholson and Sahay 2009), leading users to stop seeing the legacy IS as adequate. Delegitimization can also be triggered by implementing a new IS whose logic conflicts with that of the legacy system. For instance, the logic of integrated, standard automation that underlies ERP systems can delegitimize the fragmented, flexible automation logic behind traditional office automation systems (Wagner et al. 2011). Organizational actors can actively delegitimize an organizational path (Garud et al. 2011), for example by "avoiding institutional monitoring and sanctioning," "not selecting institutional practices/selecting others," "attacking the legitimacy or taken-for-grantedness of an institution," and "undermining institutional mechanisms" (Battilana and D'Aunno 2009, p. 48).

A third research stream focuses on the *learning* SRM: The more organizational actors know about the legacy IS, the more effectively and efficiently they will use it (Robey et al. 2002). Studies suggest that criticizing beliefs and assumptions regarding the established technologies and routines (Wang et al. 2008) can disrupt the established mindset and trigger deeper learning (Argyris 1977) and unlearning (Hedberg 1981). Hence, mechanisms such as frame-breaking and sense-breaking (Aula and Mantere 2013; Lawrence and Maitlis 2005) are important to shake the taken-for-granted mindsets about the IS practices (Vlaar et al. 2008).

A fourth research stream focuses on the *resource complementarity* SRM: The more resources are allocated to and aligned with a certain IS, the more value organizations can generate from this complementarity (Keil 1995). For instance, the literature on IS project de-escalation (Flynn et al. 2009; Montealegre and Keil 2000) suggests that the forces for continuing an IS project can override the forces for discontinuing it because of the expectation that further resource allocation can turn the project around. Therefore, reducing the investment of economic resources (Mähring et al. 2008) may help organizations eliminate the resource complementarity mechanism.

A fifth research stream focuses on the *routinization* SRM: The more widely an IS is used, the lower the coordination cost, therefore, the greater the tendency to adhere to the same

[3]Economists formalize SRM through "increasing return" on repeated courses of actions (Arthur 1989). In the language of system theory (Maruyama 1963), SRMs are known as "positive loops."

[4]We reviewed the literature iteratively through concepts that implied how various SRMs that underlie an organizational (IS) path can be discontinued: deinstitutionalization, disrupting habits and routines, de-escalation of commitments, unlearning, etc.

Table 1. Summary of Literature Regarding the Breaking of Organizational (IS) Path

Self-Reinforcing Mechanism	Ceasing Mechanisms
Indwelling (behavioral): The more actors work with an IS, the less they pay attention to its details, and become more easily engaged with it.	• Breakdown in the flow of actions (Butler and Gray 2006; Polanyi 1966)
Legitimization (political): The more widely an IS is accepted as an appropriate system, the more widely will it be accepted and followed by other organizational actors.	• Delegitimization and Deinstitutionalization (Berente and Yoo 2012; Nicholson and Sahay 2009; Oliver 1992; Sahay et al. 2010; Scott 2001)
Learning (cognitive): The more organizational actors know and learn about the legacy IS, the more effectively and efficiently will they use it.	• Deep, critical reflection (Argyris 1977) to question and criticize established IS beliefs (Wang et al. 2008)
Resource complementarity (material): The more resources are allocated to and aligned with an IS, the more value organizations can get from it.	• De-escalation of commitment to IS projects (Flynn et al. 2009; Mähring et al. 2008; Montealegre and Keil 2000)
Routinizing (economic): The more widely an IS is used, the lower the cost of coordination; therefore, the greater the tendency to adhere to the same way of using it.	• (Intending to) stop the usage of an IS (Brook et al. 2015; Whittle et al. 2011) • Disrupting the flow of IS routines and habits (Polites and Karahanna 2013)

way of using it (Polites and Karahanna 2013). While indwelling concerns gradually not paying attention to the known details, routinization concerns the gradual process of *acting* according to the way things have been done before. In this way, some studies examine how the day-to-day routines and habits associated with an established IS can be disrupted, for instance, via interference and distraction (Polites and Karahanna 2013). In particular, the embedded actors can intentionally decide not to continue a routinized course of action (Brook et al. 2015; Whittle et al. 2016).

To summarize, organizational path theory provides a powerful conceptual framework for studying IS discontinuance. First, the concept of organizational path allows us to understand legacy IS as an established pattern of choices and actions that tends to continue due to historical commitments. Second, the concept of SRMs conceptualizes the driving forces of an established path as ongoing cycles of actions and choices (Garud et al. 2010). Therefore, we can explain the discontinuance process through the actions that cease different SRMs. Third, organizational path theory provides a *process* ontology that enables us to examine the temporality of path-breaking actions and interactions. Fourth, organizational path theory allows us to *integrate* the various theoretical explanations regarding how various SRMs can be ceased. As shown above, each of the five streams of research focuses on a separate SRM—legitimization, learning, complementarity, indwelling, and routinization—thereby assuming that ceasing one and each SRMs can be independently pursued. Yet, this assumption is not valid when we note the interdependent nature of legacy IS.

Research Method

Empirical Settings

To examine the discontinuance process, we followed a multiple case study design (Yin 2002). We focused on a major replacement of the operating system and the products running on top of it in four software vendor firms that develop and sell office automation products. This context offers a rich setting for studying IS discontinuance because the old systems had to be removed rather than partially modified. Various socio-technical elements being replaced were at the core of the legacy IS: production technologies (e.g., testing platforms and the designs, codes, and modules for developing their products), production and support routines (e.g., debugging procedures), employees' skills (e.g., designers), and associated products and services.

We focused on the discontinuance process inside the four vendor firms. For confidentiality, we call the companies SmallCo (small), MedCo (medium-sized), LargLenient (large), and LargStrict (large). See Table 2 for a summary of cases. We considered their clients as part of the discontinuance context. Examining IS discontinuance in four cases that are all exposed to similar economic, industrial, and technological environments allowed us to more accurately examine the effect of the legacy IS conditions in shaping the discontinuance process (Tsoukas 2009). The cases enabled us to examine how the process differently unfolded depending on the commitments that organizations have to and the interdependences involved in the legacy IS.

Table 2. Description of Empirical Cases

Case Description	Legacy IS Conditions	
	Commitments	**Interdependences**
SmallCo: Private, small (40).* founded in 1991 • Focused on document management application, as the primary product, to a wide range of private and public organizations • Mainly young university graduates (average age of 27) • Friendly, informal atmosphere, with open culture to new technological changes • Flat structure around production, test & support, sales & marketing, finance & administration divisions	*Limited*: A small team of experts to serve around 10 clients	*Limited*: A single, separate document management product
MedCo: Private, medium-sized (100),* founded in 1988 • Focused strategy (after 2000) on document management application, mainly to large, public clients • A mixture of young (70%) and seasoned experts (30%) • Rather informal climate based on interpersonal trust and shared values among managers and employees, with centralized decision making among top managers and key experts • Strategy of avoiding hasty technological changes to provide stable and reliable service to the clients • Structured around production, test & support, sales & marketing, finance & administration divisions	*Moderate to extensive*: Around 100 clients and the strategy to admit their requests; internal specialization in legacy domains	*Limited*: A single, separate document management product
LargStrict: Private, large (900),* founded in 1988 • Diversified enterprise applications and gradually customized into industry-specific solutions (more than 45 applications) for all types of clients (small and large, public and private) • Specialized structure around technical departments • Internalizing the design and production activities and externalizing support activities to around 20 spin-offs • Structure around product management, internal systems, sales and marketing, training, consultancy, client support divisions • Pursuing a leader position in terms of market share	*Extensive, avoidable*: around 1,000 clients, strong bargaining power to force terminating and externalizing legacy IS support	*Extensive*: 10 core and 30 peripheral specialized products highly interdependent technological components
LargLenient: Private, large (450),* founded in 1987 by a group of 12 experts in developing manufacturing information systems • Diversified products (18 enterprise applications), transformed into customizable integrated solution (ERP) for various industries • Specialized structure around R&D, production, quality assurance, client support, marketing and sales, finance, and branches • Pursuing technological leadership with extensive R&D projects	*Extensive, unavoidable*: 900 clients and the admitting clients requests	*Extensive*: interdependent technological components through 10 core and 100 customized products

*Number of full-time employees in 2009.

Organizational commitment refers to the amount of specialized resources (e.g., personnel, products, services, financial resources) that companies devoted to support the legacy systems and products for their clients. In fact, the discontinuance process was completed often when the last client had migrated into the new system and/or the software vendor provided no more support for the legacy products (see Table 2 for the organizational commitments in each case).

Technological interdependence is present when changes in one component in the legacy system (e.g., data field of a database) require changes in some other components (e.g., a code module). In our context, the degree of interdependence increased when companies were developing multiple products that were interdependent in terms of supporting each other's data and functions. Both LargLenient and LargStrict had developed several products that were designed to work as an integrated solution, thus having extensive interdependences between codes, databases, functions, and configurations. These interdependences are exacerbated when they have been embedded into the products that are sold to numerous clients (see Table 2).

Data Collection

The fieldwork took 18 months, from March 2008 to September 2009. We extracted the story of the replacements, with focus on the legacy systems. First, we captured the background of the companies and the replacements through 16 exploratory interviews (on average, 40 minutes per interview) with former managers, current employees, clients, and industrial association heads. We also examined public documents regarding the companies' backgrounds, products and services, and relations with clients. Having all companies in the same market and facing similar technological replacements facilitated exploring their business and technological background.

Second, we zoomed in on capturing the story of the replacement in each company. We ran 33 interviews (average 60 minutes per interview) with managers and technical experts in the four companies to extract the chronology of events, decisions, actions, and changes related to the established systems. All interviews (except two exploratory ones) were voice recorded and transcribed. We used a semi-structured interview protocol, consisting of three parts: (1) understanding the case background, (2) describing the replacement process, and (3) eliciting the discontinuance story by focusing on the technological components, products, routines, and roles. We asked about the actions they took concerning the discontinuance of the established systems as well as their purposes and consequences. We covered informants with positive, negative, and neutral positions toward the replacements.

For each company, we also examined various documents to understand the company's history, its products and technologies, internal and external relations, and its plans and change decisions. These documents enabled us to complement and validate the interviews (see Table 3).

Data Analysis Process

We analyzed data through four steps (see Table 4). First, we constructed the story of IS discontinuance in each case to map *who* (e.g., programmers, HRM manager, or marketing manager) *did what* (e.g., calling a meeting to discuss the shortcomings of the MS-DOS technology), *when* (e.g., after receiving a client request for a major feature), and for what *purposes* (e.g., to limit learning new techniques related to the old technology), leading to what *consequences* regarding the progression of IS discontinuance (e.g., triggered a new round of efforts to transfer legacy data). In developing the stories, we focused on the legacy IS and the elements such as the products and services, relevant actors, technological objects, and the organizational routines and procedures shaped around these old systems. Each story provided an integrated understanding of events and actions that constituted IS discontinuance process and the external factors such as clients' requests (Langley 1999).

Second, to explain each discontinuance process, we examined when and how various SRMs (see Table 1) were enacted. In so doing, we mapped the progression of the IS discontinuance process in each case and examined how various SRMs were ceased in some situations and interestingly were intensified in others. We also coded how ceasing and/or intensifying certain SRMs affected the cessation or intensification of the other SRMs over time. This way, we accounted for the interactions that emerged during the process. The result showed four distinct processes of IS discontinuance, that they varied based on which and when various SRMs were enacted (described in the following section and summarized in Figures 1–4).

Third, we examined the four discontinuance processes to make sense of when various discontinuance actions took place, how they were triggered by certain decisions and external events, and how they interacted over time. Through an iterative process, we identified four common sets of interrelated actions—we call them *discontinuance phases*—each of which captures a specific temporal pattern of ceasing and/or intensifying a set of SRMs with a distinct impact on the progression of IS discontinuance process. The identified phases are

(1) **Realization**: the legacy IS is critically scrutinized, yet still remains as the dominant IS

(2) **Reversion**: the legacy IS is further developed to cope with new expectations

(3) **Handover**: the momentum of the legacy IS is leveraged to nurture the new IS path

(4) **Marginalization**: the development and usage of the legacy IS is gradually reduced

For each phase, we coded for (1) the triggering events, (2) the actions that ceased or intensified SRMs, and (3) the interactions within each phase (see Tables 6–9 and Tables A1–A4 in Appendix A for more details). We observed that the phases are not linear or sequential, rather discontinuance often emerges as an iterative process. Thus, we examined the interactions between them (see Table 9 and Table A5) for more details).

Table 3. Summary of Collected Data

Data Sources and Information Items	Contributions to Empirical Findings
Exploratory interviews (Total 16 interviews, on average 40 minutes)	
• SmallCo: three interviews with industrial association heads (3)* • MedCo: three interviews with the former manager (1) and industrial association heads (2) • LargLenient: four interviews with the former manager (1), industrial association heads (2), and clients (1) • LargStrict: four interviews with industrial association heads (2) and clients (2)	To explore (1) the company background, (2) the story of the technological replacements, and (3) the context of technological replacements (e.g., economics, industrial, and regulatory context, external relations, their clients, and their market positions, and their conditions regarding organizational commitments and technological interdependences)
Formal interviews (Total: 33 interviews, on average 1 hour)	
• SmallCo: eight interviews with top managers (3), technical managers (3), and experts (2) • MedCo: eight interviews with top managers (3), technical managers (3), and experts (2) • LargLenient: eight interviews with top managers (3), technical managers (2), experts (2), and clients (1) • LargStrict: nine interviews with top managers (3), technical managers (2), experts (2), and clients (2)	To capture (1) the chronology of events in the discontinuance process, around tasks, actors, technologies, and structures, (2) actions applied to the established IS path, by exploring actors and their beliefs, views, positions in the replacement process; main decisions and rationales behind each; discussions, controversies, and agreements, and (3) the practical tensions and how they were dealt with
Documents	
Documents on corporate websites; product specifications and technical descriptions; intranet pages on processes, routines, regulations, messages from the managers; support manuals and rules; contracts with clients and providers; termination letters to the clients; clients feedback; and R&D plans and procedures	To capture (1) background of the companies and the technological replacements, (2) formal actions and decision during IS discontinuance, and (3) the outcomes of IS discontinuance actions

Fourth, we identified similarities and differences between the cases depending on (1) whether some phases are *present*, (2) *when* the phases start and finish, (3) what specific *sequences* between the phases are important for the progression of IS discontinuance, and (4) which temporal *iterations* (e.g., back and forth) between the phases that occur during discontinuance. We then relied on organizational path theory (Schreyögg and Sydow 2011) to explain the progression of the discontinuance process (horizontal axis in Figures 1–4) as a function of ceasing and intensifying *multiple interdependent* SRMs (vertical axis in Figures 1–4). Once we mapped the four temporal scenarios into our theoretical framework, we examined how IS discontinuance was differently shaped by the commitments and the interdependences of legacy IS.

During data collection and analysis, we adopted several strategies to enhance the accuracy, validity and transparency (Gibbert et al. 2008) of our study such as the triangulation of data sources and informants (see Appendix B for more details).

Findings

We first describe the four cases of IS discontinuance. For each case, we present a process account that explains how discontinuance progressed as various SRMs were ceased (downward movements in Figures 1–4) or intensified (upward movements in Figures 1–4).

Case 1 (SmallCo): Discontinuing an IS with Limited Commitments

The first case of IS discontinuance relates to SmallCo, which embarked on discontinuing a separate document management product running on MS-DOS. The company started working on the product in 1991, selling it to about 10 clients. A team of four people designed and developed the product, and collaborated with three people giving support and maintenance services. The CEO coordinated these tasks and marketed the product to new clients. The CFOs held the position of financial officer, acting as a main shareholder, next to the CEO.

In early 1993, when the company was expanding its market, the CEO and the technical experts realized that there was no reason for investing further in MS-DOS technology since MS-Windows was already on the market. This led the technical team to reexamine the details of their existing production tools and methodologies by scrutinizing their functions and shortcomings. Interestingly, many of the discussions appeared to be novel even for the designers and developers who

Table 4. Analysis in Process

Analysis step	Empirical grounding	Key (provisional) codes	Analytical abstractions	Theory engagement
1. Mapping the empirical *stories* of IS discontinuance: who did what when?	Constructing the timeline of actions and events to map the discontinuance process in each case	Open coding for '*who*', '*did what*', '*when*', '*for what purposes*', that had 'which consequences' for discontinuing legacy IS	Abstracting *discontinuance actions* taken by embedded actors and external triggers and conditions that shaped discontinuance practices	Iterating with the conception of IS discontinuance
2. Explaining the discontinuance processes by coding for self-reinforcing mechanisms	Examining the chain of actions and events to understand how they enact each of the SRMs in each of the four discontinuance processes	Coding for the self-reinforcing mechanisms (listed in Table 1) in each process and how they were ceased and/or intensified through actions and events	Mapping and explaining the progression of the discontinuance process based on actions that ceased and intensified SRMs (Figures 1-4)	Iterating with organizational path literature based on two analytical dimensions: (1) the overall changes in the IS path and (2) the specific changes to the SRMs (listed in Table 1)
3. Identifying discontinuance *phases*	Inspecting the temporal patterns of ceasing and intensifying various SRMs in the discontinuance processes	Coding for temporal patterns of changes in the overall legacy IS by pinpointing *triggers* such as 'key decisions', 'shifts in strategies', and 'launching new products'	Articulating *discontinuance phases*: *'A specific temporal pattern of discontinuance actions that cease and/or intensify certain SRMs, resulting in a distinct temporal pattern in the discontinuance process'* (Tables 6-9)	Iterating between the SRMs and their enactment and its impact on the discontinuation of the overall path
4. Explaining the different patterns of IS discontinuance across the cases	Mapping discontinuance phases over time and examining their temporal patterns of interactions in each case (see Appendix 1)	Coding for temporal patterns based on 1) 'when' each phase started and finished, 2) what 'sequences' are relevant between the phases and 3) 'iterations' within and between phases, across the four cases	Comparing different patterns of IS discontinuance process across cases to examine *how conditions shape differently IS discontinuance* (see Figures 2-5)	Theorizing the temporality and interactions of IS discontinuance process; extending IS discontinuance and organizational path literatures

had been immersed in MS-DOS for several years. For them, many technological features had become taken for granted and scarcely noticed, especially the basic assumptions regarding data storage and communication protocols.

The technical team and the CEO had discussions about which aspects of MS-DOS and the "*stuff that they designed and implemented based on it*" such as the code, testing algorithms, and conceptual designs, were becoming obsolete and should be abandoned, and which were still viable and should be retained. Telling the viable and obsolete aspects apart was crucial because the young developers were eager to quickly jump forward to the new technology and discredit the entire old technology and the related "stuff," even though the basic design of the existing system was still relevant for implementing the MS-Windows system. Besides, it took more than a year to convince the financial officer that the current demand for MS-DOS products would not last long because he had limited knowledge of technological trends and was influenced by the fact that "*the products work and clients are asking for them*."

As a turning point, in early 1994, the financial officer and the CEO agreed to split their shares: The CEO kept the brand of the old product and the financial officer kept the license and maintenance contracts and created a new company. This triggered the second discontinuance phase through which SmallCo gradually marginalized the legacy systems and products. One of the senior developers and two support staff left SmallCo to join the financial officer's company. SmallCo redirected most of the clients' support requests to their partner company, although in a few cases, they helped them fix the bugs. The limited client base allowed SmallCo to start the new document management product as a totally distinct product:

> *In the DOS shift, we had an easy job to simply start a whole new product. For a few clients who used our old system, a small fraction of their legacy data that could be reused in the new systems were re-entered to the new systems, as if they did not exist it in the old products. The only thing we reused from old system was its brand, which helped us promote the new product* (the CEO of SmallCo).

As Figure 1 shows, the discontinuance at SmallCo was a straightforward process through which the legacy products, technologies, routines, and actors were gradually marginalized.

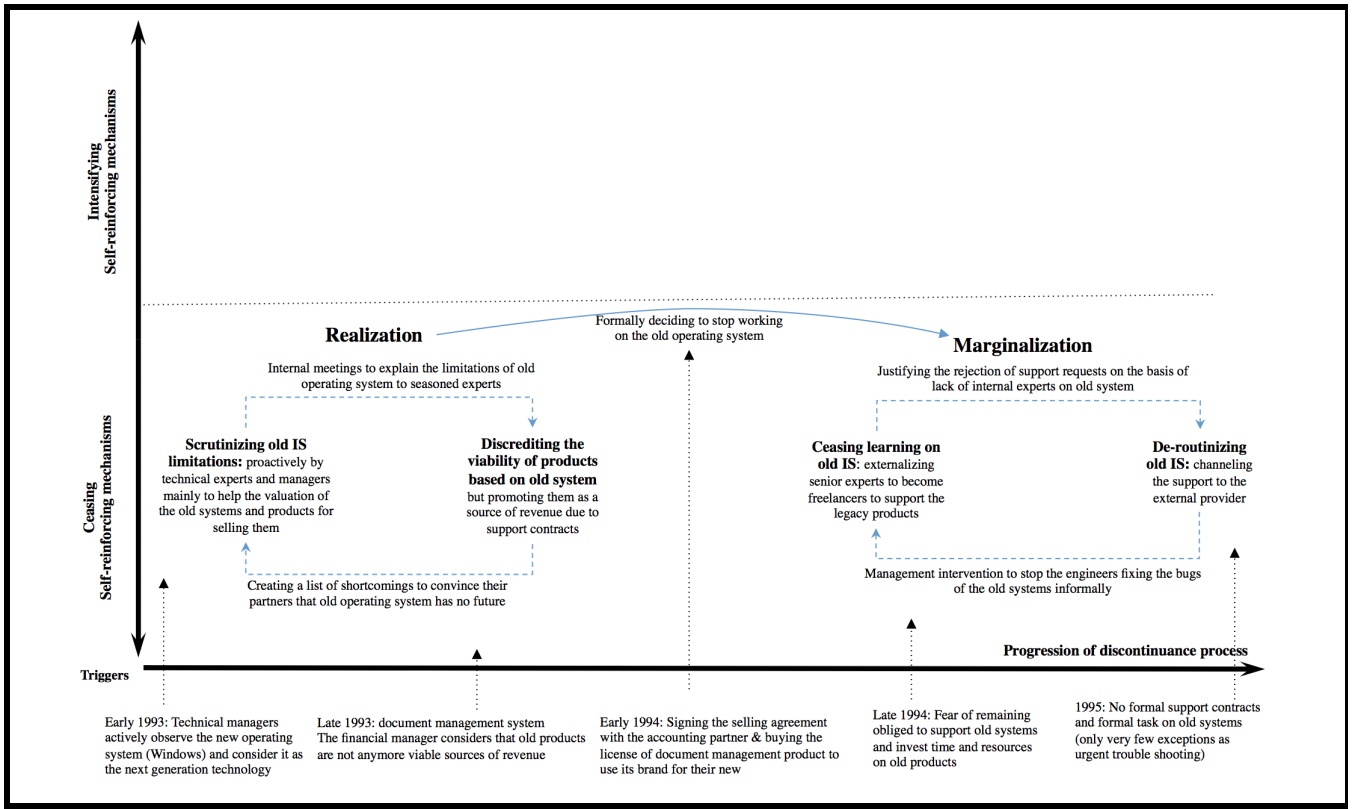

Figure 1. Discontinuing an IS with Limited Commitments (SmallCo)

Case 2 (MedCo): Discontinuing an IS with Extensive Commitments

The second case of IS discontinuance relates to MedCo, which had developed a document management product running on MS-DOS in 1988. The company had a team of 15 developers devoted to MS-DOS and a team of 20 people providing support, both managed by the Technical Manager, one of the three founders of the company. He was also active in promoting products to new clients and managing relations with approximately 50 clients. Most of the clients were large, public organizations that MedCo tried to please: "*We do even more than what is needed to delight our clients*" (Technical Manager).

In 1995, discussions about MS-Windows arose, especially among some designers who had recently joined MedCo. The Technical Manager arranged for a set of informal, open discussions in his team on: *"Are we sitting on dangerous ground, I mean MS-DOS*?" Given the range of tools, methodologies, and products developed on MS-DOS, it took more than a year to determine which of them had or would become obsolete. However, the general tendency of the senior experts and the managers was to consider most of the legacy technology and processes as viable since they had developed strong skills related to MS-DOS.

In parallel, the Technical Manager and the CEO had been concerned about the lack of product and market focus, feeling that they were "*carrying too many melons in their hands.*" In late 1996, they decided to take advantage of the shift from MS-DOS to MS-Windows "*as an excuse to kill many other products and get focused on the core document management product*" (the CEO), for which they had their main market share. This decision led a group of young, newly hired developers to discredit the current product, production tools, and routines, partly through the argument that they were based on an outdated technology. As a result, they became overly critical of MS-DOS and eager to move forward to MS-Windows.

Although the managers and most of the technicians realized the shortcomings of their existing systems, they still had to reap economic revenue from selling and supporting their legacy products since their clients continued to demand support services. In addition, the new product was still not ready, and the managers had grave doubts about when it could be implemented due to the instability of early technological developments. Hence, there was a risk of too early a depar-

ture from MS-DOS by some technicians who were eager to jump forward.

In response, for three years the Technical Manager and the heads of the technical teams actively engaged in relegitimizing their legacy systems and in improving them to cope with new client expectations. This was challenging because internal teams had already discredited the base technologies. Careful reframing of the legacy systems and sometimes decoupling the internal technical discussions from external marketing promotions was required:

> *In the last months, we have had meetings with our designers and programmers on the limitations of the MS-DOS systems and all the opportunities we are missing for accessing the systems in a real-time manner. But, yesterday, I had a demo session for a client who was interested in buying our old document management. Like other demo sessions, I took Mike, one of my senior technical fellows, to the demo session, mainly to talk about the technical features. I generally pitched the current product. Then I passed the discussion over to Mike, who was sitting just opposite me on the other side of the table. Influenced by the past weeks' discussions, he suddenly became critical and talked about the limitations of MS-DOS. I was very lucky that I managed to quickly tread on his toes without being noticed by others, to prevent him from destroying the image of our products* (Marketing Manager)

Furthermore, the exposure to the shortcomings of the legacy systems stimulated a range of new development projects to enhance the capabilities of the old products and cope with new expectations and needs. As a result, specialized teams faced a new surge of tasks to *learn more* in order to improve the old systems. Sometimes these learning projects were extended beyond the normal change requests because clients were requesting that the features MS-Windows was promising be implemented in the old systems. In addition, the experienced MS-DOS folks were also eager to push the boundaries of the current technology and learn new ways to make it comparable with the new technology. A senior project manager at MedCo commented:

> *When we developed DOS systems, many of our clients started using new LaserJet printers. But in the MS-DOS environment, there was no built-in driver for these printers. This made us go back to our DOS systems and learn even more about how DOS operates signals that can communicate with these printers. This actually made us learn even more than we knew before, because we had to carefully examine the signals that are specific to DOS protocols.*

In 2001, the new MS-Windows based document management system was ready, and MedCo started marginalizing the MS-DOS based legacy systems by limiting their *major* changes: "*changes that required further developing our capabilities in some old domains*" (Production Manager of MedCo). Externally, the support team helped clients gradually replace their legacy system with the new one. Internally, for fresh experts it was just fun to move to MS-Windows. Yet for others with long experience in MS-DOS, it was extremely challenging because the shift had made almost their entire expertise obsolete. Accordingly, the company helped them move to other companies where they could find positions related to their expertise.

Nevertheless, marginalizing MS-DOS took more than 3 years because MedCo had to frequently update the legacy systems to address its clients' requests, even long after stopping support contracts*: "It was an open, never-ending phase, meaning that you should be ready to go back and fix their issues"* (the support team head of MedCo).

Case 3 (LargStrict): Discontinuing a Set of Interdependent IS with Extensive Commitment

LargStrict was the second largest software company in the national market, selling 10 core office automation products and 45 customized versions to more than 200 clients. The products were interdependent to support each other's data, to act as the modules of a comprehensive office automation solution. In 1993, LargStrict recognized the trend toward MS-Windows. Many designers and programmers started looking into the shortcomings of MS-DOS regarding how to make systems faster through parallel processing and more attractive through new user interfaces.

However, the head of the production department had a more fundamental concern:

> *For me, the basic problem is that our products are like pieces that have been glued to each other; yet they are not inherently designed as an integrated total solution.*

In late 1994, he managed to convince the CEO (who was a former production manager) and most of his senior designers to take advantage of MS-Windows for discontinuing MS-DOS systems. Yet it proved difficult to distinguish between viable and obsolete aspects that were interdependent, espe-

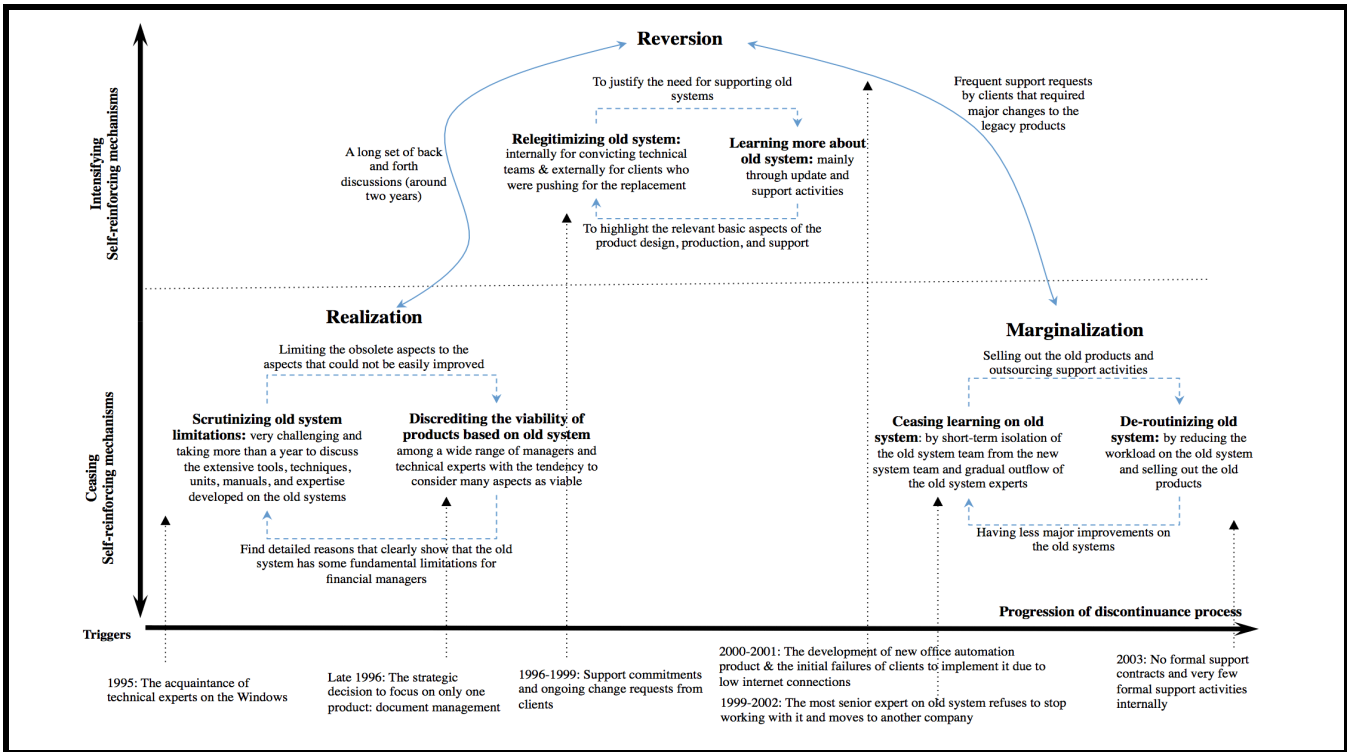

Figure 2. Discontinuing an IS with Extensive Commitments (MedCo)

cially because being viable or obsolete was, to a large extent, dependent on "*other interconnected pieces with which, for instance, our current testing platform should work"* (a senior designer).

In 1995, LargStrict launched a parallel production line to develop new integrated products on MS-Windows, while continuing the development and support of MS-DOS systems. Given the extensive client base and the still-growing revenue stream from the legacy products, the Production Manager and the heads of the technical teams actively encouraged working on MS-DOS, especially for many programmers and designers who perceived it as an inferior task compared with working on the new systems.

The development of new products was delayed 3 years more than the initial expectation; in total, it took 7 years to develop the new products. The delay was mainly due to the complexity of designing the new products in such a way that they supportd the legacy data and functions of the old products *"and their historical interdependences,"* said the Production Manager. He continued, *"We had more constraints than opportunities, as if we had to replicate the same old monster in the new product designs."* Although it was challenging, LargStrict managed to resist many major change requests to the legacy products in the last 2 years before launching the new products. This helped them reduce further complications in the new product designs. As noted by the LargStrict Product Manager,

> *It is not only the general data and functionalities that have to be transferred to the new products; every small update and change we now do on the legacy IS needs to be considered for migrating data and functionalities to the new systems.*

In 2000, the new products were developed, yet there still was a major imperative: developing gateways and convertors to automatically migrate the data from legacy systems and map their functions to the new products. Transferring data and functions from a set of interdependent legacy products to a set of interdependent new products was additionally challenging since it required not only transferring single pieces of data, but also examining and replicating those interdependences in the new systems: *"It is like you want to move all the pieces of furniture in your house to the new house while keeping them connected"* (the Production Manager). This in turn required deeper learning on MS-DOS for developing and supporting *gateways*, which in turn, delayed the discontinuance for nearly a year.

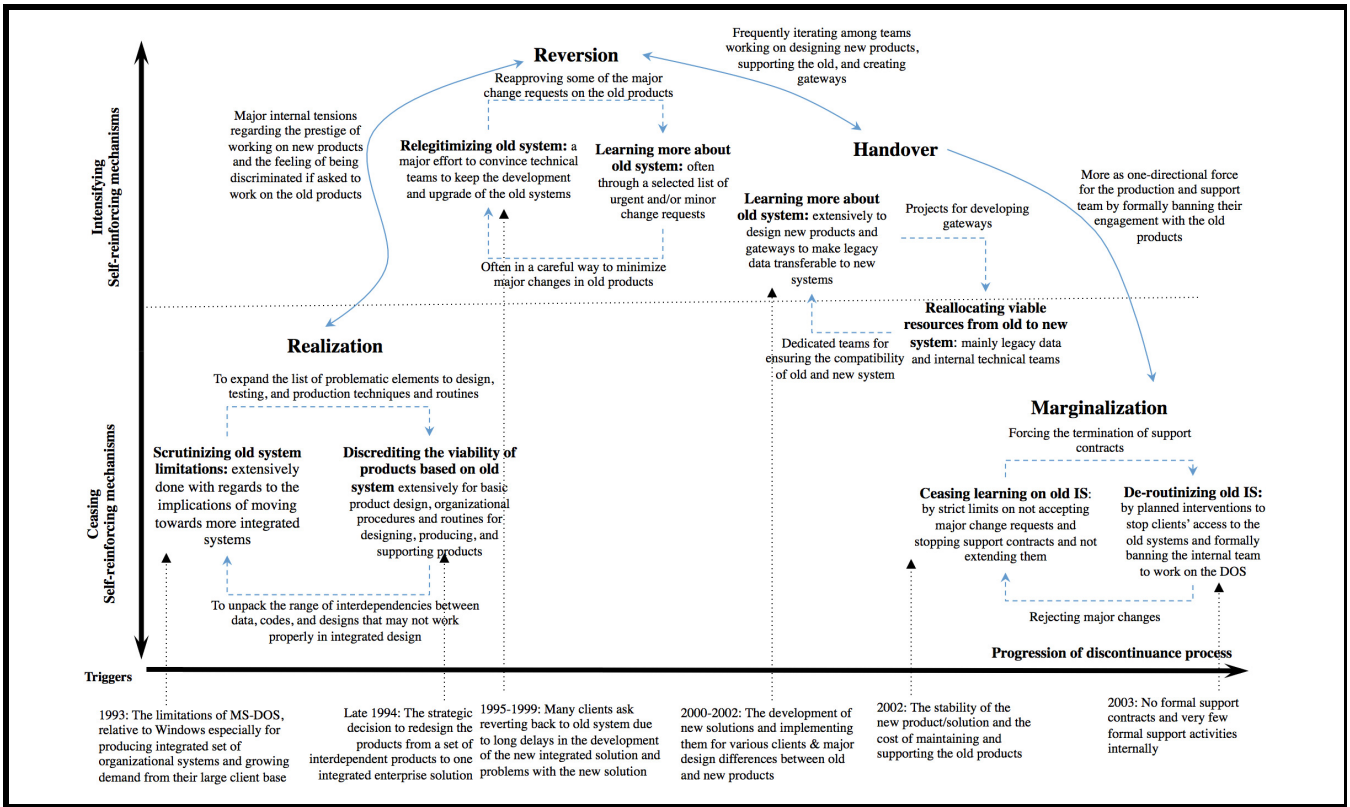

Figure 3. Discontinuing a Set of Interdependent IS with Extensive Commitments (LargStrict)

Eventually, in 2002, LargStrict started marginalizing MS-DOS by setting a strict deadline for their clients to replace their products with the new ones. The company benefitted from its strong bargaining power to resist the extension of support contracts. Beginning in 2002 and mainly during 2003, LargStrict appointed its HR manager to develop a network of more than 15 spin-offs by encouraging their MS-DOS specialists to create their own start-ups. The company had pursued this strategy since its establishment as a way to remain agile, while growing in terms of the market and product portfolio. In this way, the company redirected the support requests to the spin-offs; it also managed to speed up the transition of a considerable number of experts who were still far from ready to work on the new systems, having learned little about them, and sometimes having quite a bit of difficulty in giving up the wealth of expertise they had acquired. Figure 3 summarizes the discontinuance process at LargStrict.

Case 4 (LargLenient): Discontinuing an Integrated Legacy IS with Extensive Commitments

Since its establishment in 1987, LargLenient had developed an integrated set of office automation products running on MS-DOS. LargLenient grew into the second largest software company in the national market by focusing on large clients. Since 1987, LargLenient developed 18 office-automation products based on MS-DOS, comprising more than 50 customizable modules used by around 700 clients from 20 different industries.

Since the company proactively investigated the newest technological trends, in 1993 it opened a major R&D project to compare MS-DOS with MS-Windows. Meanwhile, the Production Manager and his team had considered the possibility of a complete redesign of their products to make them more customizable. Later, in 1994, both the Production Manager and the R&D Manager agreed that the transition from MS-DOS to MS-Windows be used for a complete redesign of their office automation products.

LargLenient had invested in developing MS-DOS systems through specialized design, production, testing, and support teams. For instance, one team was dedicated solely to optimizing applications through parallel processing. They already mimicked the idea of parallel processing and incorporated it in their product designs, *"as if we tweaked MS-DOS systems to execute parallel processes,"* said the R&D manager. He

continued, *"We were so specialized in MS-DOS that at the time that Windows came, we already had almost all its new features implemented in a more efficient way by MS-DOS."* Although this helped the company support the legacy systems, it appeared to become problematic if it were pushed too hard, "*leading us to be buried in the MS-DOS grave, not because we were not capable of adopting Windows, but rather because we were too capable in developing MS-DOS."*

In early 1996, the company launched new R&D and production lines to develop new products, while keeping most of their current technical teams active on the maintenance and support of MS-DOS systems for about 7 years. The company's strategy of accepting almost all client requests demanded even more extensive support and development of the legacy systems than before because clients were now asking for MS-Windows' features: "*Clients' systems are our systems! They tell us when we can fully drop DOS*" (the Production Manager).

From 1996 to 2000, LargLenient straddled two lines of activities in parallel: (1) development and support of MS-DOS products, and (2) designing the new MS-Windows systems. For two years, the teams were carefully isolated from one another by locating them in distinct departments and even on different floors to prevent *"injecting the wrong way of thinking and acting to the new systems"* (R&D manager). Yet, soon, the company realized that a third team was crucial to take care of designing and implementing tools to migrate legacy data to the new products and map their functionalities: the handover team. The extensive technological interdependences required examining many technological components to ensure that data and processes running on the legacy IS were properly transferred to the new system.

Therefore, since 2002, three production and support departments were active in parallel. Since the company was open to implementing even major changes to the legacy systems, the Production Manager and a team of senior experts were active to ensure that

> *We [would] consistently coordinate three major, complex lines of activities: (1) supporting and updating the old systems; (2) designing the new systems in such a way that we can comply with the backward compatibility of the data and main functions; and (3) the transition of the data and functionalities of the old to the new systems. Interestingly, for several years, we had a large number of our experienced engineers working on the gateways and connections between the old and new; even comparable with our teams dedicated to the new systems"* (the R&D Manager).

The company had to first think of designing gateways for transferring the legacy data *before* designing and implementing a specific part of the new systems:

> *Many times, we had a very brilliant design for the database of the new systems to make it faster. However, we had to put it on hold, ask our transition team to learn even more about DOS to see if they could find a feasible solution to transfer legacy data. Otherwise, we could simply create nice products that none of our clients would use* (Production Manager).

In 2002, the new products were implemented for the new clients, and the negotiations with traditional clients to replace their existing products were successfully concluded. Yet, LargLenient did not force a strict transition upon clients, but instead was lenient to keep the legacy systems running parallel to the new systems through supporting the gateways.

Nevertheless, the company realized that maintaining gateways for an extended period could backfire. Gateways had to be deployed *temporarily only* to hand over the data and functions from legacy to new systems. Therefore, LargLenient adopted two strategies: They fixed specific timelines for the removal of gateways, and they built one-way gateways that mainly transferred data from the legacy to the new system but not the other way around, thus limiting opportunities to use the legacy system via the new system.

The full discontinuance of the legacy system took more than 10 years (1995–2005). For each set of clients, LargLenient had to gradually reduce their reliance on legacy systems through carefully handing over their legacy data and reducing the usage of the legacy systems. In fact, there was no clear end to the discontinuance process despite having no formal support obligation after 2005. As a result, the company learned how to become more efficient in "*going back, opening the wounds, fixing the old systems, and not getting stuck there,"* for example, by asking their very experienced engineers to take care of the legacy systems requests:

> *I never ask junior engineers to fix the MS-DOS systems, they may go there and never come back! Although it is quite expensive, I always ask our senior designers to quickly fix the old systems* (head of support team).

Table 5 summarizes the four discontinuance cases.

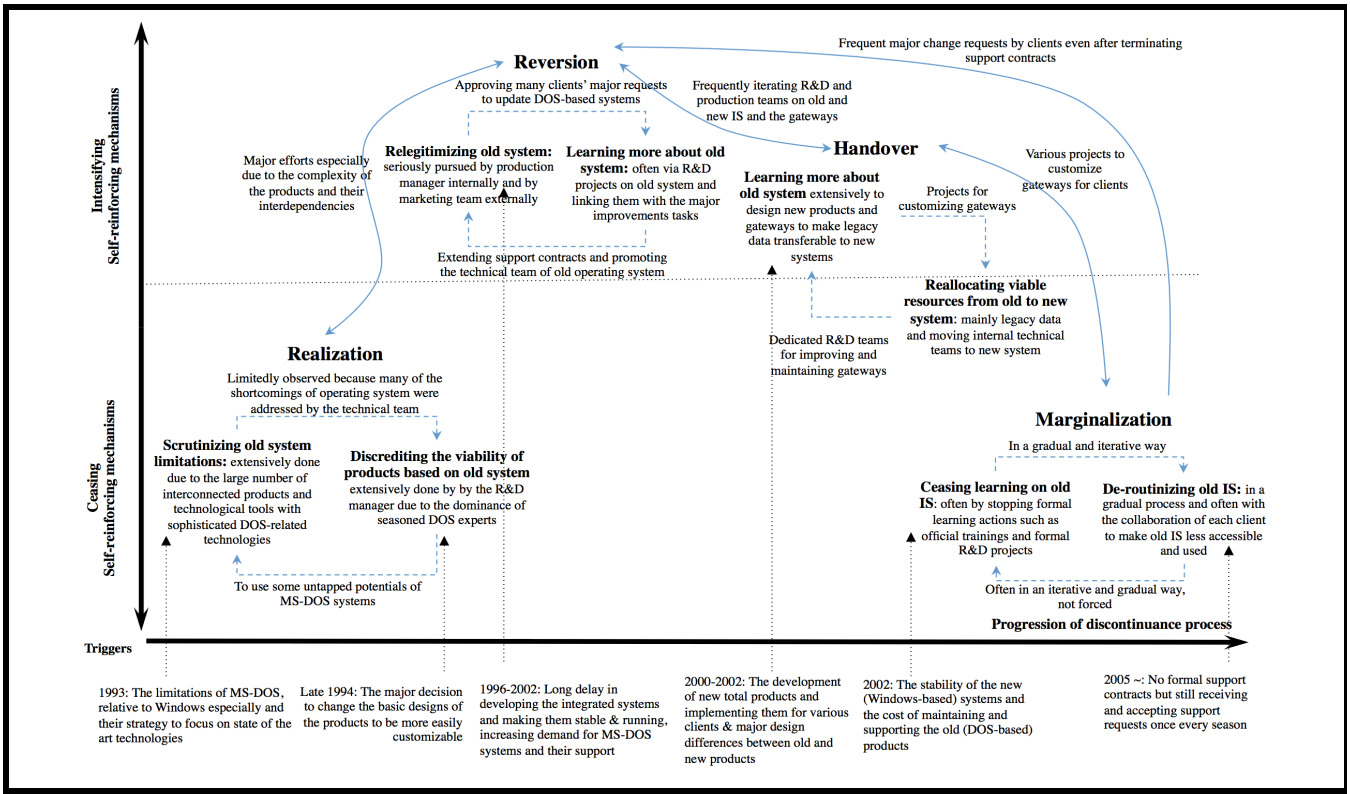

Figure 4. Discontinuing an Integrated Legacy IS with Extensive Commitments (LargLenient)

Table 5. Summary of the Four IS Discontinuance Cases

Case 1 (SmallCo): Discontinuing an IS with limited organizational commitments (3 years) • *Relevant phases and iterations*: Discontinuance mainly involved realization and marginalization, with limited reversion and handover since new IS was distinct from the legacy with limited support commitments. • *Practical challenges*: Convincing nontechnical fellows on the need to depart from legacy systems when they still provide revenue
Case 2 (MedCo): Discontinuing an IS with extensive organizational commitments (8 years) • *Relevant phases and iterations*: Discontinuance mainly involved realization, reversion and marginalization, especially with many iterations between realization and reversion to avoid premature departure from the legacy IS and remain able to meet extensive commitments; too early marginalization was hurting and needed; considerable iteration between reversion and realization. • *Practical challenges:* Avoiding too early departure from legacy IS to ensure fulfilling legacy commitments
Case 3 (LargStrict): Discontinuing a set of interdependent IS with moderate organizational commitments (11 years) • *Relevant phases and iterations:* Discontinuance involved realization, reversion, handover, and marginalization. There were many iterations between reversion and handover to ensure that historical interdependences were transferred to new systems; yet a more straightforward marginalization with limited iteration back to reversion and handover. • *Practical challenges:* Designing new gateways to support historical data and their interdependences
Case 4 (LargLenient): Discontinuing a set of integrated legacy IS with extensive organizational commitments (more than 13 years) • *Relevant phases and iterations*: The discontinuance process was a highly iterative process within and between the four phases, especially with several cycles involving reversion, handover, and marginalization. There seemed no clear-cut end to the process due to the historical commitments. • *Practical challenges:* Balancing between fulfilling historical commitments and not being buried in the grave of legacy systems.

Discussion

Toward a Process Account of IS Discontinuance

Our findings depict four generalizable discontinuance phases: *realization, reversion, handover,* and *marginalization* (see Tables 6–9). Depending on which phases, when, and the interactions that emerged in the process, we can explain how the discontinuance process can unfold differently. We observe that two discontinuance phases are present in all cases: realization and marginalization. In particular, the discontinuance process at SmallCo, where legacy IS involved limited commitment and interdependences, consisted of merely these two phases. First comes *realization*, during which the legacy IS is critically scrutinized, yet still remains as the dominant IS. During realization, the indwelling mechanism is undermined by scrutinizing the details of the legacy system. In addition, exposing the limitations of the legacy technologies, products, and routines ceased their legitimacy.

Second comes *marginalization*, during which the development and use of the legacy IS are gradually stopped. Marginalization involves ceasing to learn about the legacy system's technology, and de-routinizing the legacy IS by isolating the technical teams and imposing social and material restrictions on updating and using legacy technologies and products (Table 7).

The presence of realization and marginalization phases in the four cases confirms what the current literature suggests: for discontinuing legacy IS, we need to trigger the users' and managers' intentions to stop using it (Furneaux and Wade 2011), disrupt users' habits (Polites and Karahanna 2013), and delegitimize legacy choices and actions (Sahay et al. 2010).

Nevertheless, the process does not merely consist of realization and marginalization. As observed in Cases 2, 3, and 4, when organizations have commitments to legacy IS, *reversion* becomes a key phase, after realizing that legacy IS needs to be discontinued, yet before engaging in marginalizing it (Table 8). Reversion is not a failure of discontinuance but rather an important phase for its progression, through which the existing IS is further developed to cope with new expectations. Counterintuitively, during reversion, the legitimization and learning SRMs are *intensified* to prevent premature discontinuance of the legacy IS before the new products, production tools, routines, and expertise are properly developed. When organizations have extensive commitments to a legacy IS, ditching a system too early can lead to prematurely abandoning its viable aspects and failing to comply with organizational commitments.

As observed in Cases 3 and 4, when the legacy IS has interdependences, an iconic *handover* phase emerges, through which the momentum of the legacy IS is leveraged to nurture the new IS (Table 9). In particular, during handover, organizations need to intensify learning on the legacy IS in order to develop gateways for transferring the viable data into the new system. In our empirical cases, the gateways were imperative to break the resource complementarity of the legacy IS by moving viable resources (e.g., legacy data) to the new IS. Otherwise, not only might their clients revert back to the legacy products, but also the new products would not gain sufficient momentum to become dominant. In other words, temporarily intensifying the learning SRM was necessary to cease the resource complementarity SRM, especially since the legacy IS had extensive interdependences that had to be understood and transferred to the new IS.

Moving from Case 1 to Case 4, we observe that the process increasingly involves *interactions* between the various phases, thus gradually opening opportunities for the progression of the discontinuance process. As summarized in Table 10, the interactions enable us to explain how ceasing a specific SRM can facilitate or hamper the cessation of other SRMs.

This is well illustrated in Case 4, where the successful completion of the discontinuance took many years and involved many iterations between the reversion, handover, and marginalization phases. These iterations were triggered by both extensive commitments and interdependences involved in the legacy IS. First, given the organization's historical investment in the legacy systems and the organization's culture of satisfying the clients, the discontinuance process was not completed until the very last client was fully transferred into the new system. Second, the development of a wide range of products to act as modules of an integrated solution resulted in extensive technological interdependences. Therefore, the iteration between reversion and handover seems crucial to ensure the changes made into one specific aspect of legacy systems would be considered in the design of new products and gateways.

How Process View Advances Our Understanding of IS Discontinuance?

A process view implies that we should seriously consider the *iterations* within and between phases in explaining the progression of the discontinuance process. Otherwise, separate interventions to cease each SRM may fail to break the established IS path. Furthermore, the *timing* of acting on various SRMs is pivotal for the progression of the discontinuance process. For instance, the reversion phase shows that a process view is crucial to explain *when* ceasing a SRM that sustains

Table 6. Realization: The Legacy IS Is Critically Scrutinized, Yet Still Remains as the Dominant IS		
	Theoretical Abstractions	**Empirical Observations**
Triggers	• Technological changes • Acquaintance with new IS • New requests by leading clients	• The change in the base technology: operating system and product design architecture • Often technical experts were leading to raise the flags of obsolesce • New change requests due to new real clients' needs and due to the willingness to have the latest technology
Actions by embedded actors	**Scrutinizing legacy IS**: The *indwelling* SRM is ceased as more details of the legacy systems are scrutinized	• *Technical teams and middle managers and later top management and sales and marketing* made sense of the relative disadvantages of legacy IS, compared with new IS • *Financial Managers* often were reluctant to admit the obsolescence of the established IS since they were still profitable for support contracts • *Technical development teams* were searching for the comparable advantages and capabilities of the legacy IS in comparison with the new IS • *Senior experts with high specialization in legacy IS* were posing novel questions about legacy IS
	Discrediting the viability of (parts of) legacy IS: The *legitimization* SRM is ceased as more organizational actors critically reflect on obsolete aspects of legacy IS	• *Product managers in discussion with technical teams* were distinguishing between viable and obsolete aspects of legacy systems and discussing the boundaries between them • *Forward jumpers* were discrediting legacy systems by highlighting the new technological opportunities and deficiencies of the legacy IS • *Marketing managers and agents* were carefully decoupling between internal and external legitimacy of the legacy systems
Interactions	**Scrutinizing ➔ discrediting**	• Paying attention to the detailed, technical characteristics and examining various potential technological features that appeared to be limiting the capacities and functions of the legacy IS • Reopening the list of shortcomings of the technology that were worked around and adjusted locally in past upgrades and maintenance
	Discrediting ➔ scrutinizing	• The discussion about the boundaries between old-viable and old-obsolete required scrutinizing some detailed technological features and aspects in more details (to see which aspect is really the problematic aspect)
Outcomes	• The limitations of legacy technologies, tools, designs, and routines are exposed • Most of technical teams are eager to jump forward to the new systems	• Widespread belief that MS-DOS has no future; so much so its associated production technologies • Divisions between technical teams with the attitude to quickly jump forward and the more seasoned experts and managers who concern their historical profession and the mid-term stability of the company, respectively

Table 7. Marginalization: The Development and Practicing of the Legacy IS Is Gradually Reduced

	Theoretical Abstractions	Empirical Observations
Triggers	• The fear of keeping the legacy IS next to new • High cost of maintaining legacy systems • The stability of new products	• Often after having it tested internally and the products successfully launched for some clients • Shrinking market specially for the cases that they had major delay in developing new systems in comparison to the other competitors)
Actions by embedded actors	**Ceasing learning about legacy IS:** The *learning* SRM is ceased as the scope, depth, and frequency of learning practices related to legacy IS domains are reduced	• *Technical and HR managers* gradually stopped R&D projects, training, and hiring on the domains specific to legacy systems • *IS Product managers in negotiation with support teams* were carefully distinguishing between major and minor changes into the systems based on their learning commitments • *Production and HR managers* isolated legacy and new systems teams and located them in distinct organizational units • *R&D and HR managers* proactively prevented narrow specialization in domains that might become obsolete quickly
	De-routinizing legacy IS: The *routinization* SRMs is ceased as the scope and frequency of working on and with legacy systems is limited	• *Marketing and Support managers* specified clear deadlines for terminating the support of legacy systems • *Support teams* paralyzed the legacy systems in order not to be used by clients that already adopted new systems • *HR managers* helped some seasoned experts to move to other companies where they could find a relevant position • *Production managers and support teams (when recommending to clients)* removed the technological objects (e.g., operating systems, programming tools, and test tools) to prevent working on the legacy systems • *Top managers (often CEOs)* supported the teams of seasoned experts who were not willing to move to new domains to create their spin-offs • *Production managers and technical teams of legacy IS* documented the legacy systems to allow them move on to new systems teams and for future occasional requests
Interactions	**Ceasing learning ➜ de-routinization**	• When systems were not developed further, clients have lower chance to keep using them (terminating support contracts)
	De-routinization ➜ ceasing learning	• The fewer users and clients use a legacy system, the less is the need for support and updating activities (thus less learning)
Outcomes	• Legacy technologies and production tools and routines have limited use • Clients are migrated to the new ones	• Terminated support contracts and support activities • No access to the legacy products and technologies both internally and in the side of the clients • Documents regarding how legacy systems can be supported in case of emergencies • Specifying teams of experts who act on occasional changes requests to the legacy IS

Table 8. Reversion: Legacy IS Is Further Developed to Cope with New Requirements and Expectations		
	Theoretical Abstractions	**Empirical Observations**
Triggers	• Ongoing requests from the clients on the legacy • Not having new systems ready • Risk of too early departure from legacy systems	• Highly profitable and even growing demand regarding the support contracts of the legacy IS, especially since they were still working • Delays in developing the new systems especially when they were technologically interdependent • The internal pressure from forward jumpers to quickly drop the legacy IS, recognized as a major risk
Actions by embedded actors	**Relegitimizing legacy IS:** The *legitimization* SRM is intensified as more internal actors and clients regard legacy systems as still the viable solutions	• *Product managers* were promoting legacy IS as still viable solutions in short-term • *Middle managers* had to convince forward jumpers to not disregard the entire legacy systems as obsolete • *Product managers and marketing agents* were constantly highlighting the relative advantages of legacy systems against the immature versions of new technologies
	Learning more about the legacy IS: The *learning* SRM is intensified as deeper and more learning efforts were devoted to improve the legacy systems	• *Seasoned, specialized experts* were actively identifying the shortcomings of legacy systems and improve them in an effective way • *Senior experts in legacy IS* re-discovered untapped potentials of legacy systems and developed them • *Product managers and development teams* launched new development projects to improve the functionalities of legacy systems
Interactions	**Relegitimizing ➜ learning to improve**	• Knowing which aspects of the legacy IS are considered as viable and thus can be and should be further improved meanwhile • Specifying the boundaries of learning to prevent over-learning in obsolete domains
	Learning to improve ➜ relegitimizing	• The improvement of legacy IS makes it more as a viable solution to be presented to the clients • The cost of learning on some basic domains prevents delegitimizing them for the clients
Outcomes	• Further development of the legacy products and technologies • A selective team of experts dedicate to knowing and supporting legacy products	• The tension between keeping a team of experts busying with support and update of the legacy products and their willingness to move on to the new systems • New changes into the legacy IS need to be replicated/reconsidered in the development of the new systems

Table 9. Handover: The Momentum of the Legacy IS Is Leveraged to Nurture the New IS Path

	Theoretical Abstractions	Empirical Observations
Triggers	• The development of the new systems and ensuring they are stable • The shrinking market on the legacy systems	• Often after having it tested internally and the products successfully launched for some clients • Shrinking market specially for the cases that they had major delay in developing new systems in comparison to the other competitors)
Actions by embedded actors	**Learning more about legacy IS to connect legacy and new IS:** The *learning* SRM is intensified as efforts and experts were increasingly devoted to learn deeper about legacy IS to design, implement and support gateways for more clients	• *Production managers and a selected team of highly skilled developers who mastered both legacy and new IS domains* launched new development projects to connect legacy and new systems by developing and deploying gateways • *Support teams* were setting deadlines on using gateways for clients • *Technical developers with support team* limited the functionalities of gateways (e.g., making them one-way connections to the legacy systems)
	Reallocating viable resources to new IS: The *resource complementarity* SRM is ceased as human and economic resources were reallocated from legacy to new systems	• *HR managers* integrated legacy IS teams into new IS teams to leverage their common, basic knowledge • *HR managers in collaboration with technical managers* defined transitional tasks for seasoned experts of legacy IS to gradually move to new IS teams
Interactions	**Learning to connect legacy and new ➜ reallocating from legacy to the new**	• *Convertors* transfer legacy *data* from legacy to new IS • *Gateways* allow that many *customers* start working with the new systems through and with the help of the legacy IS • Connecting legacy and new IS makes the two systems both support the historical routines, roles, functions (backward compatibility) which then allows organizations and their clients be able to *keep the same legacy of business processes and routines* and use them in the new systems
	Reallocating from legacy to the new ➜ learning to connect legacy and new	• Once the same users and same business processes, and same functions and roles were transferred to the new IS, it requires that legacy and new IS be more connected to bring all the related legacy data and features (more need to develop gateways and convertors)
Outcomes	• The legacy IS is connected to the new IS to transfer legacy data • Viable expertise and production tools are reallocated to the new IS	• Gateways are developed, tested, and implemented for clients to convert their legacy data • Old and new products are connected for the transition from the old to the new systems • Teams are retrained and transferred to new IS teams • The further production and development of legacy products are stopped

Table 10. Interactions and Feedback Relations Between Discontinuance Phases

	Theoretical Abstractions	Empirical Observations
Between Realization and Reversion	Discrediting legacy IS ⟷ relegitimizing legacy IS	(Present in Cases 2–4; due to commitments) • Knowing which aspects are old-obsolete helps relegitimizing the other aspects that are old-viable • Tension between discrediting some aspects, but not delegitimizing the entire legacy IS
	Learning to maintain legacy ➜ discrediting legacy	(Present in Cases 3 and 4; due to interdependencies) • Discovering new shortcomings and how fundamental are some of the limitations of the legacy IS and how costly to fix them
Between Reversion and Handover	Learning to maintain the legacy ➜ learning to connect legacy and new IS	(Present in Cases 3 and 4; due to interdependencies of legacy data) • Deeper knowledge enables creating gateways (more at the fundamental levels of data structure compatibility and the consistency with the different operating systems, and the capability to support some specific business processes)
	Learning to connect legacy and new IS ➜ learning to maintain the old	(Present in Cases 3 and 4; due to interdependencies of legacy data) • Keeping legacy IS next to the new working stimulates further need for maintenance of their interactions • Deeper learning for creating gateways enhances the capability to find ways to improve the legacy and thus maintain it
Between Handover and Marginalization	Reallocating from legacy to new ⟷ ceasing learning on the old	(Present in Cases 3 and 4; due to interdependencies of legacy data) • When the legacy data and functionalities is transferred, the support activities can be stopped • Reallocating the technical teams to new IS development and support naturally reduces their learning on the legacy domains
Between Marginalization and Reversion	De-routinizing ⟷ learning more about legacy IS to maintain	(Present in Cases 4; due to interdependencies and commitments) Failing to de-routinize legacy products brings back the need to learn more about the old systems: the customers keep asking as long as they use it in action When legacy IS is de-routinized, some occasional requests demands organizations to relearning to maintain the legacy

the legacy IS may and may not contribute to the progression of IS discontinuance. Sometimes ceasing a SRM *too early* can harm the discontinuance process.

Moreover, our process account highlights various *sequences* that are important for the progression of IS discontinuance. As illustrated in Case 3 (see Figure 3), the extensive technological interdependences required major efforts to hand over the viable data and functionalities from the old to the new system *before* the organization could engage in disrupting users' habits (Polites and Karahanna 2013) and motivate them to stop using legacy IS (Furneaux and Wade 2011). Otherwise, the viable legacy elements (e.g., viable data and functions) would not be properly transferred to the new IS, and the organization would revert back to the legacy system. This can explain why, despite attempts to motivate and trigger users to discontinue their use, legacy systems remain operative (Sahay et al. 2010). Focusing on delegitimizing IS choices and actions (Nicholson and Sahay 2009) or disrupting certain IS routines (Polites and Karahanna 2013) may partially and temporarily shake the IS path, yet may not result in breaking the entire path if other SRMs are not ceased in a timely manner.

Understanding Discontinuance as an Ongoing, Distributed Process

As illustrated in Cases 3 and 4, IS discontinuance turns into an *ongoing and distributed process* when the legacy systems straddle multiple organizations (e.g., the vendor company, the various providers who maintain the legacy systems, and the wide range of clients who use the legacy products and services), encompass various technological components (e.g., the base technology and product and production technologies) that are interconnected, and implicate a diverse range of routines that concern the development and usage of these systems.

This resembles today's typical IS landscape of established companies in various industries such as financial services, manufacturing, and petrochemical, where some systems have grown for decades by the gradual accumulation of additional layers. This entails that the latest technologies being implemented are dependent on technological components that were created decades ago, all working on the legacy of historical data that is distributed across multiple systems. In fact, todays' legacy systems are characterized through complex interconnection between many heterogeneous components (Polites and Karahanna 2012). Nowadays, organizations face complex, heterogeneous legacy systems that they constantly evolve (Kelty and Erickson 2015) as new technologies are used to modernize existing systems, new cloud applications are required to work with the legacy data, and agile-developed patches interconnect separate parts of the legacy landscape.

We showed that when legacy systems consist of multiple, interconnected components, some owned by the focal organization and some others outside, handover becomes a key phase, through which companies go back and forth between the old and new systems to gradually reduce their commitment on the old and develop the new path. In addition, the discontinuance process turns into an iterative and ongoing process, as if the process is never concluded.

This has two theoretical insights. First, in the case of distributed, heterogeneous legacy IS, it is likely that the discontinuance takes a long time, handover plays a central role, and we discover emergent interactions between the phases. Hence, although the very specific temporal patterns that we observed in the four cases may change from one case to another, the four phases and their interactions can help us understand the mechanisms driving the discontinuance process and which underlying SRMs can explain the evolution of the discontinuance process.

Second, we need to understand the discontinuance process more as an ongoing evolutionary process than a complete end-to-end one. Therefore, organizations may face *multiple* ongoing discontinuance processes, running simultaneously, yet each focusing on a specific part of the legacy landscape and going through a specific phase each time. In other words, when legacy systems comprise a heterogeneous and evolving ecology of systems, we need to consider an ongoing *system* of discontinuance processes.

Understanding How IS Discontinuance Advances Breaking the Organizational Path

Our findings also contribute to the explanation of breaking the established organizational path. Organizational path theory suggests that "the possibility of escaping from or breaking a path depends very much on interrupting the logic and the specific energy of the self-reinforcing patterns of the process in question" (Sydow et al. 2009, p. 702). Therefore, the theory explains and predicts that for disrupting an established IS path, its underlying SRMs need to cease. This is also reflected in the current IS literature that explains IS discontinuance through a set of *separate* actions that cease specific SRMs such as routinization (Polites and Karahanna 2013), learning (Wang et al. 2008), legitimization (Berente and Yoo 2012; Sahay et al. 2010), and resource compatibility (Flynn et al. 2009).

Although this focus on each and every specific SRM is important, it does not take into account *how ceasing one SRM influences ceasing other SRMs*. We observed that discontinuance involves phases through which ceasing one SRM supports ceasing other SRMs. Often interactions between discontinuance phases are pivotal, especially when legacy IS involves extensive interdependences and organizations have multiple commitments to it (e.g., Cases 2–4). In this way, we contribute to the short list of empirical insights regarding how the path-breaking process unfolds (Stache and Sydow 2014).

Nevertheless, we observed that to discontinue the legacy IS, we need to *temporarily intensify* some of its SRMs in order to cease other mechanisms. For instance, during handover, organizations need to intensify learning on the legacy IS to create gateways that allow the migration of legacy data (to cease resource complementarity SRM). Unlike the prediction of organizational path theory (Sydow et al. 2009), we show that breaking an established IS path is not a function of merely ceasing its SRMs, but also requires that certain SRMs be temporarily intensified. In fact, there are nonlinear, negative relations between the discontinuance actions: to cease some SRMs (e.g., resource complementarity), some other SRMs (e.g., such as learning) need to be temporarily intensified. Our cross-case analysis shows the prediction of current literature only holds under the conditions of limited organizational commitment and limited technological interdependences (see Case 1). Yet an established IS path often enjoys the condition of extensive organizational commitment and/or technological interdependences (see Cases 2–4). Under these conditions, focusing on only ceasing SRMs will not necessarily result in breaking the overall IS path and may even trigger premature and incomplete discontinuance.

Intensifying some SRMs to break the old IS path is different from intensifying the SRMs to develop the *new* path (Stache and Sydow 2014). Accumulating momentum on the new path (e.g., developing alternative systems) is necessary, but not sufficient to break an established IS path, since there should be mechanisms to reduce the momentum of the established IS path and transfer its viable elements to the new one. Otherwise, the risk is that both the old and the new IS coexist and

the dominance of the old compromises the development of the new.

Discontinuing the Old to Innovate the New

Our study underscores the importance of focusing on the old side in the innovation process. The literature on product and technology innovation has documented the hampering role of old routines (Polites and Karahanna 2013), dominant business logic (Bettis and Prahalad 1995), and aging products and services (Burgelman 1996), when they turn into core rigidities (Gilbert 2005; Leonard-Barton 1992). Known as the innovator's dilemma (Christenson 1997), the rigidities that stem from the attachment to legacy products and technologies prevent firms from remaining innovative. These studies suggest that firms need to actively sense and seize new opportunities, diversify their product and technological capabilities, and when needed, marginalize the old products and technologies (Christensen and Raynor 2003; Henderson 2006). For instance, the studies on de-escalation of IS projects suggest actions that concern the realization and marginalization of the existing system (Flynn et al. 2009; Montealegre and Keil 2000).

Our findings extend these suggestions by unpacking *how* the old technologies, routines, and products can be discontinued through an iterative process. In particular, we showed that both historical commitments and interdependences involved in legacy technologies and products require some phases such as revitalization and handover through which the old systems are temporarily developed, rather than marginalized. In fact, marginalizing the legacy products and technologies too early is risky, when legacy commitments still exist and when part of the legacy systems can be leveraged for nurturing the new systems. We showed that the interconnected nature of digital products and services implies that the commitments and interdependences of legacy IS often go beyond the internal technological choices, routines, and structures that an organization develops; but also involve the commitments that the organization accumulates through the products and services that are based on the legacy IS. Aging products developed on legacy systems often inherit a wide range of viable functions and historical data on the client's side and need to be transferred to the new systems and products.

Therefore, our study on the discontinuance of legacy IS from the vendor companies' perspective also sheds light on how organizations ditch their aging products. We showed that when the products and technologies are interdependent and involve legacy commitments, the old and new sides have complex relations. In fact, discontinuance is not a separate step before or parallel with innovating the new products and technologies; rather, it is an embedded and ongoing part of the very process of innovation. This context seems to be increasingly relevant due to the growing embeddedness of digital technologies into products and services, where old technologies and products are interconnected with new ones.

Insights for Managing the IS Discontinuance Process

Practitioner discourse has begun, but needs to further acknowledge the importance of IS discontinuance, especially when organizational culture focuses managers' attention mostly toward adopting new systems. Rapid technological changes can worsen this situation if companies incompletely marginalize their obsolete systems in their zeal to develop new ones, thus resulting in huge, complex legacy IS, recognized as "the ticking time-bomb" (Latham 2015).

Our findings suggest that in order for specific discontinuance attempts to succeed, organizations should not be satisfied with users' intentions and decisions to discontinue but rather they would need to go through an emergent, iterative process. The four phases and their interactions can be used as a process guide to show the range of potentially relevant actions, the challenges that need to be met at each moment, and the nuanced back-and-forth iterations that prevent premature discontinuance and reversion back to the old IS.

Discontinuing legacy IS requires extensive time and effort, especially when organizations need to work out their historical commitments, carefully reexamine the legacy IS interdependences, and mindfully transfer viable legacy elements (e.g., legacy data) to the new IS. Our cross-case analysis (see Table 10) provides insights into managers' decisions and actions regarding which phases and interactions can be crucial and what practices can be effective.

Boundary Conditions and Future Research

Some boundary conditions and future research must be acknowledged. Although our cross-case study allowed us to extract mechanisms regarding how the IS discontinuance process emerges, further empirical studies can enrich our findings by investigating the proposed mechanisms and extending them under other conditions. In particular, our empirical cases focused on the discontinuance of core enterprise systems from the perspective of vendors. In this sense, studying discontinuance from the perspective of user organizations that use multiple, interdependent systems from different vendors

can yield further insights. Furthermore, the proliferation of new IS sourcing models such as software-as-a-service may require different discontinuance dynamics and challenges. Although user organizations may be exempted from dealing with obsolete technologies in these new sourcing models, they still must deal with the obsolescence of their legacy organizational processes, structures, routines, and skills associated with old technologies. Likewise, the discontinuance process in those cases might reveal additional tensions between vendors and user organizations, especially because numerous users simultaneously share a particular technological environment in which the decisions and actions of discontinuance by one actor have immediate implications for others.

Finally, our focus on an old technology is relevant for various traditional industries such as manufacturing, banking, transportation, and national infrastructures where the core operations and services are still based on such old, historical systems. This provides a rich context in which complex interdependences and long-lasting historical commitments are more visible. Yet, future studies can explore the discontinuance of more modern types of legacy systems.

References

Alderson, A., and Shah, H. 1999. "Technical Opinion: Viewpoints on Legacy Systems," *Communications of the ACM* (42:3), pp. 115-116.

Argyris, C. 1977. "Double Loop Learning in Organizations," *Harvard Business Review* (55:5), pp. 115-125.

Arthur, W. B. 1989. "Competing Technologies, Increasing Returns, and Lock-in by Historical Events," *Economic Journal* (99:394), pp. 116-131.

Aula, P., and Mantere, S. 2013. "Making and Breaking Sense: An Inquiry into the Reputation Change," *Journal of Organizational Change Management* (26:2), pp. 340-352.

Bartels, B., Ermel, U., Sanborn, P., and Pecht, M. G. 2012. *Strategies to the Prediction, Mitigation, and Management of Product Obsolescence*, Upper Saddle River, NJ: John Wiley & Sons.

Battilana, J., and D'Aunno, T. 2009. "Institutional Work and the Paradox of Embedded Agency," in *Institutional Work: Actors and Agency in Institutional Studies of Organization,* T. B. Lawrence, R. Suddaby, and B. Leca (eds.), Cambridge, UK: Cambridge University Press, pp. 31-58.

Berente, N., and Yoo, Y. 2012. "Institutional Contradictions and Loose Coupling: Postimplementation of NASA's Enterprise Information System," *Information Systems Research* (23:2), pp. 376-396.

Bettis, R. A., and Prahalad, C. K. 1995. "The Dominant Logic: Retrospective and Extension," *Strategic Management Journal* (16:1), pp. 5-14.

Bhattacherjee, A. 2001. "Understanding Information Systems Continuance: An Expectation–Confirmation Model," *MIS Quarterly* (25:3), pp. 351-370.

Brook, C., Pedler, M., Abbott, C., and Burgoyne, J. 2015. "On Stopping Doing Those Things That Are Not Getting Us to Where We Want to Be: Unlearning, Wicked Problems and Critical Action Learning," *Human Relations* (69:2), pp. 1-21.

Burgelman, R. A. 1996. "A Process Model of Strategic Business Exit: Implications for an Evolutionary Perspective on Strategy," *Strategic Management Journal* (17:S1), pp. 193-214.

Butler, B. S., and Gray, P. H. 2006. "Reliability, Mindfulness, and Information Systems," *MIS Quarterly* (30:2), pp. 211-224.

Christensen, C., and Raynor, M. 2003. *The Innovator's Solution: Creating and Sustaining Successful Growth,* Boston: Harvard Business School Press.

Christenson, C. 1997. *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail*, Boston: Harvard Business School Press.

Flynn, D., Pan, G., Keil, M., and Mahring, M. 2009. "De-Escalating IT Projects: The DMM Model," *Communications of the ACM* (52:10), pp. 131-134.

Furneaux, B., and Wade, M. 2011. "An Exploration of Organizational Level Information Systems Discontinuance Intentions," *MIS Quarterly* (53:3), pp. 573-598.

Garud, R., Dunbar, R. L. M., and Bartel, C. A. 2011. "Dealing with Unusual Experiences: A Narrative Perspective on Organizational Learning," *Organization Science* (22:3), pp. 587-601.

Garud, R., Kumaraswamy, A., and Karnøe, P. 2010. "Path Dependence or Path Creation?," *Journal of Management Studies* (47:4), pp. 760-774.

Gibbert, M., Ruigrok, W., and Wicki, B. 2008. "What Passes as a Rigorous Case Study?," *Strategic Management Journal* (29:13), pp. 1465-1474.

Gilbert, C. G. 2005. "Unbundling the Structure of Inertia: Resource Versus Routine Rigidity," *The Academy of Management Journal* (48:5), pp. 741-763.

Hainaut, J.-L., Cleve, A., Henrard, J., and Hick, J.-M. 2008. "Migration of Legacy Information Systems," in *Software Evolution*, T. Mens and S. Demeyer (eds.), Berlin: Springer, pp. 105-138.

Hedberg, B. 1981. "How Organizations Learn and Unlearn," in *Handbook of Organizational Design, Volume 1,* P. C. Nystrom and W. H. Starbuck (eds.), New York: Oxford University Press, pp. 3-27.

Henderson, R. 2006. "The Innovator's Dilemma as a Problem of Organizational Competence," *Journal of Product Innovation Management* (23:1), pp. 5-11.

Keil, M. 1995. "Pulling the Plug: Software Project Management and the Problem of Project Escalation," *MIS Quarterly* (19:4), pp. 421-447.

Kelty, C., and Erickson, S. 2015. "The Durability of Software," *There Is No Software, There Are Only Services*, I. Kaldrack, and M. Leeker (eds.), Lüneburg, Germany: Meson Press, pp. 39-56.

Langley, A. 1999. "Strategies for Theorizing from Process Data," *Academy of Management Review* (24:4), pp. 691-710.

Latham, M. 2015. "The Ticking Time-Bomb at the Heart of Our Big Banks' Computer Systems," *The Herald (Scotland)*, July 11 (https://www.heraldscotland.com/business_hq/13416691.display/).

Lawrence, T. B., and Maitlis, S. 2005. "The Disruption of Accounts: Sensebreaking in Organizations," paper presented at the Academy of Management Annual Meeting, Honolulu, Hawaii.

Leonard-Barton, D. 1992. "Core Capabilities and Core Rigidities: A Paradox in Managing New Product Development," *Strategic Management Journal* (13:SI), pp. 111-125.

Lyytinen, K., and Newman, M. 2008. "Explaining Information Systems Change: A Punctuated Socio-Technical Change Model," *European Journal of Information Systems* (17:6), pp. 589-613.

Mähring, M., Keil, M., Mathiassen, L., and Pries-Heje, J. 2008. "Making IT Project De-Escalation Happen: An Exploration into Key Roles," *Journal of the AIS* (9:8), pp. 462-496.

Maruyama, M. 1963. "The Second Cybernetics: Deviation-Amplifying Mutual Causal Processes," *American Scientist* (51:2), pp. 164-179.

MeriTalk. 2013. "Innovation Inspiration: Can Software Save IT?," MeriTalk, Alexandria, VA.

Montealegre, R., and Keil, M. 2000. "De-Escalating Information Technology Projects: Lessons from the Denver International Airport," *MIS Quarterly* (24:3), pp. 417-447.

Nicholson, B., and Sahay, S. 2009. "Deinstitutionalization in the Context of Software Exports Policymaking in Costa Rica," *Journal of Information Technology* (24:4), pp. 332-342.

Oliver, C. 1992. "The Antecedents of Deinstitutionalization," *Organization Studies* (13:4), pp. 563-588.

Parthasarathy, M., and Bhattacherjee, A. 1998. "Understanding Post-Adoption Behavior in the Context of Online Services," *Information Systems Research* (9:4), pp. 362-379.

Polanyi, M. 1966. *The Tacit Dimension*, London: Routledge and Kegan Paul Ltd.

Polites, G. L., and Karahanna, E. 2012. "Shackled to the Status Quo: The Inhibiting Effects of Incumbent System Habit, Switching Costs, and Inertia on New System Acceptance," *MIS Quarterly* (36:1), pp. 21-42.

Polites, G. L., and Karahanna, E. 2013. "The Embeddedness of Information Systems Habits in Organizational and Individual Level Routines: Development and Disruption," *MIS Quarterly* (37:1), pp. 221-246.

Robey, D., Ross, J. W., and Boudreau, M.-C. 2002. "Learning to Implement Enterprise Systems: An Exploratory Study of the Dialectics of Change," *Journal of Management Information Systems* (19:1), pp. 17-46.

Sahay, S., Sæbø, J. I., Mekonnen, S. M., and Gizaw, A. A. 2010. "Interplay of Institutional Logics and Implications for Deinstitutionalization: Case Study of HMIS Implementation in Tajikistan," *Information Technologies & International Development* (6:3), pp. 19-32.

Schreyögg, G., and Sydow, J. 2011. "Organizational Path Dependence: A Process View," *Organization Studies* (32:3), pp. 321-335.

Scott, W. R. 2001. *Institutions and Organizations* (2nd ed), Thousand Oaks, CA: SAGE Publications.

Simon, H. A. 1977. *The New Science of Management Decision*, Upper Saddle River, NJ: Prentice Hall.

Singh, R., Mathiassen, L., and Mishra, A. 2015. "Organizational Path Constitution in Technological Innovation: Evidence from Rural Telehealth," *MIS Quarterly* (39:3), pp. 643-665.

Slee, C., and Slovin, M. 1997. "Legacy Asset Management," *Information Systems Management* (14:1), pp. 12-21.

Stache, F., and Sydow, J. 2014. "Path-Breaking Organizational Change: Effective Cancer Treatment through Multi-Center Cooperation," in *Academy of Management Proceedings*, p. 10567.

Sydow, J., Schreyögg, G., and Koch, J. 2009. "Organizational Path Dependence: Opening the Black Box," *The Academy of Management Review* (34:4), pp. 689-709.

Sydow, J., Windeler, A., Schubert, C., and Möllering, G. 2012. "Organizing R&D Consortia for Path Creation and Extension: The Case of Semiconductor Manufacturing Technologies," *Organization Studies* (33:7), pp. 907-936.

Tsoukas, H. 2009. "Craving for Generality and Small-N Studies: A Wittgensteinian Approach Towards the Epistemology of the Particular in Organization and Management Studies," in *The Sage Handbook of Organizational Research Methods,* D. Buchanan and A. Bryman (eds.). London: SAGE Publications, pp. 285-301.

Vergne, J.-P., and Durand, R. 2011. "The Path of Most Persistence: An Evolutionary Perspective on Path Dependence and Dynamic Capabilities," *Organization Studies* (32:3), pp. 365-382.

Vlaar, P. W. L., van Fenema, P. C., and Tiwari, V. 2008. "Cocreating Understanding and Value in Distributed Work: How Members of Onsite and Offshore Vendor Teams Give, Make, Demand, and Break Sense," *MIS Quarterly* (32:2), pp. 227-255.

Wang, C. L., Ahmed, P. K., and Rafiq, M. 2008. "Knowledge Management Orientation: Construct Development and Empirical Validation," *European Journal of Information Systems* (17:3), pp. 219-235.

Wagner, E. L., Moll, J., and Newell, S. 2011. "Accounting Logics, Reconfiguration of ERP Systems and the Emergence of New Accounting Practices: A Sociomaterial Perspective," *Management Accounting Research* (22:3), pp. 181-197.

Whittle, A., Mueller, F., Gilchrist, A., and Lenney, P. 2016. "Sensemaking, Sense-Censoring and Strategic Inaction: The Discursive Enactment of Power and Politics in a Multinational Corporation," *Organization Studies* (37:9), pp. 1323-1351.

Yin, R. K. 2002. *Case Study Research: Design and Methods*, Newbury Park, CA: SAGE Publications.

About the Authors

Mohammad Hosein Rezazade Mehrizi is an assistant professor in the Knowledge, Information and Networks (KIN) Research Group at VU University Amsterdam. He received his first Ph.D. from Sharif University of Technology, Tehran, Iran, and his second Ph.D. from ESADE Business School, Spain. His interest involves understanding and helping organizations and their associated communities unlock the historical chains that prevent them from making a better future. He is researching how organizations unlearn their outmoded experiences, discontinue their obsolete information systems, learn and unlearn from their incidents, and renovate their established business models to tap the opportunities of new technologies such as big-data analytics.

Joan Rodon Modol is an associate professor in the Information Systems Department at ESADE Business School, Universitat Ramon Llull. His research focuses mainly on the processes of emergence and evolution of digital platforms and infrastructures. His research has been published in journals such as *Journal of the AIS, European Journal of Information Systems, Communications of the ACM, Journal of Information Technology, Information Systems Journal,* and *Decision Support Systems*.

Milad Zafar Nezhad is a Ph.D. candidate of Industrial and Systems Engineering at Wayne State University, Detroit, Michigan. He received a Bachelor's degree in Industrial Engineering from Sharif University of Technology, Tehran, Iran, a Master's degree in Industrial Engineering from AmirKabhir University of Technology, Tehran, and a second Master's degree in Computer Science Wayne State University. His major research interests include data analytics, machine learning, healthcare informatics, and management information systems.

MIS Quarterly

RESEARCH ARTICLE

INTENSIFYING TO CEASE: UNPACKING THE PROCESS OF INFORMATION SYSTEMS DISCONTINUANCE

Mohammad Hosein Rezazade Mehrizi
School of Business and Economics, Vrije University Amsterdam, De Boelelaan 1105,
1081 HV Amsterdam, NETHERLANDS {m.rezazademehrizi@vu.nl}

Joan Rodon Modol
Universitat Ramon Llull, ESADE Business School,
08172 Sant Cugat del Vallès, SPAIN {joan.rodon@esade.edu}

Milad Zafar Nezhad
Industrial and Systems Engineering, Wayne State University,
Detroit, MI 48202 U.S.A. {m.zafarnezhad@wayne.edu}

Appendix A

Coding for the Discontinuance Phases and Their Interactions Across the Cases

Table A1. Realization

Realization	SmallCo	MedCo	LargStrict	LargLenient
Triggers • Acquaintance with new IS • Technological changes • New requests by leading clients	*(Started in early 1993)* The technical designers and programmers and the CEO actively observed the technological changes and saw the trend of MS-Windows becoming the next technology generation	(*Mainly became a hot discussion in 1995*) The acquaintance of a few fresh programmers with the MS-Windows The strategic decision to abandon many old products and only focus on office automation systems	(Mainly started in 1993) The limitations of MS-DOS relative to Windows for producing integrated set of organizational systems and growing demand from their large client base	*(Mainly started in 1993)* The recognition of the limitations of MS-DOS relative to Windows, and their strategy to focus on state of the art technologies
Actions and the roles of embedded actors **Scrutinizing old IS limitations (ceasing indwelling SRM)** • Technical teams and middle managers and later top management and sales and marketing made sense of the limitations of old systems relate to new ones • Technical development teams were searching for the comparable	*(For the entire year 1993 and before March 1994)* Examining the limitations of MS-DOS in terms of graphical functionalities, parallel processing programming; scrutinizing mainly happened to help the valuation of old product for selling them out	*(Frequently observed during 1995 and 1996; and later as an important set of actions until 1999)* It took two years of discussion about the tools, techniques, units, manuals, and expertise developed around MS-DOS to exactly see which features were problematic and which	*(Intensively for the years 1993-1994; and later as occasional actions during 1995-1999 when major limitations of MS-DOS systems was detected or when addressing a major change request was difficult due to basic limitations of MS-DOS)* Scrutinizing MS-DOS	*(From 1993-1994 as the intensive period of realization, yet still often surging in the discussions until 2002)* Extensively done by R&D team and the production team and required formal R&D projects due to the huge investment in developing various technical tools related to

advantages and capabilities of the old technologies and related products relative to the new ones • Senior experts highly specialized in the old systems were posing novel questions about old systems		ones were still viable and should be kept in the new office automation systems	aspects was extensively done with regard to the implications of moving toward more integrated systems as well as the routines and organizational procedures for designing, producing, and supporting isolated systems	MS-DOS and huge number of interconnected products with sophisticated DOS-related technologies
Discrediting the viability of (parts of) old IS (ceasing legitimization SRM) • Product managers in discussion with technical teams were distinguishing between viable and obsolete aspects of old systems and discussing the boundaries between them • Forward jumpers were discrediting old systems by highlighting the new technological opportunities and deficiencies of old systems • Middle managers had to constantly convince forward jumpers to not disregard the entire old systems as obsolete • Marketing managers and agents were carefully decoupling between internal and external legitimacy of the old systems	Most of the managers (except the financial manager) and technical team (except two seasoned programmers) already considered the old systems as obsolete	Extensively engaging a wide range of managers and technical experts with general tendency to consider many aspects as viable at the beginning, especially the aspects that had been improved through further development of the systems	Discrediting MS-DOS extensively happened at the level of basic product design, as well as the organizational procedures and routines for designing, producing, and supporting products	Initially (during the first half of 1993) it was only done for few aspects of MS-DOS (e.g., parallel processing); yet later it required more effort by the R&D manager due to the dominance of seasoned MS-DOS experts
Scrutinizing ➜ delegitimizing • Paying attention to the detailed, technical characteristics and examining various potential technological features that appeared to be limiting the capacities and functions of the old IS • Reopening the list of shortcomings of the technology that were worked around and adjusted locally in past upgrades and maintenance	*(Through free discussions at the end of the weeks during 1993)* A list of MS-DOS shortcomings was created in order to convince the financial office and their partners that MS-DOS had no future	Scrutinizing was limited to only those aspects that could not be easily improved based on their deep expertise in MS-DOS technology	To expand the list of problematic elements to design, testing, and production techniques and routines	Often happened as systematic tasks to reflect on the limitations of MS-DOS even though many of the shortcomings of the MS-DOS was improved by the technical team
Delegitimization ➜ scrutinizing • The discussion about the boundaries between old-viable and old-obsolete required inspecting some detailed technological features and aspects in more details (to see which aspect is really the problematic aspect)	CEO asked in middle 1993 the head of technical team to create a list of limitations of the product to distinguish between what was problematic about MS-DOS and what were the things that needed to be kept in the new product *(an ongoing task during year 1993)*	It required a lot of effort to articulate detailed reasons for the fundamental limitation of MS-DOS; Technical Manager always stressed: "*let's be careful not to through away the good apples*"	To unpack the range of interdependencies between data, codes, and designs that might not work properly in integrated design	It became relevant when some new features of MS-DOS were further developed, e.g., the possibility of working with graphical peripheral devices such as laser printers and high-resolution monitors
Contextual conditions shaping realization • The longer and deeper experience and specialization in old IS ➜ required more time and efforts to reopen the taken-for-granted details of old IS and carefully distinguish viable and obsolete aspects • The larger number of interconnected technological elements and products related to old IS (relevant in LargLenient and LargStrict) ➜ required more extensive discussions and examinations for distinguishing between old-viable and old-obsolete aspects of old IS • The dominance of forward jumpers with	Small number of clients (10 clients) Their long experience with MS-DOS and up-to-date technical managers and employees made them have the feeling "we are done with MS-DOS"! A simple, stand-alone product that facilitated its sale	Highly specialized skills and technological elements related to MS-DOS systems, with a few seasoned experts on MS-DOS challenged the realization, yet the small number of products facilitated the realization that the products should be sooner or later abandoned	The large customer base and still growing demand for extending the support contracts created tensions between the marketing and production departments regarding the obsolescence of MS-DOS based products	Large number of clients and high interconnectivity of MS-DOS based product, which made it difficult to easily realize that MS-DOS was becoming obsolete; yet the technological strategy of the firm to work on state-of-the art technology facilitated realization

the culture of appeal to new technology, in comparison with seasoned specialized experts on old IS ➜ supporting the de-legitimization of old IS, though increasing the risk of immaturely de-legitimizing the entire old IS path too early				

Table A2. Marginalization

Marginalization	SmallCo	MedCo	LargStrict	LargLenient
Triggers • The fear that the old systems might never be discontinued although the new systems were developed • Realizing the high cost of keeping and maintaining the old systems and especially the gateways (cutting gateways) • The usage of the new systems in a stable way (initial bugs and problems were solved and it was routinized)	(In early 1994), The decision and agreement to sell the document management product and all the support contracts to the financial officer to start a new business	The development of the new Windows-based office automation as the core product to focus on, in late 2000 and early 2001	*(Early 2002)* when the new MS-Windows products became stable and passed various tests	*(Started in 2002 for some clients)* The usage of new systems in a stable way and the huge cost of maintaining MS-DOS integrated systems stimulated a gradual marginalization of MS-DOS systems
Actions and the roles of embedded actors **Ceasing learning about old IS (ceasing learning SRM)** • Technical and HR managers gradually stopped R&D projects, training, and the hiring on the domains specific to old systems • IS Product managers in negotiation with support teams were carefully distinguishing between 'major' and 'minor' changes into the systems based on their learning commitments • Production and HR managers selectively isolated old and new systems teams and located them in distinct organizational units • R&D and HR managers proactively prevented narrow specialization in domains that might become obsolete quickly	Naturally happened due to the limited client base and limited requests for update and maintenance; but was sometimes suddenly reversed when an influential client urged the company to fix a major bug or add a major feature in the old products	Mainly by short-term isolation of the MS-DOS team from the new MS-Windows team during 2000 and 2001 when the new product was designed, and gradually outflow of the MS-DOS experts during 2002 and 2003	*(Mainly started in mid 2002 and mostly done through 2003)* Often by strict limits on not accepting (major) change requests and stopping support contracts and not extending them	*(Officially started in 2002, but continued as an ongoing process until 2005 and even later until 2009)* For each set of clients who were ready to replace the MS-DOS products, marginalization was executed with iterations back with handover (to customize the gateways for them) Often started by stopping formal learning actions such as official trainings and formal R&D projects and gradually extending to informal learning activities such as learning through support and debugging the old systems
Action: De-routinizing old IS (ceasing routinization SRM) • Marketing and Support managers specified clear deadlines for terminating the support of old systems • Support teams paralyzed the old systems in order not to be used by clients that already adopted new systems • HR managers helped some seasoned experts to move to other companies where they could find a relevant position • Production managers and support teams (when recommending to clients) removed the technological objects (e.g., operating systems, programming tools, and test tools) to prevent working on the old systems • Top managers (often CEOs) supported the teams of seasoned experts who were not willing to move to new domains to create their spin-offs • Production managers and the technical team of old IS carefully codified the old systems to allow them move on to new systems teams and for future occasional requests	Naturally and smoothly happened since the few employees were almost all technical experts, enthusiastic in working with MS-Windows; the two seasoned MS-DOD experts moved to the new company created by the financial officer	*(Gradually during 2002 and 2003; but still ongoing for the following five years)* Mainly by reducing the workload on the MS-DOS by stopping the support contracts and selling out the old products	Often by planned interventions to stop clients' access to the old systems and formally banning the internal team to work on the DOS systems (and asking for documenting the important aspects of the DOS systems in case of some urgent future requests) Actively creating spinoffs by MS-DOS experts and redirecting support requests to them	In a gradual process and often with the collaboration of each client for making it less accessible to use the old systems and assigning less work on the upgrading the old systems to the technical team

Ceasing learning ➜ de-routinization • When systems were not developed further, the clients have lower chance to keep using them (terminating support contracts)	*(During 1994 and 1995)* The CEO rejected clients' requests for supporting the MS-DOS systems by arguing that SmallCo no longer had the capabilities to do so	By selling out the old products and outsourcing the support contracts mainly in 2003	Extensively and often strongly forcing the termination of support contracts and rejecting the major changes	Often did not happen immediately, given the bargaining power of clients to ask for changes to the old systems
De-routinization ➜ ceasing learning • The fewer users and clients used the legacy system, the less the need for support and updating activities (thus less learning to do so)	*(Mainly in 1994, and less in 1995)* CEO and the head of technical team were constantly asking their programmers not to fix the bugs related to the old systems; yet this was once every two or three months interrupted due to the major support requests of clients	Having less major improvements on the old systems during 2002 and 2003 and discussing with their clients to use their new product instead of having the old products improved	Often happened for a large part of clients by rejecting their major change requests	Often effective for junior programmers and support team For senior experts, often happened in a gradual way with a lot of back and forth
Contextual conditions shaping realization • The relatively low bargaining power of companies against their influential clients (in all cases except LargStrict) ➜ forced the companies to come back occasionally to the old systems to relearn about the old systems in order to apply major change requests • Deep specialization ➜ made it hard for seasoned experts to stop relying on their deeply rooted expertise • The scope and interdependency of legacy systems (mainly in LargLenient and LargStrict) ➜ posed serious challenges on dissolving legacy resources; and made it difficult to move clients who have been using many inter-connected products on the old systems	Limited number of clients facilitated marginalizing the legacy IS The fact some of the experienced MS-DOS programmers joined the partner company also helped redirecting support requests to them	Low bargaining power against the clients and deep specialization in MS-DOS slowed down marginalization; despite limited number of commitments in terms of the clients and product diversity	The strong bargaining power vis-à-vis their clients and the isolated systems facilitated marginalizing MS-DOS systems Having and forming a strong network of spinoff companies helped redirect support requests to them	The strategy of admitting clients' requests and deep specialization in MS-DOS and the large amount of technical interdependencies between the old products took a lot of time and effort to marginalize the old IS

Table A3. Reversion

Reversion	SmallCo	MedCo	LargStrict	LargLenient
Triggers • Ongoing requests from the clients on the old system • Lack of readiness and instability of the new system • Risk of too early departure from incumbent systems	It almost never happened since the old systems were sold to partners and the new product was completely different from the old one; very occasional requests of the legacy clients that the partner company could not handle (almost a couple of times in the years between 1993 and 1995)	*(Mainly in 1996 and continued until 1999)* The deep, specialized technical capabilities triggered some efforts to keep MD-DOS and several of its related tools	*(Started in 1995 but was completed in 2000)* Long delay in developing the new integrated products and making them stable and some problems in the new products kept many clients asking for reverting back to MS-DOS	*(Mainly started in 1994)* Long delay (more than ten years) to develop new alternative systems and the ongoing demand for the MS-DOS systems triggered an extensive, long reversion
Actions and the roles of embedded actors • Re-legitimizing old IS (intensifying legitimization SRM) • Product managers were promoting old IS as still viable solutions in short-term • Product managers and marketing agents were constantly highlighting the relative advantages of old systems against the immature versions of new technologies		*(As an ongoing attempt during 1996-1999)* Often focused on internal attempts by senior MS-DOS experts who were arguing that many of the DOS-related technological capabilities were viable; the Technical Manager had to carefully ensure that even internal discussion regarding the problems of MS-DOS did not affect their credibility for clients who were still actively using the legacy products	*(For the entire period of 1995-2000 intensively and then as one of the side-production lines during 2000-2002 for urgent change requests)* A major effort to convince technical teams to keep the development and upgrade of the old systems especially when some other teams were working on the new systems (a feeling of being left behind)	*(Started mainly in early 1995 but continued actively until 2002 as a formal department and later as additional projects and later occasional projects until 2005)* Seriously pursued by production manager and support manager internally and by marketing team externally
Learning more about the old IS to maintain it (intensifying learning SRM) • Seasoned, specialized experts were actively identifying the shortcomings of old systems and improve them in an effective way • Senior experts in old IS rediscovered several untapped potentials of old systems and developed them • Product managers and development teams launched new development projects to improve the functionalities of old systems		Mainly through update and support activities to improve the old systems and therefore keep clients satisfied; Technical Manager had to motivate the young programmers to still see working on MS-DOS as an important job	Often through a selected list of change requests after being approved by the production manager to exclude major design changes and limit to really urgent requests	Often in the form of R&D projects on the MS-DOS and linking them with the major improvements in the old products
Re-legitimizing ➜ learning to improve • Knowing which aspects of the old IS were considered as viable and thus could and should be further improved meanwhile		*(Mainly from 1996-1999, for every major request from clients and every month for the small change requests)* Through the approval of most of change and support requests by the technical manager and the head of support team	Reapproving some of the change requests on the DOS systems that were initially rejected (and later accepted because the alternative systems were not yet ready and stable)	Approving many clients' major requests to update DOS-based products
Learning to improve ➜ re-legitimizing • The improvement of old IS turn it into a viable solution to be presented to the clients		*(Mainly from 1996-1999, often when a major limitation was discovered that could not be easily addressed in the current systems)* To highlight the relevant basic aspects of the product design and production and support routines	Often in a careful way to minimize major changes in old products	Through extending the support contracts and promoting the internal MS-DOS technical team by the managers
Contextual conditions shaping	Given the small	Being open to clients'	Large number of clients	Extensive client base and

realization • Legacy commitments to old IS (for LargLenient and MedCo) and the dominance of public, large clients (for MedCo and SmallCo) and increasing demand for support contracts ➜ forced the companies to keep learning on the old systems and improve them by dedicating teams to learn on and develop the old systems • The larger number of interconnected technological elements and products related to old IS (relevant in LargLenient and LargStrict) ➜ heightened the reversion since improving one part requires subsequent improvement in other parts • Deeper specialization in old IS (mainly in LargLenient and LargStrict) ➜ enabled the companies to dedicate further resources and capabilities to improve old systems	client base, the fact that there was only a separate product developed on MS-DOS, the reversion was not relevant; Some seasoned DOS experts helped making local, occasional improvements by moving to the other partner company that bought the old product and the support contracts	requests required extensive reversion for around four years Deep specialization on MS-DOS helped reversion to the MS-DOS and the related technological capabilities; especially by seasoned experts	using the multiple legacy systems and the fact that the new systems were not stable required considerable reversion efforts; yet it was challenging due to the dominance of the forward jumpers	large number of interconnected products combined with the strategy of the company to address the support requests of the clients resulted in extensive reversion activities; deep specialization on MS-DOS provided enough capability for reversion

Table A4. Handover

Handover	SmallCo	MedCo	LargLenient	LargStrict
Triggers • The development of the new systems and making sure they are stable • The shrinking market on the old systems (especially for the cases that they had major delay in developing new systems in comparison to the other competitors)	Given that the old product was completely sold out and the new document management system was completely different in terms of the design and technology, handover was very limited; using design ideas and transferring the brand (e.g., similar name of the product); for the clients who used the old product, the data entry to the new systems was done anew	Since the new product was a completely different product and did not use many of the aspects of the old products, it required very limited migration of legacy data for many clients who were using completely different products; in many cases, the clients (often with the help of MedCo) started entering new data to their systems	*(Started during 2000 and gradually expanded until the end of 2002)* The development of Windows-based systems in which the basic design of the products were also significantly improved comparing with the MS-DOS	*(Mainly at the end of 2001 for the core products and until the end of 2002 for other peripheral products)* When new systems became stable
Actions and the roles of embedded actors • Learning more about old IS to connect old and new IS (intensifying learning SRM) • Production managers and a selected team of highly skilled developers who mastered both old and new IS domains launched new development projects to connect old and new systems by developing and deploying gateways • Support teams were setting deadlines on using gateways for clients • Technical developers at the support team limited the functionalities of gateways (e.g., making them one-way ; that is, from the old to the new system)			*(From 2000-2002 as a formal line of production, later as extension projects until 2005 and afterward as occasional tasks when a major change to old products was done)* Extensively engaged in creating ways of converting the data from MS-DOS to MS-Windows especially because the basic product designs were changed (gateways created a major line of R&D and production projects)	*(During 2000-2002 in parallel with designing the new systems)* Mainly for converting the legacy data from systems to new integrated database structures (often for large clients); highly challenging because of the many interdependent products, data elements, and business processes that had to be integrated in the new systems and still support legacy data and functions
Reallocating viable resources from old to new IS path (ceasing resource complementarity SRM) • HR managers integrated old IS teams into new IS teams to leverage their common, basic knowledge • HR managers in collaboration with technical managers defined transitional tasks for seasoned experts of old IS to gradually move to new IS teams			Often the designers and support team were reallocated, but less programmers; transferring legacy data and the functionalities of the old systems became a major technical challenge; tendency to abandon many technological tools	Mainly focused on retraining the technical team and support teams to be able develop and support the new systems (less concern about the transfer of technological components and configurations since the entire product design was changed)
Learning to connect old and new ➜ reallocating from old to the new • Convertors allow for transferring legacy data from old to new IS • Gateways allow that many customers start working with the new systems through and with the help of the old IS (e.g., using old IS legacy data but through the applications of the new IS) • Connecting old and new IS makes the two systems both support the historical routines, roles, functions (backward compatibility) which then allows organizations and their clients be able to keep the same legacy of business processes and routines and use them in the new systems (thus for them not everything should be changed)			Often was extensively done for transferring data; also happened for transferring the configurations and settings between the various products in the entire integrated system (e.g., the compatibility between HR and finance modules)	Extensive projects and teams for developing gateways (in 2001 it became comparable with the projects dedicated to design new systems)
Reallocating from old to the new ➜ learning to connect old and new • Once the same users and same business processes, and same functions and roles were transferred to the new IS, it requires that old and new IS be more connected to bring all the related legacy data and features (more need to develop gateways and convertors)			Through creating dedicated R&D projects and teams to focus on learning more about MS-DOS for improving the performance of gateways	Dedicated teams for ensuring the compatibility of old and new system

Contextual conditions shaping realization • The technological differences between old and new systems posed challenges for connecting them via gateways, and created a knowledge gap between old and new IS teams, which challenged their migration to new team • The business process continuity in clients' side and the instability of the new systems (in all cases) increased the period during which gateways had to be deployed • The scope and interdependency of legacy systems (mainly in LargLenient and LargStrict) required the development of complex gateways in order to deal with many interconnections between old and new IS paths	The completely different product design in the new systems to comply with backward compatibility; limited commitment to clients to support and transfer their data	The completely different product design in the new systems to comply with backward compatibility	Because of the major technological differences between MS-DOS and Windows-based products, the extensive interdependencies between different products and technological components, and the need to guarantee business process continuity, the handover involved a lot of effort and expertise	The big difference between the design and production of products in MS-DOS and integrated systems in Windows and the many interdependencies in the legacy products in terms of the various data elements and business processes

Table A5. Iterations Between the Phases

Interactions Between the Phases	SmallCo	MedCo	LargStrict	LargLenient
Between Realization and Reversion **Discrediting old IS ⟷ re-legitimizing old IS** • Knowing which aspects are old-obsolete helps relegitimizing the other aspects that are old-viable • Tension between discrediting some aspects, but not delegitimizing the entire legacy IS **Learning to maintain the old ➜ discrediting old IS** • Discovering new shortcomings and realizing how fundamental are some of the limitations of the old IS and how costly to fix them	Not observed	A long set of back and forth discussions over around two years (1996-1998) to internally discredit MS-DOS but externally still keep it as a viable solution for their clients	*(For two years since 1993 as a continues discussion among different groups of designers, programmers, and gradually in late 1994 among the middle and top management)* Major tensions regarding the prestige of working on new systems and the feeling of being discriminated if asked to work on the old systems; by realizing the cost and efforts needed for making such change and improvement in the old IS	*(For around four years from 1994 to 2000, iteratively)* A lot of discussions due to the complexity of the products and their interdependencies Often happened at the very late phases that the cost of marinating MS-DOS systems appeared to become too much
Between Reversion and Handover **Learning to maintain the old ➜ learning to connect old and new** • Deeper knowledge that requires creating gateways (more at the fundamental levels of data structure compatibility and the consistency with the different operating systems, and the capability to support some specific business processes) **Learning to connect old and new ➜ learning to maintain the old** • Keeping legacy IS next to the new working and thus the need for maintenance continues • Deeper learning for creating gateways enhances the capability to find ways to improve the legacy and thus maintain it	Not observed	Not relevant since the new MS-Windows system was complete different from the old system and the new technical team was newly hired	*(Intensively happening during 2001 and 2002 in parallel with designing the new products)* To ensure that the changes to the legacy products are considered in (re)designing and implementing gateways and the new systems The complexity of the many interdependent products often required a lot of efforts to systematically examine the parts that can be affected by a change in the legacy IS and therefore be considered in designing gateways and new products	*(Frequently during 2000-2002 in parallel with redesigning new products)* Through formal, dedicated R&D projects on gateways and appointing the senior MS-DOS experts to them The senior MS-DOS experts acting as brokers between the gateways teams and MS-DOS support teams
Between Handover and Marginalization **Reallocating from old to new ➜ ceasing learning on the old** • Especially when the legacy data and functionalities is transferred, then the support activities can be stopped • Reallocating the technical teams to new IS development and support naturally reduces their learning on the legacy domains	Not observed	Not relevant since the new MS-Windows system was complete different from the old system and the new technical team was newly hired	*(During 2012 and 2013)* More as one-directional force for the production and support team by formally banning their engagement with the old products	*(Iteratively done from 2002 until 2005 for most of the clients and later for clients who were late in replacing MS-DOS)* Often by reallocating the production and R&D experts to work on the MS-Windows and for the clients when their legacy data and systems' configurations were transferred to the new systems; to customize gateways for clients
Between Marginalization and Reversion **De-routinizing ⟷ learning more about old IS to maintain** • Failing to de-routinize required learn further about the old systems since the customers kept asking changes as long as they used it • Even when legacy IS was de-routinized, some occasional requests will take back and requires relearning to maintain the legacy	Not observed	Several support requests that most of them were handled by MedCo but gradually some of them were outsourced to other companies after 2001	Limitedly observed after 2002; Only occasionally happened for very urgent problems requested by large companies	*(Iteratively, for each set of major change requests from 2002 until 2005 and later as occasional projects and tasks until 2009)* Frequently happening due to the extensive use of MS-DOS systems by large number of influential clients

Appendix B

Validity and Transparency Considerations

Table B1. Strategies for Enhancing the Validity and Transparency

Validity Measure	Strategies for Addressing
Construct validity	• Relying on a theoretically informed conceptual framework based on unlearning literature to operationalize various IS elements, discontinuance practices, and involved roles. • Sticking to interview protocols consisting of three sections: (1) getting to know the background of the company; (2) describing the whole replacement process; and (3) eliciting the discontinuance story by focusing on the various IS elements, practices and roles. • Adopting an open inquiry approach to reduce the bias due to imposing some terms on the informants (e.g., using "old stuff" instead of just saying "obsolete knowledge" (that could be interpreted differently by various actors). This also helped us capture a variety of elements in the discontinuance process. • Triangulation of data sources by asking questions at least from two relevant informants, and in case of inconsistencies, continuing the process to resolve it; covering external informants (e.g., ex-employees) to complement internal informants' opinions. • The main findings were presented to three companies (SMLFC, MedCo, and LargLenient) to identify and fix potential misunderstandings. • In the data analysis process, a second researcher re-coded 20% randomly selected of interviews (coding for IS elements and discontinuance practices). We discussed the new insights emerged in the second round of coding.
Internal validity	• (Through research design and analysis) by selecting extreme cases and through cross-case comparison we could capture the potential conditions that shaped the discontinuance process. • (During data collection) paying specific attention to "who did what," "when," and "why did actors did so" to identify the rationales and causes behind discontinuance practices.
External validity (generalization)	• There is no claim about generalization, since the aim of the paper is understanding deeply the discontinuance process. However, the comparison between companies helped us articulate mechanisms that explain how the process can be differently shaped by legacy IS conditions; thus showing the boundary conditions of the propositions.
Transparency and responsibility	• Relying on case study protocol and interview protocol. • Using a case study report for each company as a compendium of all information that helped us codify and articulate information from various sources. • Using ATLAS.ti for managing codes and memos, helping systematic iterations between data analysis stages, especially when some ambiguities required re-examination of the codes instances.