01 Apr 2020

The Reinforcing Effects of Formal Control Enactment in Complex IT Projects

Gloria Hui Wen Liu

Cecil Eng Huang Chua
Missouri University of Science and Technology, cecq8z@mst.edu

Recommended Citation

Liu, G. H., & Chua, C. E. (2020). The Reinforcing Effects of Formal Control Enactment in Complex IT Projects. *Journal of the Association for Information Systems, 21*(2), pp. 312-340. Association for Information Systems (AIS).
The definitive version is available at https://doi.org/10.17705/1jais.00603

Journal of the Association for Information Systems (2020) **21**(2), 312-340
doi: 10.17705/1jais.00603

RESEARCH ARTICLE

ISSN 1536-9323

The Reinforcing Effects of Formal Control Enactment in Complex IT Projects

Gloria Hui Wen Liu[1], Cecil Eng Huang Chua[2]

[1]Xi'an Jiaotong-Liverpool University, China, huiwen.liu@xjtlu.edu.cn
[2]Missouri University of Science and Technology, USA, cchua@mst.edu

Abstract

Complex IT projects pose particular challenges for the application of control, because of the dynamism and uncertainty involved. Prior studies suggest self-control can complement formal control within complex projects. However, how managers can enact controlee self-control remains an unsolved question. This paper proposes and investigates how enacted formal control unfolds during the course of an IT project and, in particular, how formal control enactment can promote or hinder controlee self-control. We demonstrate through case studies of a control in two wireless communication product development projects that an enabling control style can induce controlees to act to the benefit of both the controller and the controlee, while an authoritative control style encourages controlees' self-interested behavior. We also show how controlees influence the enactment of control within complex projects and demonstrate the reinforcing effects of the controller's enactment and controlee response on project outcomes. For practice, this research identifies preconditions necessary for inducing controlee self-control.

Keywords: Formal Control, Self-Control, Control Enactment, Control Style, Controlee Response

Jens Dibbern was the accepting senior editor. This research article was submitted on July 25, 2016, and underwent seven revisions.

1 Introduction

When complex projects are initiated, there is often no precise vision of the resulting product/system and no detailed specifications of the project tasks or a clear route to achieve project goals (Ahern, Leavy, & Byrne, 2014; Baccarini, 1996; Tatikonda & Rosenthal, 2000). New product development integrating new software and hardware is one form of complex project. Technology solutions of this type, such as internet of things (IoT) solutions (e.g., networked wireless communication products) are becoming commonplace. Such complex IT projects require the adoption of control practices from both product development as well as IS development (Tarafdar & Tanriverdi, 2018).

Controlee self-control is particularly important for such complex IT projects. Controlee self-control allows the "man on the spot" (i.e., the controlee) (Hayek, 1945, p. 524) to make decisions that cannot be addressed by prespecified controls. However, how managers can enact controlee self-control remains an unsolved question. Increasingly, empirical evidence suggests formal control is somehow related to controlee self-control (Grabski & Leech, 2007; Gulati & Puranam, 2009; Henderson & Lee, 1992; Huber, Fischer, Dibbern, & Hirschheim, 2013; Tiwana & Keil, 2009; Wiener et al., 2016). For example, in the context of 79 internal IS projects, Tiwana and Keil (2009) noted that both formal control (behavior control) and self-control contribute to effective project performance; Grabski and Leech (2007) found formal control and self-control statistically interact to explain the success of an internal IS project. However, the actual processual relationship between the two modes of control remains unknown (Gopal & Gosain, 2010; Tiwana, 2010). Often, controllers rely on

selection/recruitment processes to choose controlees who have the abilities and knowledge to exercise self-control (e.g., through screening or performance evaluation processes) (Choudhury & Sabherwal, 2003; Kirsch et al., 2002). We argue that part of the answer lies in the way that formal control is enacted.

An enacted view of control sees control enacted by the controller as an ongoing process (Orlikowski & Iacono, 2000). It is through the control enactment and controlee response that control comes into existence. One key aspect of control enactment is the controller control style, i.e., the manner by which controllers exercise their authority to shape controlees' experience of control (vs. the control itself that regulates controlees' behaviors) (Wiener et al., 2016). This paper studies such formal control enactment in a comparative case study of two wireless communication product development projects (i.e., IoT solutions) in which a newly implemented formal control (i.e., a checklist) was enacted in distinct ways, leading to distinct controlee responses.

In many complex projects, it is impossible to a priori specify all necessary controlee behaviors—there are often situational contingencies specific to the project that might possibly happen and cause problems (i.e., the situational contingency-formal control gap). These contingencies may arise from a project's environment and/or structures and procedures employed in a project (Engwall, 2003). Clearly, to achieve controller goals, it is necessary to address situational contingencies. This paper's principal contribution is a process model showing how the controller's different enactment of a formal control encourages particular controlee responses and vice versa, ultimately leading to the gap between the situational contingencies and formal control either being bridged by controlee self-control or being exploited for controlees' self-protection purpose only. The model explains how formal control can lead to controlee self-control, by highlighting the importance of an enactment of formal control in an enabling style (as opposed to an authoritative one). Enacting control in an enabling style involves the provision to controlees of contingent information (i.e., transparency) and the right to make decisions (i.e., repair). This creates a safe space and a collaborative culture for controlees to take independent, creative actions. When unspecified issues emerge, the controlee is able to exercise self-control to experiment with new behaviors to resolve these issues, leading to the situational contingency-formal control gap being bridged. Conversely, controlee self-control does not arise when formal controls are enacted in an authoritative style to the point where maladaptation and tension result. Here, the controlee attempts to protect him- or herself by performing behaviors the controlee feels will minimize punishment, rather than exploring new behaviors to address issues. The model provides practical recommendations for the controller to facilitate controlee self-control.

The rest of the paper is organized as follows. In Chapter 2, we review the relevant literature on organizational control and an enacted view of control. We continue with a description of the methodology in Chapter 3. Our findings are presented in Chapter 4, followed by a discussion in Chapter 5. Finally, some limitations to this study are noted and conclusions are drawn in Chapter 6.

2 Background Literature

The study of control has a long history in organization studies (Braverman, 1974; Jaworski, 1988; Ouchi, 1979, 1980). However, most research has focused on control in routine and permanent organizations (Wiener et al., 2016). One important contribution of contemporary IS research is to adapt the organization control literature to the nonroutine and temporary context of IS projects, leading to behavioral control theory (Choudhury & Sabherwal, 2003; Chua, Lim, Soh, & Sia, 2012; Kirsch, 1997; Mähring, 2002; Wiener, Remus, Heumann, & Mähring, 2015).

Within behavioral control theory, control refers to any attempt by a controller to ensure that a controlee acts according to predefined strategies to achieve organizational objectives (Kirsch, 1997). Researchers have identified two principal modes of control, formal and informal control. Formal control relies on the controller's hierarchical authority to monitor, evaluate and reward the controlee. There are three "sub" modes of formal control: input, behavior (process) and outcome (output) control (Eisenhardt, 1985; Govindarajan & Fisher, 1990; Kirsch, 1996, 1997; Ouchi, 1979; Wiener et al., 2016). Input control involves the management of human, financial and material project resources allocated by the controller. The controlee is rewarded or punished for his/her ability to utilize those resources. Behavior control prescribes rules and procedures. The controlee is rewarded or punished, depending on how faithfully procedures are followed. Outcome control prescribes the desired outcomes or goals and the controlee is rewarded/punished for meeting/failing goals.

Informal control is noted by the absence of the use of hierarchical authority to monitor, evaluate and reward. There are likewise two "sub" modes of informal control, clan and self-control (Jaworski, 1988; Kirsch, 1996; Kirsch & Cummings, 1996; Ouchi, 1980). Clan control and its equivalents of "cultural control," "normative control," (Fleming & Sturdy, 2011; Kunda, 1992) or "concertive control" (Barker, 1993) refer to proscriptions of behavior based on norms, ceremonies, and shared experiences, with particular significance placed on emotional relations of unity and solidarity with the organization or colleagues (Alvesson & Kärreman, 2007; Costas, 2012; Ouchi, 1980). Clan control works by socializing controlees to common norms or values (Ouchi, 1980). The "power source" of formal and clan control is external to the self. Controlees

are regulated either through the application of hierarchical authority (i.e., formal control) or institutions, or norms (i.e., clan control). Self-control, in contrast, refers to proscriptions of behavior managed by the self. It is commonly understood to be self-chosen and carries the connotations of individualism and self-actualization (Maruping, Venkatesh, & Agarwal, 2009; Ryan & Deci, 2000b; Wiener et al., 2015).

Substantive research has been done on formal control in organizations (Gopal & Gosain, 2010; Gregory & Keil, 2014; Heumann et al., 2015; Keil et al., 2014; Wiener et al., 2016). An increasing stream of research has acknowledged the risks associated with excessive formal control (e.g., operating delays, bureaucratic inefficiency, adaptive limits) and the need to leverage the unique strengths of informal control (Huber et al., 2013; Lioliou et al., 2014). Recent studies demonstrate that formal control can be enacted to induce clan control by creating/leveraging social capital (Chua et al., 2012; Kirsch, Ko, & Haney, 2010). However, research results on the interplay between formal and self-control are sparse and inconclusive. Some suggest that formal control impedes self-control because formal control signals a lack of trust (Ghoshal & Moran, 1996). Others argue that formal control can facilitate self-control by increasing interaction for building trust (Huber et al., 2013) or clarifying project boundaries (Kirsch & Cummings, 1996). There is a need for more in-depth research explaining how formal control unfolds and relates to self-control. We aim to fill this gap by studying the way control is enacted.

2.1 An Enacted View of Control

By enacting a control, we mean bringing it into existence by means of action (Weick, 1995). Control is not merely a mechanism selected and implemented by the controller. Controlee action is also essential for realizing the potential of the selected control toward the fulfillment of controller goals (i.e., goals the controller wants to achieve through implementation of control). Control is thus an ongoing production by both the controller and controlee (Orlikowski & Iacono, 2000). The resulting outcome is essentially the product of both the controller's (control enactment) and controlee's making (controlee response) (Chua & Myers, 2018; Weick, 1995).

When controllers enact controls, they set or clarify rules, define the conditions for the controlee to act in, and employ a specific vocabulary of motives. For example, given a standard operating procedure (SOP), an employee may perform a task incorrectly and receive a verbal reprimand. This reprimand can take a number of forms. The controller could denigrate the controlee without clarifying the fault and threaten the controlee with severe punishment if any deviation happens again. Alternately, the controller could explain the fault and encourage the controlee to propose improvements to the SOP. In both cases, the same control is employed, but the control is enacted differently. Indeed, the controller can enact a control to create positive (e.g., clarification of rules) or negative (e.g., psychological distress) conditions for controlee task performance.

Likewise, the control enacted by a controller does not fully determine a controlee's actions (Chua & Myers, 2018). When the controller enacts a control, he or she creates a space in which the controlee can perform an action. The controlee's response could be one of a range of possibilities, which creates consequences that are intended or unintended by the controller. Continuing with the same example, as a result of the verbal reprimand, the controlee could choose to be more careful (i.e., intended by the controller), or could feel resentful and deliberately sabotage work in a way that could not be traced to him or her (i.e., unintended by the controller) (Prasad & Prasad, 2000).

2.2 A Preliminary Analytical Model

To account for the aforementioned duality between control enactment and controlee response, a temporal model is required for structuring the process. When a controller enacts a new control, this causes changes to extant patterns of action. The full implications of these changes are not immediate, as controlees need to reinterpret the situation and respond (Van de Ven & Poole, 1995). Furthermore, the actual outcomes of these changes may be uncertain (Merton, 1936). We therefore model the enactment process sequentially, without presuming any specific progressive developmental logic (Barley, 1986; Langley, 1999). Figure 1 presents our theorization of the relationship between control enactment and controlee response. In the figure, the enactment of formal control occurs in temporal episodes (E1, E2, E3, etc.). Both the controller's enactment and the controlee's response shape the context.

The controller's enactment shapes the context, because the control itself is now part of the context. Both the controller and controlee must make future decisions based on the existence of this control. For example, a controller offers a $1 bounty for every bug found in someone else's code. This, however, leads to collusion among developers who deliberately insert easy-to-find bugs in their code so that everyone gets the bounty. The controller then has to enact new controls in this new context, for example, by declaring that in instances where there are more than 10 bugs per module, the developer who created the bug has to pay the bounty. Control thus presents the controller and controlee with opportunities to reciprocally shape their context. The variety of ways through which the controller may enact formal control and the controlee may respond to control enactment under evolving contextual situations is the focus of our analysis.

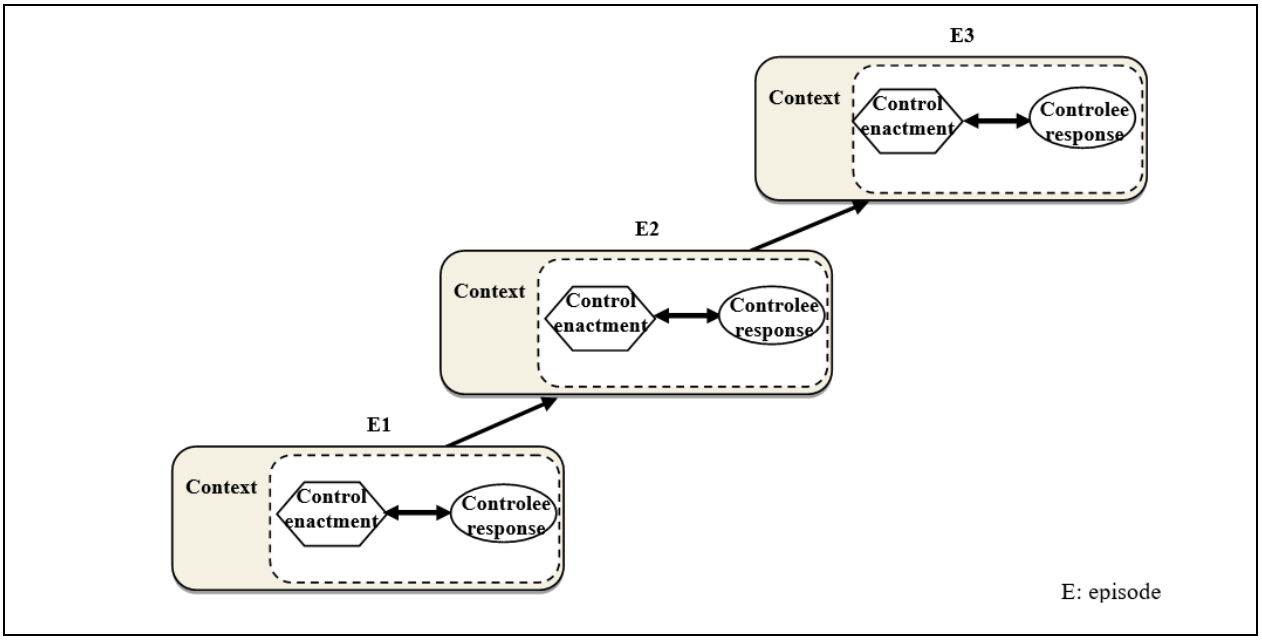


Figure 1. A Preliminary Analytical Model

2.2.1 Control Enactment

The controller can choose to emphasize formal control through the use of threats or sanctions, or through choice and feedback (Deci & Ryan, 1987). The way the controller enacts control can shape controlees' psychological reaction (Steinberg, 2005) and attitudes toward a control (Ryan & Deci, 2000b). This in turn shapes their behaviors. The controller can enact control in two distinct styles: enabling and authoritative (Wiener et al., 2016), representing the two extreme ends of a continuum. The enabling style, and similar styles of "responsible autonomy" (Friedman, 1990) and "quasi-autonomous pattern" (Alvesson & Willmott, 2002), recognizes that the controlee is intelligent and that not all tasks are programmable. Therefore, when adopting the enabling style, the controller focuses on mobilizing the controlee's capacity to exercise discretion to attain controller goals. In contrast, the authoritative style, and its similar styles of "direct control" (Friedman, 1990) and "managerial pattern" (Alvesson & Willmott, 2002), is top-down, unyielding, and punitive (Ahrens & Chapman, 2004). The focus of the authoritative style is on controlee compliance through detailed specification of tasks and close supervision.

Two features distinguish the enabling from the authoritative style, namely *transparency* and *repair* (Adler & Borys, 1996; Wiener et al., 2016). In transparency, information about control activities and the context are provided to controlees to develop their understanding as to how formal control facilitates controller goals and how their tasks relate to these goals (Adler & Borys, 1996). However, instead of having information randomly pushed toward the controlee, the information is made available on demand and is made intelligible to the controlee. For example, the controller makes available related information on a shared server and categorizes it to make it easy for controlees to navigate (e.g., by organizing Excel worksheets in temporal order). In contrast, controllers that adopt the authoritative style treat controls as assertive instructions or prescriptions, keeping information about the context and control activities from controlees (Wiener et al., 2016). Information distributed by authoritative-style controllers is meant to determine and structure controlee behavior and put constraints on controlee judgment (Alvesson & Willmott, 2002). For example, a controller issues a policy and takes away controlees' rights to challenge it.

Repair is concerned with deviations and breakdowns in control activities. A controller adopting the enabling style anticipates breakdowns and appreciates the controlee's potential contribution to repair them. The enabling style views formal control as an imperfect tool for achieving controller goals. The controlee is given discretion to disregard control when judgment is required (Ahrens & Chapman, 2004; Hoy & Sweetland, 2001; Wouters & Wilderom, 2008). For example, when controlees realize that a requirement is more complex than anticipated, they discuss changes to the project schedule with the controller. In contrast, authoritative-style controllers anticipate no deviations and fear controlee opportunism more than their potential contribution to fix breakdowns; they enact control to ensure compliance under all circumstances (Adler & Borys, 1996; Ashforth, 1997; Cooper & Taylor, 2000). For example, a controlee is penalized if he or she is unable to make a deadline even though the

issue concerning the requirement is more complex than initially estimated. In the authoritative style, threats and sanctions are frequently used to minimize deviations (Huebner, 2003).

2.2.2 Controlee Response

Given the enacted control, the controlee can choose to embrace the goals intended by the control (i.e., a precursor to self-control) or can consider such goals to be external and separate from the controlee's goals. *External control* means controlees perform an activity because they feel external pressure to perform the activity (e.g., formal control, clan control). They have no intrinsic desire to perform the activity. When a controlee responds to controls as external, the controlee expends relatively little capacity (e.g., energy, cognitive resources) to execute commands because the controller or peer is charged with setting the goals or rules, monitoring the controlee's performance, and taking actions to reward or punish the controlee. In contrast, *self-control* requires a controlee to set his or her own goals, monitor the difference between the actual state and the goals, and take actions to close the gap accordingly (Baumeister, 2002; Baumeister, Vohs, & Tice, 2007; de Ridder et al., 2012; Vohs et al., 2014). Self-control comprises three factors: self-goal setting, self-monitoring, and intrinsic motivation (Baumeister, 2002; Baumeister & Heatherton, 1996; Kirsch, 1996; Kirsch & Cummings, 1996).

Self-goals are the desired outcomes or ideals set up by the controlee (Baumeister, 2002). The goals can direct attention, mobilize efforts, increase persistence and motivate personal action toward goals not prescribed by the controller (Latham & Locke, 1991). For example, an employee wants to achieve a certain standard of performance because this would give them a positive feeling. While goals are usually set up by the controller, the controlee can internalize or derive higher goals from said controller goals. For example, a project member decides to submit deliverables earlier than required by management. At the workplace, a defined performance standard, quota (the amount of work or production), time limit, or budget for completing a task can influence a controlee's goals and become internalized.

Self-monitoring involves systematic information gathering about one's own behavior and comparing the actual state with the ideal standard. For example, office workers attempting to achieve a specified level of energy saving must monitor their own energy consumption patterns (Yun et al., 2013). Self-monitoring is critical to self-control because it allows the controlee to obtain important information for the regulation of personal behavior (Baumeister, 2002; Baumeister & Heatherton, 1996; Miller, 1987). Substantial empirical work has demonstrated the necessity of self-monitoring. For example, the failure to monitor posture while performing typing tasks or assembly tasks means that safe posturing does not occur (Gravina et al., 2008). Similarly, when forklift drivers failed to monitor their performance, time spent on-task increased (Ludwig & Goomas, 2009).

Intrinsic motivation is the most important element of self-control (Baumeister & Heatherton, 1996; Manz, 1986; Manz & Sims Jr., 1987). To be motivated means to be moved to do something. Without motivation, self-goal setting and self-monitoring are insufficient for inducing self-control. Motivation can be either intrinsic or extrinsic (Ryan & Deci, 2000b). Intrinsic motivation exists when tasks are interesting, enjoyable, or satisfy the innate psychological need of competence (Harter, 1978), autonomy or relatedness (Baumeister & Leary, 1995). Nidumolu and Subramani (2003) found that intrinsic motivation facilitated self-control in software developers. Extrinsic motivation occurs when performing a request allows a person to obtain a separable outcome, such as to satisfy an external demand or obtain some reward (Ryan & Deci, 2000a).

2.2.3 Context

Controllers and controlees are embedded in a context that has other preexisting controls (Choudhury & Sabherwal, 2003; Kirsch, 1997). These *other controls* along with *noncontrol factors* (e.g., parallel events, organizational politics, institutionalized routines) constitute the context that shapes and is shaped by the controller's and controlee's actions.

First, there is a conglomeration of controls of different origins and disparate ages in the organization. While some may be tailored to a specific task, others originate from separate functions or professional communities. While some endure for a long time, others are only enacted for a brief duration (e.g., the life of the project). For example, the employee's employment contract and dictates from the human resources department all influence controlee behavior. If existing controls align with controller goals, there is a strong probability that controlee behaviors will likewise align with these goals. If not, inefficiency, conflicts of interest, and confusion about the contradiction between what these controls dictate and actual behavioral patterns on the part of the controlee will probably exist. It is also likely that the controller can improve the situation by removing or relaxing some controls or the controlee can resist or work around the controls.

Second, there are many noncontrol factors that occur simultaneously with control enactment and controlee response (e.g., other projects) (Engwall, 2003), or that accumulate from prior interaction (e.g., trust) (Huber et al., 2013). These factors require resources and energy to deal with and sometimes produce outputs of little utility or even negative consequences. For

example, a controlee tasked to work on project A might be distracted by his work on project B, especially if project B is late. In other words, project B drains the controlee's energy and motivation to work on project A. However, the controller can help conserve the controlee's energy by rescheduling project B or allocating extra resources.

Clearly, control can become outdated and fail in changing circumstances. Control can thus impede performance and can cause negative feelings such as stress or alienation (Chua & Myers, 2018). Yet self-controlled controlees will take the initiative to set self-goals that are aligned with controller goals and monitor said goals, regardless of situational contingencies. Self-control can thus complement formal control to address any project-specific situation for achieving controller goals. We refer to the possibility that formal control can misspecify behavior in specific situations as the situational contingency-formal control gap. This gap can be potentially bridged by self-control.

3 Methodology

To develop a process model about control enactment, we followed an exploratory multiple-case research approach (Dubé & Paré, 2003; Eisenhardt, 1989; Yin, 2003). This is an incremental, theory-building approach where researchers generate and augment insights about control enactment by iterating between theory and data. We entered the field with the preliminary model and constructs in Figure 1. The first researcher was engaged in collecting data about the two cases and theorizing; the second researcher carried on with the theorizing but did not participate in data collection. Incidents in each case were constantly compared with each other, between the two cases, and with theory.

In this study, the same control mechanism was applied to two product development projects in the same organization in two distinct ways. In one project, the controller enacted formal control to enable staff to exercise self-control, benefiting both the controller and controlee; in the other, the controller enacted the control to prompt controlee action. However, enacted controlee action was only for the purpose of self-protection.

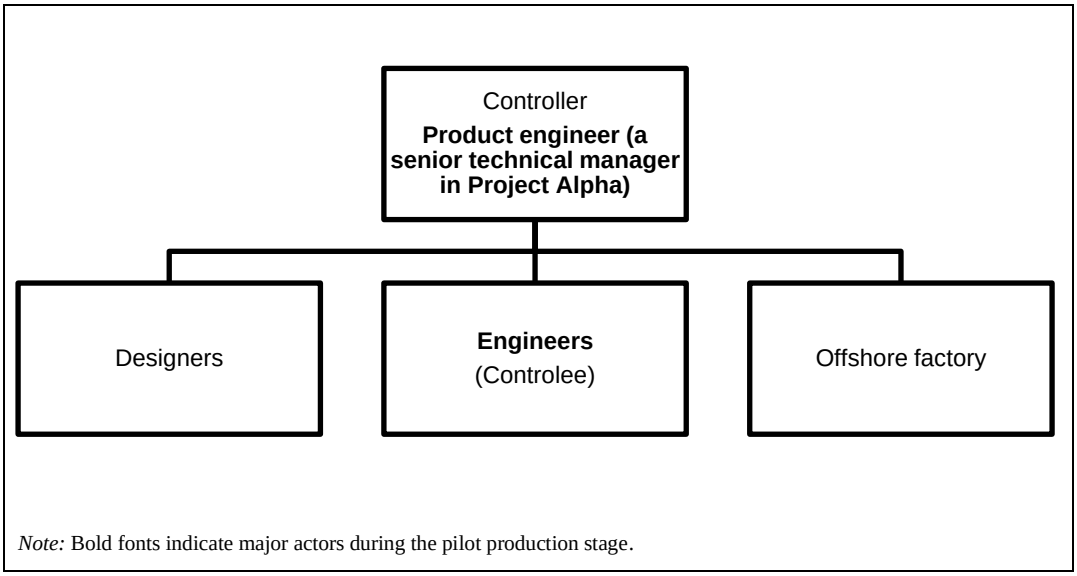
3.1 Site Description and the Manufacturability Readiness Review

The study was conducted at a large manufacturer (hereinafter designated as MassCo). MassCo designs and manufactures wireless communication products that embed both software and hardware to develop new product features/functionality (e.g., TV boxes that stream video on demand). Thus, software engineering was a critical element of each product. Its headquarters are in Taiwan and it has production sites across Asia. MassCo employs about 1500 employees worldwide and has six major business units. Given the distribution of manufacturing sites in Asia, a manufacturing center (hereinafter called "the Center") was established to coordinate and control production. The two product development projects studied were part of the Center, comprising 45 engineers (e.g., mechanical engineers, product engineers) working in an open-plan office. These engineers had undergone distinct training associated with their areas of specialization. Engineers also tended to work on specific product lines.

Therefore, the same groups of engineers tended to work together over time, being responsible for separate issues associated with production planning and pilot production of new products. Apart from formal control, the engineers had professional autonomy based on their collective or individual competence. Their daily routines required that they make many "judgment calls" (mechanical engineer, Project Beta) to deal with issues emerging from the shop floor.

Table 1. The MRR's Major Functions in Different Technical Areas

Technical area	Description
Product design	Guides engineers through the process of verifying/testing product design, both hardware and software (e.g., PCB assembly, mechanical assembly, and underlying software programs).
Machinery	Guides engineers through the process of programming for process automation and reviewing equipment and fixtures needed, including their capacity, availability, and performance.
Workmanship	Guides engineers through the process of reviewing manpower needed (e.g., operators, maintenance technicians, training)
Materials	Guides engineers through the process of reviewing materials needed, e.g., PCBs, electronic parts, mechanical parts, packing materials, etc.
Production procedures	Guides engineers through the process of verifying/testing standard operating procedures.
Routine reports	Generates reports of pilot production outcomes in tabular/chart format.



Note: Bold fonts indicate major actors during the pilot production stage.

Figure 2. Chart of Project Organization During Pilot Production

It is often the case that poor performance tends to encourage the controller to enact more controls (Choudhury & Sabherwal, 2003). In an attempt to raise the yield rates (i.e., ratio of good units produced) of new products at MassCo, a new control was enacted called the Manufacturability Readiness Review (MRR) for the purpose of better control over the pilot production process. This took the form of a checklist with multiple worksheets encapsulated in a Microsoft Excel workbook designed to verify production- and design-related problems. The MRR was both an outcome and behavior control. It aimed at regulating the pilot production outputs (e.g., yield rates) and activities associated with generic design/production requirements of all products at MassCo. The MRR was arranged in temporal order with items on the same worksheets that could be executed simultaneously. The controller could easily reorganize the checklist for information processing/transmission. Table 1 lists the major MRR functions in different technical areas.

At MassCo, most production issues, including those related to materials and production procedures were specified by designers in the bills of materials (BOM) and preliminary SOP. The engineers then applied their technical skills to perform routine checks. One intent of the MRR was to increase engineers' early participation in product analysis by requiring them to audit product design, including both hardware and software and "their integration" (surface mount technology engineer, Project Alpha). The product design function required engineers to apply in-depth domain knowledge to detect design errors early to reduce modification and production cost. For example, items on the checklist about the position of components (e.g., angles or distance from other components, or external covers) were introduced to encourage engineers to use their judgment to prevent undesired outcomes, such as damage to or interference with other components, unstable assembly, or even functional failure. Items about software testing (e.g., associated parameters, versions, procedures, hardware support/deployment) were also included to allow engineers to better manage the process and identify causes of execution failure (e.g., software bugs, testing methods, or hardware design).

The idea was that the engineer could compare pilot production preparation against an MRR item, and then check off each item after the process was executed. Once all processes were executed, the MRR was passed to one of MassCo's production sites for volume production. The MRR checklist was linked to engineers' annual appraisals. If a project resulted in volume production problems and the engineer failed to execute and check off corresponding items, that engineer could not be higher than the 60th percentile in performance evaluations.

3.1.1 Control Relationship

New product development at MassCo followed a sequential process, with the product engineer taking control once the project entered the pilot production stage. Generally, product engineers were charged with overseeing production (including pilot production) to meet an overall schedule set by the project manager and performance indicators by the Center. The MRR thus was the most comprehensive of many controls that the product engineer could apply to monitor the project. Figure 2 depicts various stakeholders in a product development project and their formal relationships as controllers and controlees. During the pilot production stage, the major actors included the engineer (controlee), and product engineer (controller) who directly controlled the engineer or mediated the influence of other controllers on the engineer. In Project Alpha, the controller was a senior technical manager.

3.2 Data Collection

The first author was invited to study the MRR adoption by the Center director. She entered the field site in May, about five months after the MRR rollout, and stayed until November 2009 to collect retrospective data. The first author visited the field at fortnightly intervals and stayed for half a day or longer to collect data on each visit. Because the MRR was mainly associated with three engineering functions, the first author focused on getting information from these three engineering sections (i.e., surface mount technology, mechanical engineering, and product engineering). The senior technical manager who led the first project (Project Alpha) suggested it as a case study to the first author. Data about how control enactment unfolded in Project Alpha was collected within one month of its completion. As data collection continued, the theme of self-control emerged, as we noticed the engineers voluntarily monitored their own performance and set challenging goals for themselves.

To extend and test our understanding of how and why formal control can be enacted to lead to controlee self-control, we needed to examine a project where the MRR was poorly enacted, in accordance with the principles of theoretical sampling in which contrasting results occurred (Eisenhardt, 1989; Lee, 1989; Yin, 2003). The senior technical manager provided a list of recent projects (both finished and ongoing ones). The first author screened these projects looking for incomplete MRR checklists. An incomplete MRR checklist signaled the MRR was not enacted in line with controller goals. It was thus a proxy for outcomes unintended by the controller. Initially, the researcher flagged two ongoing projects as potential contrasting cases. After the projects finished, the researcher solicited the MRR checklist of these projects through the assistance of the senior technical manager. Project Beta featured the most improperly filled checklists (demonstrating unsatisfactory MRR enactment). Therefore, it was chosen for comparison with Project Alpha (Yin, 2003). In terms of similarities, both projects employed the MRR to control production and both had similar numbers of engineers and two designers: Project Alpha involved seven engineers, while Project Beta had nine. Both projects had engineers with similar types of work experience (see the information for industry and company tenure in Table 2). Both the senior technical manager in Project Alpha and the product engineer in Project Beta had formal authority to implement the MRR as a controller. The similarities between the projects allowed for the replication/extension of emergent theory (Eisenhardt, 1989).

For each project, data were collected through interviews, internal control documents, and site visits. The predominant method of data collection was in-depth interviews at the site over a period of seven months. The first author conducted semistructured interviews with interviewees from different organizational levels (management and nonmanagement) and engineering sections. Questions were asked about interviewees' positions and roles within the organization. Then, questions were asked about the processes of the new product project from a project management perspective. Such questions focused on general control and coordination problems in new product projects. Finally, specific questions about the MRR enacted in the pilot production process were asked. Such questions included those on facilitation, performance standards, monitoring, reward/punishment, and consequences associated with the MRR implementation. Overall, interviews lasted between one and two hours and were recorded and transcribed verbatim. A total of 17 interviews with 13 interviewees were conducted in MassCo's Taiwan premises. Table 2 summarizes the interviewees' background information for both projects.

The collected control documents contained 569 pages of data and included MRR checklists for each project, including meeting minutes, orientation and training materials, engineering change requests, and email communication. We also reviewed engineering design drawings, meeting presentation slides and other project documents. In many cases, the documents helped illuminate and clarify earlier insights drawn from interviews. They alerted the first author to important leads to pursue or contradictions to understand and allowed inferences to be made about the agendas and interests of the various actors. For example, in an email thread on a minor production issue, a product engineer carbon copied the email to an SMT (surface mount technology) engineer's direct supervisor to ensure his compliance. The behavior pattern we observed in the email exchange corresponded with our interview data. Documents also helped ameliorate the retrospective nature of the interviews. We could cross-index statements made by the interviewees, making it easier to establish when events actually occurred, because of the date/time stamps on the documents.

During site visits, the first author observed the site layout, and the interaction between participants. Site visits were arranged before interviews started so that the first author could gain a preliminary understanding of the site. For example, the first author, wearing a cleanroom suit, was accompanied by the Center director on visits to assembly lines and cells in MassCo's Taiwan facilities. After data collection started, the first author sometimes went into town with interviewees for meals. Field notes were taken on the day of the visit. They included notes on informal conversations, drawings of the site, the researcher's reflections and further questions to be pursued.

Table 2. Background Information of Interviewees

	Title	Industry tenure (in years)	Company tenure (in years)	# interviews
Management	Center director	23	13	2
	Senior technical manager	12	4	3
	Sectional head of mechanical engineering	7	7	1
Case Alpha	Project manager	11	10	1
	SMT engineer	12	3	1
	Mechanical engineer	7	6	1
	Product engineer	11	4.5	1
	Designer	3	2	1
Case Beta	Project manager	11	10	1
	SMT engineer	11	5	1
	Mechanical engineer	4	4	2
	Product engineer	11	2	1
	Industrial engineer	12	4	1
Total				17

3.3 Data Analysis

Informed by our preliminary analytical model, each case was split into episodes of interaction that were triggered by a major project event (e.g., release of a prototype) and ended with changes in the control context (e.g., MRR as a reference point for two-way communication). Each episode had a certain continuity in the activities, separated by discontinuities in the behavioral patterns associated with the dissemination and adoption of the MRR in the project (Langley, 1999). For example, in one episode, the controller lifted communication barriers and crafted channels for communication (e.g., meetings) to improve project transparency; controlees then could use the MRR to collect information for product analysis. This facilitated the construction of controlees' capacity for self-monitoring and became the point of departure for another episode. By examining successive episodes of social interaction, we could understand how (inter)action in one episode led to changes in the context that affected action in subsequent episodes. Three episodes of social interaction surrounding the focal control (i.e., the MRR) were compared across the two cases.

The analysis proceeded in three stages, following the logic of constant comparison (Eisenhardt, 1989; Huberman & Miles, 1998; Yin, 2003). In the first stage, we analyzed documents and interviews to uncover the MRR's main technical areas and the sequences for the enactment of those areas in each case. By the end of the first stage, we had an understanding of the chronological flow of events and could identify three key episodes for each case. In the second stage, we coded and mapped pieces of evidence to elements of our preliminary model to draw a rich picture of social interaction within each case. A draft codebook was created after the two authors read and discussed separate case profiles and mapped a set of quotes to concepts in our preliminary model (i.e., deductive coding) (Crabtree & Miller, 1999). The first author coded the data, while the second author played devil's advocate. Discrepancies were discussed and the codebook was modified to achieve agreement. Episodes of social interaction were examined to analyze how and why one episode led to another. New concepts were allowed to emerge and were categorized (i.e., inductive coding). These new codes captured the patterns and reasons for controller/controlee (inter)action, enactment outcomes of individual episodes, and project outcomes. Together, the coding uncovered (1) what the controller and controlee did in association with the MRR; (2) why they behaved in a particular way; and (3) what consequences the enacted MRR had. Sample categories/concepts coded and developed are shown in Table 3.

Table 3. Representative Quotes Grouped According to Code Categories

Category	Concept	Definition	Quote
Context	Other controls (formal)	A mechanism specifying desired outcomes/behaviors as a policy or rule for the engineers to engage in with explicit contingent rewards/punishments ensuing	*… had to meet our departmental KPIs [key performance indicators], like overall productivity, MOH [manufacturing overheads] …* (industrial engineer, Project Beta)
	Other controls (clan)	A mechanism instituted by a group of individuals and relying on shared norms, values, or beliefs to regulate behaviors	*[Team members] could check each other's schedule, knowing who had been careless filling in the doc or who had lagged behind … a source of pressure … forced them to close issues … or the team would be in trouble* (senior technical manager, Project Alpha)
	Noncontrol factors	Contextual factors influencing the controller's or controlee's choices of action	*… I felt like I had no power to make any decision.… The R&D [designers] meddled in things. We only made decisions on minor issues. For important issues they would step in and make decisions for us* (mechanical engineer, Project Beta).
Control enactment	Enabling style	Controllers disseminate information about the MRR and control activities and allow controlees to adapt the MRR	*I was not more knowledgeable than the mechanical or product engineers.… They told me what to add to the checklist and then I did it accordingly … new version is available [on a shared server]* (senior technical manager, Project Alpha)
	Authoritative style	Controllers hide information about the MRR and control activities and disallow controlees to deviate from the MRR	*I queried each engineer [about their progress] and checked off items…. I didn't send the doc to the engineers … unless something serious happened. Then I would highlight the MRR item and send [the checklist] to the PIC [person in charge]…. I would send the doc to his direct supervisor…* (junior product engineer, Project Beta)
Controlee response	External control	Controlees receive goals/means/ rewards/punishments from others and take no ownership	*… We're only there to support them [the project manager and the designers]…. It's the PM who had to face the time-to-market pressure…* (industrial engineer, Project Beta)
	Self-goals	Controlees have a high degree of influence in determining goals for themselves or set goals higher than those set by others	*[Finding design bugs] is an important part of our work…. [One bug] persisted … we judged it to be a design issue. They [designers] first were not persuaded and argued that this couldn't be the cause. We kept testing it with many different methods.* (product engineer, Project Alpha)
	Self-monitoring	Controlees gather cues about their behaviors/performance, track deviations and strive to come up with solutions	*I suggested the addition of two or three items to the [MRR] checklist [for the controller's approval]. For the past few months, I'd encountered several problems with packaging and labeling, like the label stuck out from the surface or did not stick ... they were minor issues, which could easily escape our notice…* (mechanical engineer, Project Alpha)
	Intrinsic motivation	Controlees demonstrate a high level of autonomy or identify design auditing as congruent with personal needs	*We started the manufacturability checks at the lab pilot run phase. If we could, we would have done it much earlier… it's exactly what we had to do. I don't think it increased our workload.* (product engineer, Project Alpha)

Table 3. Representative Quotes Grouped According to Code Categories

Enactment outcomes	Facilitation/ hindrance	Formal control enacted enhances/depletes controlees' capacity/motivation for self-control	*… [With the MRR] now the project manager, layout [engineers], R&D [designers] could simply use the checklists on their own.* (SMT engineer, Project Alpha)
	Compensation/ reinforcement	Weaknesses of formal control are compensated for/reinforced	*There's no way I could get the checks properly done [due to situational constraints]…. [The product engineer] kept asking for the data to fill in the MRR. In the end, I made a guess…. The guess could be very wrong.* (industrial engineer, Project Beta)
Project outcomes	Project deliverables	A combination of deadlines, product functionality or service quality in deliveries	*… two weeks ahead of the scheduled delivery date … impressed by our quick response and service quality … [the client] was very happy about that*. (project manager, Project Alpha)
	Socioemotional consequences	Nonfinancial aspects of projects that (fail to) meet controlees' psychological needs (e.g., the ability to exercise influence, recognition)	*We didn't have the power to block the product … very frustrated … whenever there's a problem, the project manager always got very nervous and questioned our solution…* (SMT engineer, Project Beta)

In the third stage, we moved from episode-level, to case-level analysis and comparative case analysis. A model was inductively constructed for each case, explaining linkages between control enactment and controlee response in the context of the three episodes. These two models were then compared to allow us to arrive at our final process model.

4 Findings

The MRR was implemented and available for use in early January 2009. By the beginning of data collection, engineers had already signed up for and attended one of two MRR training sessions at the Center. During the training, the engineers were introduced to the major technical areas of the MRR. Members from both projects thus heard about what the MRR was and the information it contained. However, when and how to apply the MRR along with other controls was the purview of a senior technical manager in Project Alpha and a product engineer in Project Beta.

An examination of the social interaction surrounding the MRR for the two projects revealed the same three episodes, coinciding with three product development phases at MassCo. These were the: (1) lab pilot run, (2) engineering pilot run, and (3) pilot production run.

During the lab pilot run (Episode 1), designers were charged with producing prototypes, occasionally with engineers' input about product design. The lab pilot run was critical for engineers to learn about the new product and to identify design errors from the production perspective. In relation to this episode, we discuss how the controllers in the two projects distributed distinct messages about and enacted the MRR in ways that would develop (or not) engineers' capacity to handle their new responsibilities (i.e., "constructing capacity" vs. "constructing constraints").

During the engineering pilot run (Episode 2), engineers produced a small number of products to ensure the new design was producible within a particular manufacturing environment. Production problems were explicitly considered as part of product design. The two groups applied the MRR differently to debug the product and production process—engineers in Project Alpha adapted the MRR to their tasks and communicated interactively with designers ("negotiating discretion"), whereas in Project Beta, the MRR was principally used to coerce engineers into performing tasks they felt were not in their purview ("negotiating dependence").

During the pilot production run (Episode 3), a relatively large number of products were produced using the mass production process. This run was designed to identify issues that would arise during mass production. Deliverables to controllers became more serious because any problem would increase production costs, resulting in a poor project outcome. In this episode, engineers used the MRR to resolve pending problems and detail future solutions for unresolved problems in Project Alpha ("toward problem solving"). In Project Beta, the resolution of many problems was delayed, and engineers used the MRR to cover problems and avoid blame ("toward self-protection").

Overall, we noticed a decisive difference in the way the MRR was used within the two projects right from the beginning. Understandably, the final enactment outcomes differed.

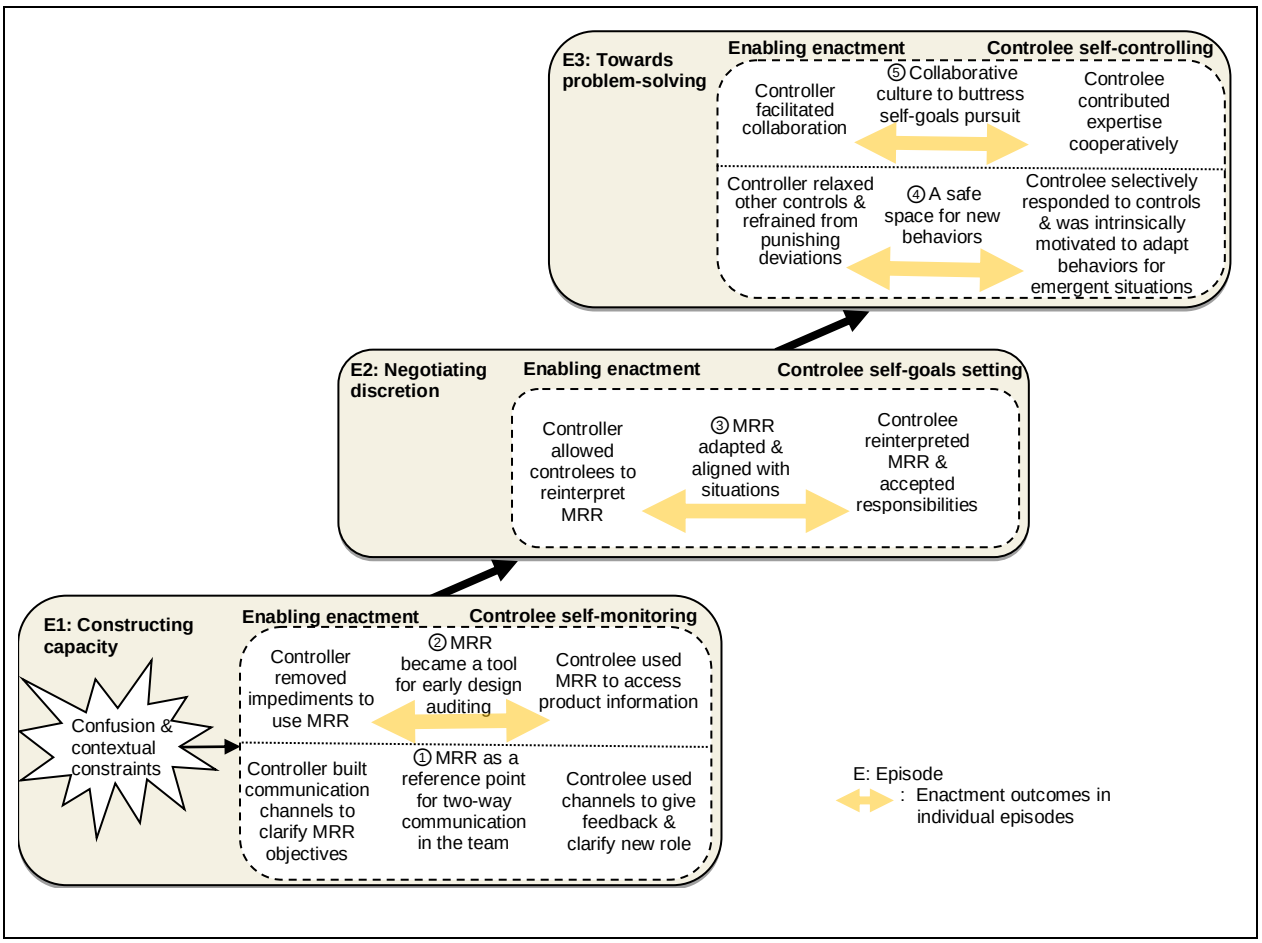


Figure 3. How Formal Control was Enacted to Facilitate Self-Control in Project Alpha

4.1 Project Alpha

Project Alpha was commissioned by a Fortune 500 company. This was the first time MassCo became part of this client's supply chain. However, the project schedule was extremely tight. The client's international purchase officer in Taiwan even considered on-time completion to be "impossible."

When the project started, a senior technical manager from the Center, Clark (the controller), solicited project information from the project manager. He realized engineers had been given about four weeks to complete pilot production and volunteered to help implement the MRR in Project Alpha. Clark was originally trained as an SMT engineer, was known for his emphasis on punctuality, and was nicknamed "Clock." He was an architect of the MRR, and "[had] good knowledge of its spirit" (Center director) and the authority to settle disputes over the MRR. Figure 3 provides an overview of how self-control was enabled in Project Alpha over the three episodes. The figure will be elaborated on in the following results description, with italicized headers of individual paragraphs corresponding to enactment outcomes in each episode (i.e., yellow arrows in Figure 3).

Episode 1: Constructing Capacity. Early in the lab pilot run phase, Clark wanted to provide Project Alpha with a high degree of formalization. He felt the MRR implementation would facilitate the swift delivery of the overall pilot production outcome. However, there were situational contingency-formal control gaps between the formal control (i.e., the MRR) and the project situation, including engineers' confusion about the MRR and contextual constraints that made it difficult for engineers to follow the MRR. He thus made clear his expectations of the MRR to the engineers. Clark also coordinated with the project manager and Center director to remove obstacles to the MRR's implementation.

MRR as a reference point for two-way communication in the team. Clark held a briefing to explain the MRR context and emphasized how its implementation would help identify and solve problems, especially those concerning product design. He also placed the MRR checklist on a shared server accessible to the engineers, the project manager, and the designers.

Engineers thus understood they had the additional role of a product design auditor and "the MRR was originally designed for [another product line with

problematic yield rates]" (SMT engineer, Project Alpha). Engineers were encouraged to get involved in product analysis activities as early as they could and to raise issues with the MRR.

The engineers discussed with Clark obstacles that could impede the MRR implementation. Two major obstacles at that time were (1) designers' domination over engineers, and (2) other projects that competed for resources available to engineers (e.g., limited factory facilities or having to pay attention to multiple ongoing projects). At MassCo, designers rarely sought engineers' opinions, and tended not to share project information without being asked. Clark thus solicited support from the project manager for the MRR implementation. He invited the project manager to the first meeting with engineers. As a result, the project manager understood that the MRR could "provide a platform for discussion" between engineers and designers. After the meeting, the project manager explained the MRR to designers and demanded they respond to issues raised by engineers. Two-way communication between engineers and designers was thus established, thereby enhancing transparency to help engineers clarify controller goals and their new role (Figure 3, Arrow ①).

MRR became a tool for early design auditing. With the consent of the Center director, Clark also helped negate noncontrol factors that could constrain engineers' performance of their new roles/responsibilities. For example, he set aside other big projects for them and rescheduled several smaller ones to ensure that the facilities they required were made available.

Engineers felt that they were well supported and could use the MRR to access product information for design auditing. For example, engineers could use the MRR to demand the early release of information from designers to discover design flaws soon after introduction. The information included engineering drawings, designers' design bug reviews, and functionality test plans (e.g., versions of software and parameter configurations). They were sometimes hyperlinked to MRR items to support engineers' judgment about the design. Engineers could also provide results of their analyses to designers as early as "the lab pilot run phase" (product engineer). Once engineers and designers agreed on the design, engineers could commit themselves to improving production (Figure 3, Arrow ②). That is, impediment removal further enhanced transparency to allow engineers' self-monitoring.

Episode 2: Negotiating Discretion. When the engineering pilot run started, engineers already understood the MRR's history and rationale (and its embedded goals). They executed the MRR and checked off items accordingly. When disputes and problems with the MRR arose (e.g., relevancy or ownership of MRR checklist items), they were fully discussed to further clarify controller goals (i.e., product auditing).

MRR adapted and aligned with situations. Clark allowed engineers to individually reinterpret and adapt MRR items to the situation (e.g., reassigned ownership). For example, engineers could classify some items as "must-do" or "recommend-to-do" (SMT engineer) and were able to perform checks as they considered appropriate. Furthermore, engineers were able to suggest additional items or critical elements that had not been considered in the MRR based on their prior experience. Clark used those suggestions to improve the MRR.

> *I suggested the addition of two or three items to the [MRR] checklist [for the controller's approval]. Over the past few months, I'd encountered several problems with packaging and labeling, like labels that stuck out from the surface or did not stick ... they were minor issues, which could easily escape our notice. With them being included, we could prevent them* (mechanical engineer, Project Alpha).

Engineers successfully "revealed many design problems" (senior technical manager, Project Alpha) and recorded them in the MRR for further designer audits. For example, the MRR encouraged engineers to examine the alignment of holes in the mechanical assembly from different perspectives, including manufacturing efficiency, ease of maintenance, and client use. Three major potential defects were discovered. Engineers informed designers about these defects and inquired about design changes. Designers explained that one of them was unchangeable due to the client's insistence on this design feature and demanded more evidence of the other two defects. Engineers kept monitoring the defects, finding approximately 40 production problems but resolving many of them.

Because of engineers' individual interpretation of the MRR and its ensuing modification (i.e., repair enacted), they accepted product auditing as their responsibility and readily embraced the adapted MRR to master their tasks (i.e., self-goals setting) (Figure 3, Arrow ②). Their acceptance of this additional responsibility was due to the utility of the MRR in helping them perform their tasks (i.e., review and monitor design defects), rather than their intrinsic desire to do so.

Episode 3: Toward Problem Solving. Subsequently, the focus moved to addressing the problems identified to meet relevant control requirements. Over the years, the engineers at Project Alpha developed a routine to deal with and report engineering problems within one

week. Publicity associated with the checklist put engineers under each other's scrutiny and made them aware that lateness could have a snowball effect impacting the overall project. The senior technical manager also emphasized punctuality. As a result, punctuality became a buzzword to allow the project team to "have enough time to evaluate the manufacturability of product design, prepare components, and take action" (SMT engineer, Project Alpha).

A safe space for new behaviors. Clark left the engineers to perform their tasks. While the project was under a tight schedule, because the MRR was new, Clark relaxed many formal controls (i.e., tolerance for interim deviations, loose relationships between rewards/punishments, and performance) to accommodate necessary changes to current behavioral patterns. Engineers selectively responded to controls and adapted behaviors appropriate to emergent situations. The only time Clark intervened was when they lagged behind the agreed schedule or did not fill in the MRR properly. Even when Clark intervened, he did so through friendly emails or corridor talks rather than through formal disciplinary processes. Because some controls were relaxed (i.e., repair), a space emerged for engineers to adapt behaviors appropriate for emergent situations without fear of being punished (Figure 3, Arrow ④). "We did what we needed to do … Clark only occasionally intervened. Like when I missed some details or forgot to check off some items, Clark would send a reminder or chatted with me about it (product engineer, Project Alpha).

A collaborative culture to buttress the pursuit of self-goals. Norms and routines conducive to collaboration started to emerge (e.g., punctuality, respect for professional expertise). Due to the repeated enactment of transparency and repair, individual engineers understood that being late would create a snowball effect that would impact the project's schedule and the relationship between their tasks and others' tasks. Consequently, they contributed their expertise in a cooperative way. Collaboration buttressed the individual's setting and the pursuit of challenging self-goals. Since information was available and potential threats (e.g., schedule delay) could be spotted early, engineers were better able to plan actions: "…like the mechanical engineer needed extra time for getting a fixture. I had to get [a tool] and I estimated that it would take us a month to get it … issues were flagged in advance. If there's any difficulty, we negotiated. (product engineer, Project Alpha)

Also, the checklist, compiled from the Center's best practices, was applied as an overarching control to reduce control conflicts. The checklist reduced mistakes and reworking, saving engineers' energy and mental capacity.

Although most tasks were allocated a week for completion, engineers were intrinsically motivated to use their excess capacity to impose a more challenging time limit of "3 days" for addressing minor engineering problems (SMT engineer). This gave them more time and energy to "focus on addressing [real] problems" (product engineer, Project Alpha).

To request a change in the design, engineers first had to prove there was a design defect (faulty software or/and hardware design), which caused a product malfunction or manufacturing inefficiencies. To find one defect related to errors in the region code, engineers repeatedly tested the product in different ways. As a result of the repeated testing, designers acknowledged that this was a design defect (and not a defect of the testing process). Although designers failed to come up with a solution, both designers and engineers agreed that it was an "open issue, to-be-solved in the next [product] version" (product engineer).

Another defect was related to the design of a shielding cover. Engineers indicated that the one-piece design increased the risk to components when they had to remove the cover for inspection or maintenance purposes. Engineers provided evidence, showing that "one piece [out of five] was damaged during pilot production" (SMT engineer, Project Alpha). Engineers voluntarily came up with a two-piece design with a removable cover and helped source suppliers for the new component. Designers thus had a healthy respect for engineers and considered them "real experts" with whom the designers would willingly work (designer, Project Alpha). Overall, because collaboration was continually facilitated, a collaborative culture emerged, with engineers across the team being able to set challenging self-goals and competently pursue these goals (Figure 3, Arrow ⑤).

Project outcome. The outcome of Project Alpha was positive. First, the project was completed two weeks earlier than planned. The client was impressed by the team's detailed interim reports and product quality. Second, engineers felt empowered to identify and resolve problems. They voluntarily uncovered and identified the causes of some problems early and helped to solve these problems with designers. The project manager and designers thus attributed project success partially to engineers' "timely identification of design defects" (project manager, Project Alpha).

4.2 Project Beta

Project Beta was commissioned by a Fortune 500 company that was a repeat customer. The project involved the design and manufacture of a mature product that used an old technology. A junior product engineer was charged with overseeing its pilot production and was informed that they had seven weeks to complete the task.

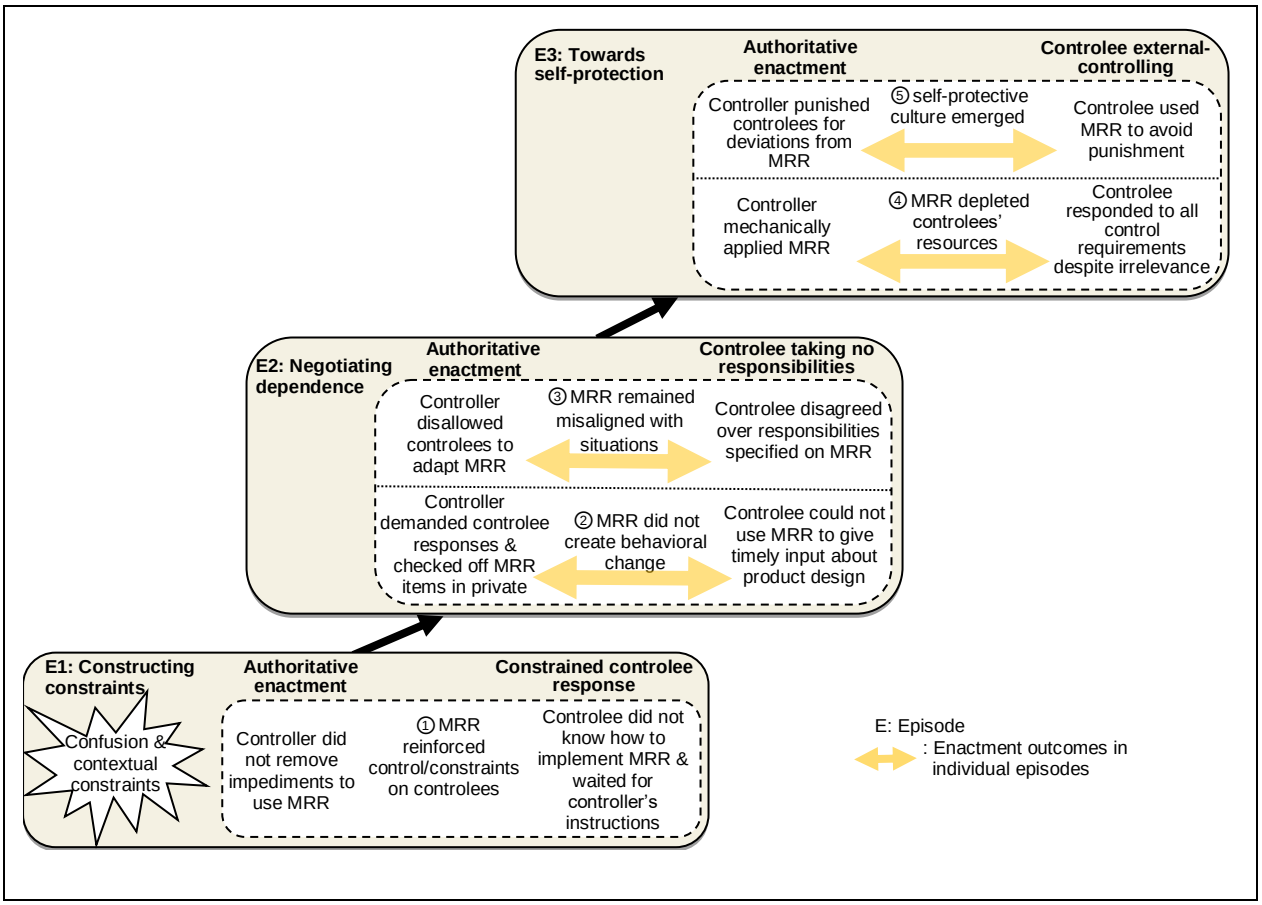


Figure 4. How Formal Control was Enacted to Impede Self-Control in Project Beta

Episode 1: Constructing Constraints. Like Project Alpha, there was a gap between what the MRR mandated and the actual work required on the project. However, during the lab pilot run, the junior product engineer took no action to clarify or circulate information about the MRR for engineers and other project members.

MRR reinforced control/constraints on controlees. The junior product engineer considered the MRR as another administrative hassle engineers had to face. He thus enacted the MRR as just one of many controls to regulate engineers, which allowed him to "[keep] an eye" on individual engineers and to record their progress. Designer domination was also obvious in Project Beta, with designers hoarding project information and engineers acting as passive information receivers.

The junior product engineer was informed about the project before engineers. Designers invited him to see a prototype. However, the junior product engineer did not update other engineers on the project status, nor did he make clear when and how the MRR would be applied. He kept the MRR to himself and assumed most associated administrative functions, including checking off items on engineers' behalf. Engineers became quite confused about "where the MRR stood" and "how it should be implemented" (mechanical engineer, Project Beta). They waited for instructions about the MRR implementation.

The project manager was also kept in the dark about the MRR and did not see the MRR as a project management tool. Instead, she viewed the MRR as a way for the Center to exert control on engineers. She used a separate checklist to assess product design. In Project Beta, designers were their own auditors, and engineers were not allowed to give input. Designers did not even hear about the MRR. This cut the engineers off from product analysis activities (Figure 4, Arrow ①). Overall, because the junior product engineer did not remove impediments (i.e., confusion, designer domination), transparency was inhibited, leading to engineers' misunderstanding of controller goals and the MRR.

Episode 2: Negotiating Dependence. When the engineering pilot run started, engineers only received project information from designers about routine production checks.

MRR did not create behavioral change. The junior product engineer queried engineers about their progress and checked off items in private, without sending them the checklist. However, without more

knowledge about the MRR, engineers could not use the MRR to give timely input about the product. They fell back on their usual routine checks and took no responsibilities for product auditing. Many items in the MRR were left with remarks, such as "under review" or "to be done by [a date]" (product engineer).

The junior product engineer segregated the MRR according to different functions and informed engineers of isolated tasks they were responsible for. Engineers had few opportunities to physically access the MRR. Engineers did not know how their behaviors impacted others or the overall project. Due to a lack of transparency, they performed the same behaviors on this project that they normally performed on other similar projects. Thus, at this stage, the MRR did not result in a change in engineer behavior (Figure 4, Arrow ②).

MRR remained misaligned with situations. The only time an engineer saw the checklist was when the junior product engineer noticed that "something serious happened" (junior product engineer, Project Beta) and felt the engineer in question was responsible for the problematic item. At that point, the engineer received a copy of a part of the checklist with problematic items highlighted. The junior product engineer would also carbon copy the engineer's direct supervisor to ensure compliance.

When conflicts over responsibilities for MRR items occurred or the relevance of MRR items to the product was raised, the junior product engineer was the one who determined who was responsible for the problematic item or whether an item was relevant.

This led to disagreements and perfunctory performance for some items. Engineers would perform checks to uncover problems and come up with temporary solutions. Due to a lack of repair, there remained a gap between the MRR and the situations engineers faced (Figure 4, Arrow ③).

> *I didn't think [one task] was my responsibility. It should have been under the purview of the product engineer… we should have discussed about this, but we didn't. Up to today, I still disagree that it should be my responsibility* (SMT engineer).

Episode 3: Toward Self-Protection. At the onset of the pilot production run, five potential design errors and a dozen production issues were identified, based on limited information from designers. In this episode, a decisive change in engineers' behavior occurred. Instead of applying the MRR to root out issues, the engineers applied it to absolve themselves of responsibility for problems.

MRR depleted controlees' resources. First, engineers sought the permission of the junior product engineer to reschedule solutions for several production issues. They all agreed to ensure that issues were resolved by the last week of pilot production. To ensure compliance, the junior product engineer strengthened his control through frequent reviews of engineers. In the last week of pilot production, he queried engineers about their progress every day. As controls and constraints increased, engineers began to feel overstretched.

> *I had many on-going projects and each of them also had to meet our departmental KPIs [key performance indicators], like overall productivity, MOH [manufacturing overheads] … we're asked to help each other out, but really had no energy left. We're totally overstretched.* (industrial engineer, Project Beta)

The junior product engineer required that many checks in the MRR be performed exactly as specified, regardless of the on-the-ground relationship between those tasks and project achievement. For example, the MRR mandated engineers to perform two checks: one a week before, and the other a day before each pilot production run. The junior product engineer would then check on the exact time the task was performed and the process taken. These checks required that resources that were not forthcoming be made available to engineers. For example, the assembly lines the engineers were supposed to use were often booked by others.

To overcome these constraints and to satisfy the immediate demands made by the junior product engineer, engineers would perform perfunctory checks or even "make a guess" (industrial engineer, Project Beta). Engineers devoted energy to address issues in the MRR instead of addressing concerns in production itself (Figure 4, Arrow ④). The mechanical application of the MRR (no repair) led to controlees' depletion of capacity.

A self-protective culture emerged. Second, designers rejected engineers' requests for design changes without any explanation. This showed designers' lack of respect toward the engineers. Because of designers' mistrust of engineers, they discounted engineers' opinions, even responding to engineers with sarcasm.

> *[In a meeting] our people told [a designer] that the 0.5mm distance between two pads was far too small. It would be better if it were over 0.8. The designer then replied derisively, saying that it's the global trend to produce smaller and more sophisticated gadgets* (mechanical engineer, Project Beta).

Engineers "could only suggest changes to the designer" (mechanical engineer, Project Beta), but received a disproportionate amount of blame when things went

wrong. Therefore, engineers used the MRR as a document to avoid blame and disclaim responsibility during an audit (i.e., unintended by the controller). For example, after his request for a design change was refused, an engineer asked the junior product engineer to put a remark reading "design couldn't be changed" and "[designer's name] as the PIC [person in charge]" (SMT engineer, Project Beta) in the checklist. Overall, disproportionate punishment led to controlees' enacting self-protective behaviors (Figure 4, Arrow ⑤).

Project outcome. The outcome of Project Beta was not so positive. First, Project Beta was completed with a pilot production yield rate of 88.54% and several design errors were unaddressed. Although the rate surpassed the requested threshold (85%), engineers were not satisfied because this was a mature product. Second, mechanical execution of the MRR reinforced engineers' sense of subjugation. They were closely monitored by the product engineer, given "no power to block the [problematic] product" (SMT engineer, Project Beta), and blamed for problems about which they "could do nothing" (mechanical engineer, Project Beta). Engineers felt unfairly exploited. The Appendix summarizes the temporal sequence of controller/controlee (inter)action in each project.

5 Discussion and Implications

This study aims to advance our understanding of control enactment in the complex IT project context. To achieve this, we studied the emerging process surrounding an enacted formal control and, specifically, how the enactment process could lead to controlee self-control. Our findings suggest the importance of the enactment of both transparency and repair for controlees' understanding of controller goals and the building of a trusting/collaborative culture (i.e., clan control). In Episode 1 of Project Alpha, because of the repeated enactment of transparency (e.g., two-way communication), engineers understood controller goals and means to deliver results (i.e., via early involvement in product analysis). However, expectations were formally conveyed or relayed (e.g., to designers) in formal meetings or in the name of the controller and/or project manager. The engineers thus had no personal attachment to assigned tasks. In Episode 2, because the controller repeatedly allowed for repair (e.g., controller's relaxing of other controls), engineers could derive personal goals and meaning from their tasks. Further, given the complexity of some issues (e.g., complex design errors), they could not perform effectively without others contributing their energy, ideas or time toward reaching or exceeding controller goals. Therefore, it was not until Episode 3 when both transparency and repair became rooted in the control context and collaborative norms were reinforced (e.g., information sharing) that engineers felt genuinely confident about directing their own activities (e.g., expediting routine checks) to work toward their self-goals and controller goals. In contrast, transparency and repair were inhibited from the outset of Project Beta. Repeated misunderstanding and mistrust led engineers to engage in self-protective behaviors.

5.1 The Role of Transparency and Repair in Reinforcing Cycles

Figure 5 integrates theory and our empirical results to demonstrate how controllers can enact the same formal control, but because they employ distinct control styles, they create divergent controlee responses and reinforcing cycles. In a complex project situation, it is impossible for the controller to anticipate all possible circumstances and design formal controls accordingly. New controls introduced to change controlees' behaviors thus must be amended as situations emerge ("situational contingency-formal control gap"). Within a project, there are some tasks where controlees know the appropriate behaviors for completing their tasks, exact outcomes to be achieved, or exact means-ends connections (*clear tasks*), and some tasks where controlees do not know the appropriate behaviors, outcomes to be achieved, or means-ends connections (*uncertain tasks*) (March & Simon, 1958). *Clear tasks* provide opportunities for controlees to perform appropriate behaviors for that specific project situation and to infer and validate their understanding of controller goals.

The combination of clear tasks, empowerment, and the controller providing feedback on task performance allowed controlees to correctly interpret project situations and understand controller goals. For example, in Project Alpha, engineers were told to use the MRR to audit product design and knew that effective auditing required early participation in product analysis. Because they could adapt the MRR according to controller feedback to complete their tasks, they were able to understand and confirm these goals. In contrast, when tasks are unclear, and the controller hoards information and does not provide feedback, controlees become confused and avoid responsibility (to avoid punishment). Distinct styles of control enactment by the controller thus trigger separate reinforcing cycles.

In our theorization, we identified two principal elements in formal control enactment, namely transparency and repair. It is the repeated enactment of the two elements by the controller that enables the development of controlees' capacity and motivation to exercise self-control in response to emerging situations.

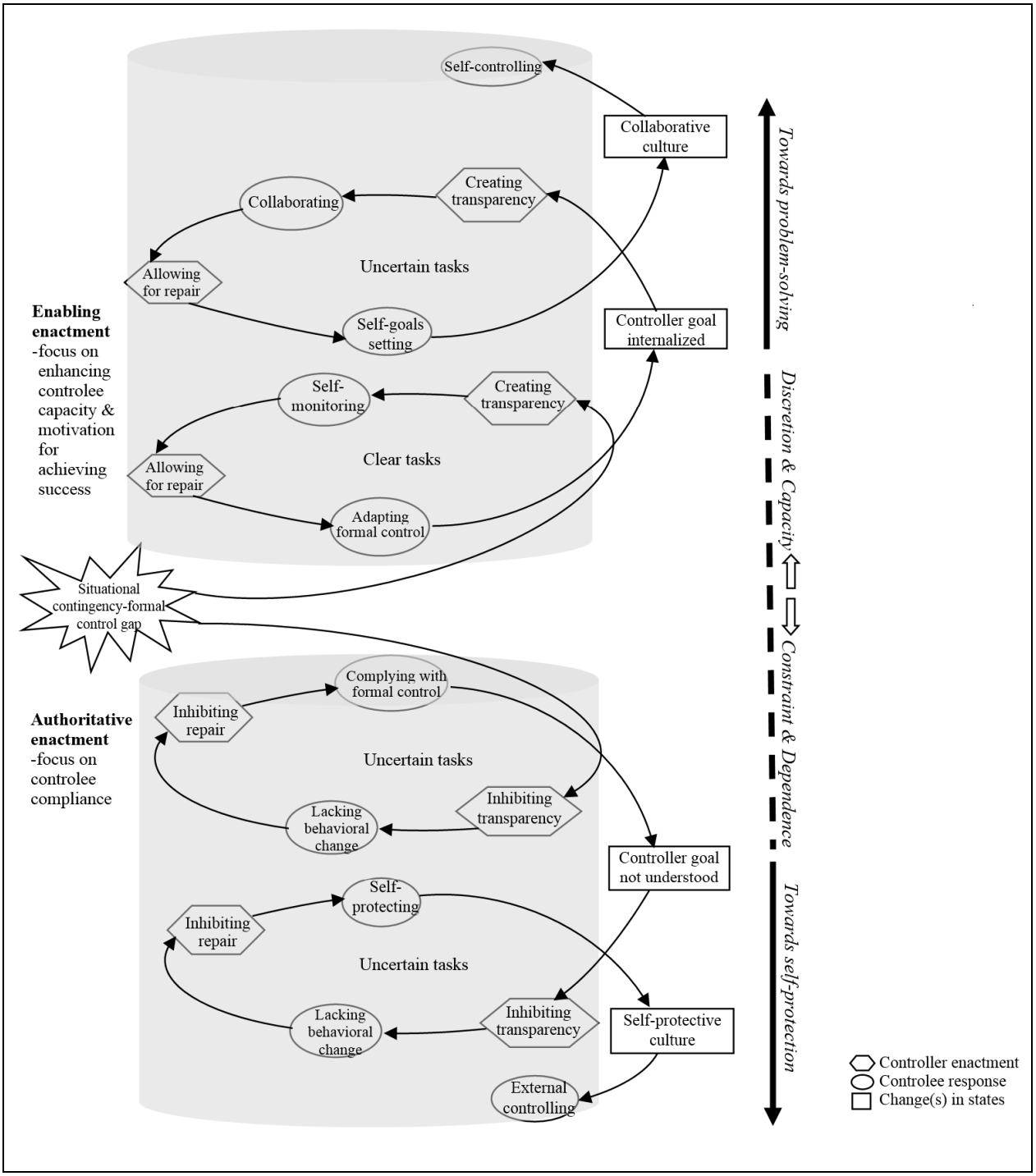


Figure 5. Reinforcing Cycles of Formal Control Enactment in Complex IT Projects

Creating transparency involves the controller's crafting of two-way communication channels (and lifting constraints on those channels) to clarify goals/requirements and to solicit continuous feedback. Thus, controllers who are open with controlees can send clear signals about their motives and create an implicit promise that controlees' feedback and opinions will be appreciated. Because controlees repeatedly witness the communication and action based on such implicit promises (e.g., controllers ask for and follow up on controlees' feedback), they can make correct inferences about the controller's expectations and feel they can come forward with any issue (Vogelgesang, Leroy, & Avolio, 2013). For example, in Episode 1 of Project Alpha, Clark spent time explaining to engineers the objective of the MRR and discussed the means to deliver those objectives (e.g., early involvement in product analysis). He explicitly told them he wanted feedback about improving the manufacturing process. This gave engineers the opportunity to involve themselves in

design decisions. Clark thus understood the constraints faced by engineers and helped alleviate them. Because of the communication, the engineers could determine which behaviors would help achieve controller goals and then chose to perform them, monitoring their own progress. In contrast, *inhibiting transparency* involves the controller hoarding information from controlees. The controlees thus cannot make correct inferences about controller goals and may thus distrust the controller.

In some cases, formal controls can malfunction, causing inappropriate behavior to arise, especially if a changing situation requires changed behavior. If controlees perceive they are likely to be punished for emergent, innovative behaviors appropriate to changing circumstances, it is unlikely they will perform such behaviors. Thus, controllers must create a safe space for controlees to try new behaviors. *Allowing for repair* involves relaxing formal control to allow controlees to take the initiative to try behaviors they otherwise would not. In contrast, *inhibiting repair* involves the use of punishment to strengthen control on controlees. Repair allows for two kinds of controlee behaviors. First, it allows for behaviors that formal control might unintentionally suppress. For example, one initial formal control exerted was assigning certain tasks to individual controlees. However, complex problems could have multiple causes (e.g., hardware incompatibility, software bugs) and the individual formally assigned to a role may not be appropriate for it. Hence, the more appropriate behavior is for controlees to take collective ownership. In Project Alpha, one MRR item was assigned to an engineer to check a design feature. The engineer explained to the controller that this would actually impede dynamic problem diagnosis. Several engineers thus jointly examined the design feature from their respective perspectives.

Second, repair allows emergent behaviors to develop to address unanticipated issues. Research has demonstrated that professionals often genuinely want things to work and to make a difference (Alvesson & Kärreman, 2007). If given the space, they will enact behaviors concordant with a desire to be responsible. In Episode 3 of Project Alpha, for example, because of the way control was enacted in prior episodes, engineers understood their new responsibility (i.e., product auditing) in relation to others and the overall project. They thus began to take on a broader range of activities beyond those initially prescribed by the MRR (i.e., testing product design, reporting design issues). These included challenging the existing design, proposing joint solutions of design issues, and monitoring project progress, something they had never done previously. We next detail separate reinforcing cycles triggered by distinct control styles and controlee responses as illustrated in Figure 5.

In Episode 1 of Project Alpha, because of transparency, controlees began to use the MRR to acquire product information from designers and to seek designer feedback; and in Episode 2, because of repair and the continuous creation of transparency, the controlees could suggest informed changes to the MRR, and selectively test the relevant design features specified in the MRR without the fear of being punished. As a result, controlees come to know controller goals and the means-ends relationships, constructing their *capacity* to handle their responsibilities. They also know they can adapt control, i.e., to negotiate *discretion* to complete their tasks.

As controlees witness greater transparency and then repair, they *internalize* controller goals (i.e., transform controller goals into their own), experience decreased vulnerability, and become confident in their own judgments. A positive reinforcing cycle is created. In Episode 3 of Project Alpha, the controller further relaxed controls to accommodate new behaviors for *uncertain tasks*. This leads to the creation of a safe space for controlees to judiciously apply their personal resources to improvise or experiment with new behaviors that are potentially beneficial to the project (e.g., collaborating behaviors). Furthermore, the coordination of collective efforts is required because of the inherent uncertainty of complex projects. In Episode 3 of Project Alpha, the controller facilitated collaboration by emphasizing information sharing and respect for expertise. *A collaborative culture* (i.e., clan control) emerged to support controlees to contribute their expertise in a cooperative manner. The engineers thus could improvise for joint problem solving and the pursuit of challenging goals beyond the MRR requirements (i.e., *self-goal setting*). In sum, transparency and repair together foster controlees' creative behaviors toward *problem solving* to benefit both the controller and controlee (i.e., *self-controlling*). For example, in addition to reporting design errors, engineers also proposed a new design and helped source new components to lower production risks. This improved not only product design but also production performance, leading to high performance.

Negative reinforcing cycles. Project Beta illustrates what happens when formal control is enacted without transparency and repair. Without transparency, controlees are unsure of controller goals and what appropriate behaviors would be. The most likely response in this situation (as observed in Project Beta) is for controlees to perform behaviors in alignment with the status quo (i.e., *lacking behavioral change*), with formal control becoming *constraints* on controlees. Thus, engineers in Episode 1 of Project Beta remained passive and allowed designers to run the project. Of course, this need not be the only possible controlee response. However, in complex IT projects, the likelihood of a controlee who does not understand

controller goals responding in a way consistent with controller goals is low.

When new controls are introduced, there is a general expectation that controlees' behaviors will change. When controlees are not performing new behaviors anticipated by an authoritative-style controller, the controller enacts punishment (i.e., *inhibiting repair*). In Episode 2 of Project Beta, the controller demanded engineers execute pre-specified checks under all circumstances and threatened to punish them for deviance. Without repair, controlees are unwilling to risk adaptive behaviors and are not able to learn which behaviors are effective for delivering desired results. Because controlees *do not understand controller goals* and distrust the controller, they feel that they are forced to *comply with formal control*. They develop *dependence* on the controller and perform tasks as they are told to. Because those tasks are expectable, controlees are still able to attain acceptable performance.

However, the consistent lack of transparency and repair creates a negative reinforcing cycle, with a sense of a lack of safety and incompetence emerging. This exacerbates misunderstanding and distrust among the project team. When situations emerge, controlees would like to perform appropriate behaviors (at least to avoid punishment). However, because they do not know what these behaviors are (only knowing existing behaviors which are ineffective for delivering the outcome), this drains their personal resources and induces dysfunctional behaviors. The controller's overemphasis on punishment could further induce *a self-protective culture* where controlees focus on minimizing loss to themselves rather than try to ensure the project proceeds successfully (i.e., failed clan control). In Episode 3 of Project Beta, an engineer identified some design errors and reported them to the designer. Because the designer refused to change the design, to protect himself, the engineer asked the controller to record the designer as responsible for associated production risks. The engineer then left the issue unresolved. Indeed, authoritative enactment can exacerbate conflicts in goals, and induce controlees' creative behaviors for achieving outcomes not harmful to themselves (i.e., *external controlling*), eventually resulting in low performance.

5.2 Relationship between Formal Control and Self-control

Our study explains why enacting control in an enabling style facilitates self-control to close the situational contingency-formal control gap for the controller's and controlee's benefit, while an authoritative enactment prompts controlees to exploit the gap for self-protection. Recall that self-control requires three elements, self-goals, self-monitoring, and intrinsic motivation.

Self-goals. Our results show that an enabling enactment helps controlees align their self-goals with controller goals for two reasons. First, *transparency* allows controlees to understand controller goals. Enabling-style controllers apply controls to explain to controlees why controller goals are in place, and why these controls are enacted (i.e., transparency). In Episode 1 of Project Alpha, the enabling-style controller explicitly highlighted the MRR's development intention (i.e., to identify design issues) and made it accessible to controlees. He used it to demonstrate how individual tasks were a part of controller goals (e.g., via the temporal arrangement of worksheets). Second, *repair* allows controlees to derive personal meaning and purpose from tasks. The enabling-style controller highlights to controlees when controls should be ignored or adapted for individual tasks. Controlees thus adapt controls and/or their behaviors and see how they contribute to controller goals. In Episode 2 of Project Alpha, the controller allowed controlees to adapt the prescribed working pace (i.e., one week for completing tasks). Controlees differentiated between routine and nonroutine components of their tasks and quickly completed the routine component to expedite the overall process. However, in Project Beta, the authoritative-style controller disallowed adaptation of controlees' work arrangement and instead used the MRR as a stick to beat nonperforming controlees. Controlees thus performed tasks perfunctorily and used the MRR to absolve them from their responsibilities.

Self-monitoring. Our results show that an enabling enactment facilitates self-monitoring for two reasons. First, *transparency* exposes controlees to project-relevant information and performance feedback. This helps controlees to improve both their individual and collective performance. In Episode 1 of Project Alpha, the controller enacted the MRR to help controlees easily access (via a shared server) and navigate through diverse information (by emphasizing the temporal connection of tasks). Controlees thus knew they needed to get involved early for effective product auditing. Given the opportunity, controlees then chose to demand designers' release of product information and provide their analysis to designers as early as possible. Second, *repair* allows controlees to try new behaviors to monitor and improve performance. In Project Alpha, the controller enacted the MRR to facilitate controlees' independent judgments with input/feedback from others. Controlees could thus adapt or synthesize means to monitor and conduct their tasks (e.g., examining product features from multiple perspectives). In contrast, in Project Beta, because the controller punished new behaviors, controlees chose not to adapt their behaviors, but relied on the controller's guidance and evaluation.

Intrinsic motivation. Although professionals are initially self-motivated, they can only express this if they feel safe (Alvesson & Kärreman, 2007). Our results show that an enabling enactment improves controlees' motivation to succeed for two reasons. First, *transparency* fosters trusting relationships and increases controlees' perceived odds of success. Enabling-style controllers are open with their intent and make relevant information available for everyone's consideration. Information thus reduces dependence and allows controlees to cope effectively with less optimal situations. In Project Alpha, the enabling-style controller openly shared information (Episode 1) and fostered norms for open communication and collaboration (Episode 3). Controlees thus felt confident in pursuing higher goals and making independent judgments while collaborating with others. Second, *repair* relaxes controls, which in turn acts to conserve controlees' personal resources (e.g., cognitive/emotional resources) for coping with emerging problems. Personal resources are essential for supporting controlees' feelings of competence and autonomy (i.e., intrinsic motivation) (Ryan & Deci, 2000a). In Episode 3 of Project Alpha, the controller reduced redundant controls and tolerated temporary deviations from the MRR. Controlees thus experienced limited fear of failure and could adapt the MRR as they considered appropriate (e.g., for expediting routine checks). In contrast, in Project Beta, the authoritative-style controller applied the MRR to punish deviance under all circumstances. This drained the controlees' resources and motivation because they had to spread out their efforts to manage all controls and were held accountable for failures over which they had little influence.

Together, transparency and repair foster trusting/collaborative relationships, and controlees' understanding of (1) the connection between controller goals and their self-goals, and (2) means for achieving both goals, leading to controlees' increased motivation and capacity to succeed. In contrast, the lack of transparency and repair introduces misunderstanding and mistrust, prompting controlees to embrace self-protective behaviors.

5.3 Contributions to Behavioral Control Theory

Our research contributes to behavioral control theory in several important ways. First, this study explains the role of control enactment in facilitating self-control. Chua et al. (2012) argued that clan control is fostered in projects through the use of formal control. We extend that thinking by arguing that controllers can facilitate controlee self-control, specifically by an enabling enactment of formal control to promote a safe space for personally risky behaviors (Park, Im, & Keil, 2008) and a collaborative culture (i.e., clan control). This suggests that researchers should examine control enactments in various styles, with the control style having a strong interactive effect (Keil et al., 2014; Wiener et al., 2016). While our study supports the effects of control enactment on controlee self-control via the promotion of clan control, we do not examine the interplay among the three control modes. Future research may examine how self-control interacts with formal and/or clan control (Huber et al., 2013).

Second, this study is a response to a call by Wiener et al. (2016) for studying control enactment. There is much that we do not know about the effects of control enactment on people and processes in complex projects. Our study finds that control enactment effects can be reinforced by controlee response. Positive reinforcing effects occur when the controller repeatedly enacts control in an enabling manner to facilitate the development of controlees' capacity and motivation to succeed. As a result, controlees can exercise self-control to respond to emergent situations unspecified by formal control for benefiting both the controller and the controlee (as observed in Project Alpha). Formal control and controlee self-control are like stone and mortar for building a wall. Once mortar is used to fill irregular gaps in stone, a strong wall can be built to withstand all elements. In contrast, negative reinforcing effects occur when the controller enacts control in an authoritative manner to reinforce misunderstanding and mistrust to the point where controlees' motivation and capacity for achieving success are depleted. As a result, controlees embrace self-protective behaviors and exploit the situational contingency-formal control gap for self-protection only (as observed in Project Beta). This is like building a wall using only stone. Without mortar, wind can blow through the wall and the wall is not as stable. It would be wrong to simply conclude that enacting control in an enabling style is always desirable; however, just as you need mortar to build a high wall, we believe an enabling enactment is necessary for complex projects because controlee initiative is a prerequisite for success. Our study shows an enabling style allows controlees to more accurately sense and flexibly respond to emergent situations in complex projects (Goh, Pan, & Zuo, 2013). Given the increasing complexity of IT projects (e.g., IoT solutions), enacting control in an enabling style is likely to be necessary in more IT projects.

This study also highlights that controlees play an essential role in the controller-controlee dynamic: one cannot assume that controlees will behave in a wholly expected way when a particular control is enacted by a controller. This study demonstrates that although controllers attempt to enact controls to ensure controlees' performance, controlees are not necessarily driven toward behaviors beneficial to the overall

project. Instead, they can be driven to defensive behaviors, especially if their attention is narrowly directed toward individual task performance (Parker & Collins, 2010). This also supports the conjectured association between authoritative enactment and control distortion (i.e., controlees misunderstand controller goals and/or means to achieve said goals) proposed by Heumann et al. (2015) who say the authoritative control style could lead to higher control distortion because of the lack of transparency on the use of control.

This study shows the importance of fostering a collaborative culture to support controlee self-control across the project team. In complex IT projects, tasks are often interrelated and rely on collaboration for completion of other tasks. Prior studies have argued for the importance of norms of independence and individualism for promoting controlee self-control (Choudhury & Sabherwal, 2003; Wiener et al., 2015). However, the coexistence of independence and a collaborative culture is not paradoxical, if we consider that independence can be facilitated by collegial peers. In Project Alpha, engineers could pursue challenging goals because of peers who released more information and adhered to agreed schedules to prevent bottlenecks. Additionally, a collaborative culture is conducive for a clear separation of role/responsibilities (i.e., independence) because expectations are articulated and communicated. Finally, a collaborative culture encourages controlees to forgo immediate, self-interested behaviors in favor of acts based on broader and long-term considerations. In Project Alpha, engineers collaborated with designers to solve design errors, instead of covering up the problems or pushing them on to designers. A collaborative culture, in turn, can be fostered by controllers showing respect for expertise and facilitating information exchange. Within Project Beta, an authoritative enactment caused controlees to compete for resources (e.g., facilities) and take refuge in defensive efforts, reinforcing mistrust that was dysfunctional for self-control.

Finally, our research highlights the importance of controlees' capacity to work as an important consideration in enacting control. Individual controlees' capacity to work is a limited resource (Baumeister et al., 1998). The execution of self-goals and self-monitoring requires controlees to tap into and consume that capacity to work (Muraven & Baumeister, 2000). Complex projects can quickly deplete the capacity to work because of task complexity and the need to switch mindsets in an evolving context (Hamilton et al., 2011). In Project Beta, merely complying with the demands of all controls overstretched the engineers. This left them with no energy to pursue goals beyond their narrow, formal responsibilities. However, in Project Alpha, extra support/resources and easy access to project information helped increase/conserve engineers' capacity. When permitted to adapt to the environment, the engineers applied the excess capacity to align their self-goals with the controller goals and self-monitor their progress. Our case thus suggests that controlee self-control may be stronger when controllers enact control to conserve/replenish controlees' capacity to work.

5.4 Managerial Implications

Managers have been given advice on which control to enact according to the characteristics of the context (Kirsch, 1997; Nidumolu & Subramani, 2003). However, we theorize that controlees play an essential role in control enactment, and there is often a gap between control and emerging situations in complex IT projects. Controllers need to interact with controlees in an enabling manner to facilitate controlee self-control to close this gap. Our results demonstrate that an enabling enactment helps enhance controlees' capacity and motivation for achieving success and addressing emergent issues in complex projects. In return, the enhanced capacity and motivation increases the effectiveness of control enacted. The interplay between control enacted and controlee self-control contributes to superior outcomes of complex IT projects.

A key question is: How does the controller start the positive reinforcing effect of formal control enactment to facilitate self-control or stop the negative reinforcing effect once it begins? To get the controller and controlee interacting to positively reinforce control enactment, it may be useful to create opportunities for interaction that go beyond command and control. For example, controllers can regularly discuss emerging situations with controlees and support individual and collective problem solving. This makes both controllers and controlees aware of the inadequacies of formal control and allows formal control to structure controlees' work in a way that still enables the exercise of self-control (Molnar, Nandhakumar, & Stacey, 2017). Moreover, while it is encouraging that the controller can enact transparency and repair to facilitate self-control, an ad hoc approach cannot be expected to sustain the positive reinforcing effect or reverse a negative one. It is the repeated controller-controlee interaction that enables the controlee's accumulation of capacity and motivation for exercising self-control.

5.5 Limitations

Several limitations of this study should be noted. First, this study relies on retrospective accounts of new product development projects. We addressed this limitation by using multiple interviewees and by collecting data within one month after the projects were completed. We also collected time-stamped

control documents for triangulation of interview data. Second, we do not claim that self-control is the only way to bridge situational contingency/formal control gaps. Several studies have suggested that formal control can be complemented by other factors, such as boundary-spanning activities (Gopal & Gosain, 2010), trust (Das & Teng, 1998), or group norms and individual reputation (Gallivan, 2001). More research should be done to identify the processes through which formal control can be complemented to improve project outcomes. Third, this study is based on a relatively small number of cases in a specific context. However, while our research may not be generalizable to a larger population, it is likely generalizable to theory (i.e., the process model in Figure 5) (Lee & Baskerville, 2003).

6 Conclusion

Formal control cannot be sufficiently prespecified for controlling complex IT projects. Controlee self-control is needed to address unanticipated issues. This research explores the enactment of a formal control (i.e., a checklist) on two complex IT projects in a wireless manufacturing organization. We found that how a control is enacted is critical to understanding whether controlee self-control will emerge to bridge the gap between the initial control and emerging situations. The key finding is that an enabling enactment facilitates controlee self-control by creating a safe space and a trusting/collaborative culture for independent actions of benefit to the overall project. In contrast, an authoritative enactment induces dysfunctional behaviors among controlees who seek to exploit the gap for their own protection. Repeated controller-controlee interaction contributes to the reinforcing effects of control enactment on project outcomes.

Finally, one could argue that controllers may adopt different control styles toward different controlee groups within a project (e.g., designer vs. engineer) (Soh, Chua, & Singh, 2010). This might be true. However, the focus is on how the control style shapes controlee behavior. This study suggests that controller behaviors predispose controlees to behave in a particular way. Future studies may be carried out to examine the interaction between the controller and different controlee groups.

One could also argue that the superior outcomes of Project Alpha arose from differences in controller seniority (senior technical manager) and/or client relationship (new client), because the senior controller had more power to enact effective control and the new client would attract more controller attention. However, based on the results of this study, project outcomes are not necessarily associated with the actual resources commanded by the controller (e.g., power, time, workforce). In Project Beta, regardless of a relaxed schedule and workforce (7 weeks and 9 engineers vs. 4 weeks and 7 engineers in Project Alpha), the controller chose to enact control authoritatively and invested energy/time blocking information flow and scrutinized controlees to the point that resources were exhausted and mistrust was reinforced, leading to below-par project performance. The importance of the structural condition is somewhat unclear. This suggests that additional studies in which control is enacted by controllers having a similar power status would be helpful.

References

Adler, P. S., & Borys, B. (1996). Two types of bureaucracy: Enabling and coercive. *Administrative Science Quarterly, 41*(4), 61-89.

Ahern, T., Leavy, B., & Byrne, P. J. (2014). Complex project management as complex problem solving: A distributed knowledge management perspective. *International Journal of Project Management, 32*(8), 1371-1381.

Ahrens, T., & Chapman, C. S. (2004). Accounting for flexibility and efficiency: A field study of management control systems in a restaurant chain. *Contemporary Accounting Research, 21*(2), 271-301.

Alvesson, M., & Kärreman, D. (2007). Unraveling HRM: Identity, ceremony, and control in a management consulting firm. *Organization Science, 18*(4), 711-723.

Alvesson, M., & Willmott, H. (2002). Identity regulation as organizational control: Producing the appropriate individual. *Journal of Management Studies, 39*(5), 619-644.

Ashforth, B. E. (1997). Petty tyranny in organizations: A preliminary examination of antecedents and consequences. *Canadian Journal of Administrative Sciences, 14*(2), 126-140.

Baccarini, D. (1996). The concept of project complexity-A review. *International Journal of Project Management, 14*(4), 201-204.

Barker, J. R. (1993). Tightening the iron cage: Concertive control in self-managing teams. *Administrative Science Quarterly, 38*(3), 408-437.

Barley, S. R. (1986). Technology as an occasion for structuring: Evidence from observations of CT scanners and the social order of radiology departments. *Administrative Science Quarterly, 31*(1), 78-108.

Baumeister, R. F. (2002). Yielding to temptation: Self-control failure, impulsive purchasing, and consumer behavior. *Journal of Consumer Research, 28*(4), 670-676.

Baumeister, R. F., Bratslavsky, E., Muraven, M., & Tice, D. M. (1998). Ego depletion: Is the active self a limited resource? *Journal of Personality and Social Psychology, 74*(5), 1252-1265.

Baumeister, R. F., & Heatherton, T. F. (1996). Self-regulation failure: An overview. *Psychological Inquiry, 7*(1), 1-15.

Baumeister, R. F., & Leary, M. R. (1995). The need to belong: Desire for interpersonal attachments as a fundamental human motivation. *Psychological Bulletin, 117*(3), 497-529.

Baumeister, R. F., Vohs, K. D., & Tice, D. M. (2007). The strength model of self-control. *Current Directions in Psychological Science, 16*(6), 351-355.

Braverman, H. (1974). *Labor and monopoly capital: The degradation of work in the twentieth century*. New York University Press.

Choudhury, V., & Sabherwal, R. (2003). Portfolios of control in outsourced software development projects. *Information Systems Research, 14*(3), 291-314.

Chua, C. E. H., Lim, W. K., Soh, C., & Sia, S. K. (2012). Enacting clan control in complex IT projects: A social capital perspective. *MIS Quarterly, 36*(2), 577-600.

Chua, C. E. H., & Myers, M. D. (2018). Social control in information systems development: A negotiated order perspective. *Journal of Information Technology, 33*(3), 173-187.

Cooper, C., & Taylor, P. (2000). From Taylorism to Ms. Taylor: The transformation of the accounting craft. *Accounting, Organizations and Society, 25*(6), 555-578.

Costas, J. (2012). "We are all friends here": Reinforcing paradoxes of normative control in a culture of friendship. *Journal of Management Inquiry, 21*(4), 377-395.

Crabtree, B., & Miller, W. (1999). A template approach to text analysis: Developing and using codebooks. In B. Crabtree & W. Miller (Eds.), *Doing Qualitative Research* (pp. 163-177). SAGE.

Das, T. K., & Teng, B.-S. (1998). Between trust and control: Developing confidence in partner cooperation in alliances. *Academy of Management Review, 23*(3), 491-512.

de Ridder, D. T., Lensvelt-Mulders, G., Finkenauer, C., Stok, F. M., & Baumeister, R. F. (2012). Taking stock of self-control: A meta-analysis of how trait self-control relates to a wide range of behaviors. *Personality and Social Psychology Review, 16*(1), 76-99.

Deci, E. L., & Ryan, R. M. (1987). The support of autonomy and the control of behavior. *Journal of Personality and Social Psychology, 53*(6), 1024-1037.

Dubé, L., & Paré, G. (2003). Rigor in information systems positivist case research: Current practices, trends, and recommendations. *MIS Quarterly, 27*(4), 597-636.

Eisenhardt, K. M. (1985). Control: Organizational and economic approaches. *Management Science, 31*(2), 134-149.

Eisenhardt, K. M. (1989). Building theories from case study research. *Academy of Management Review, 14*(4), 532-550.

Engwall, M. (2003). No project is an island: Linking projects to history and context. *Research Policy, 32*(5), 789-808.

Fleming, P., & Sturdy, A. (2011). "Being yourself" in the electronic sweatshop: New forms of normative control. *Human Relations, 64*(2), 177-200.

Friedman, A. (1990). Management strategies, techniques and technology: Towards a complex theory of the labour process. In D. Knights & H. Willmott (Eds.), *Labour process theory*. Macmillan.

Gallivan, M. J. (2001). Striking a balance between trust and control in a virtual organization: A content analysis of open source software case studies. *Information Systems Journal, 11*(4), 277-304.

Ghoshal, S., & Moran, P. (1996). Bad for practice: A critique of the transaction cost theory. *Academy of Management Review, 21*(1), 13-47.

Goh, J. C.-L., Pan, S. L., & Zuo, M. (2013). Developing the agile IS development practices in large-scale IT projects: the trust-mediated organizational controls and IT project team capabilities perspectives. *Journal of the Association for Information Systems, 14*(12), 722-756.

Gopal, A., & Gosain, S. (2010). The role of organizational controls and boundary spanning in software development outsourcing: Implications for project performance. *Information Systems Research, 21*(4), 960-982.

Govindarajan, V., & Fisher, J. (1990). Strategy, control systems, and resource-sharing: Effects on business-unit performance. *Academy of Management Journal, 33*(2), 259-285.

Grabski, S. V., & Leech, S. A. (2007). Complementary controls and ERP implementation success. *International Journal of Accounting Information Systems, 8*(1), 17-39.

Gravina, N., Austin, J., Schoedtder, L., & Loewy, S. (2008). The effects of self-monitoring on safe posture performance. *Journal of Organizational Behavior Management, 28*(4), 238-259.

Gregory, R. W., & Keil, M. (2014). Blending bureaucratic and collaborative management styles to achieve control ambidexterity in IS projects. *European Journal of Information Systems, 23*(3), 343-356.

Gulati, R., & Puranam, P. (2009). Renewal through reorganization: The value of inconsistencies between formal and informal organization. *Organization Science, 20*(2), 422-440.

Hamilton, R., Vohs, K. D., Sellier, A.-L., & Meyvis, T. (2011). Being of two minds: Switching mindsets exhausts self-regulatory resources. *Organizational Behavior and Human Decision Processes, 115*(1), 13-24.

Harter, S. (1978). Effectance motivation reconsidered. Toward a developmental model. *Human Development, 21*(1), 34-64.

Hayek, F. A. (1945). The use of knowledge in society. *The American Economic Review, 35*(4), 519-530.

Henderson, J. C., & Lee, S. (1992). Managing I/S design teams: a control theories perspective. *Management Science, 38*(6), 757-777.

Heumann, J., Wiener, M., Remus, U., & Mähring, M. (2015). To coerce or to enable? Exercising formal control in a large information systems project. *Journal of Information Technology, 30*(4), 337-351.

Hoy, W. K., & Sweetland, S. R. (2001). Designing better schools: The meaning and measure of enabling school structures. *Educational Administration Quarterly, 37*(3), 296-321.

Huber, T. L., Fischer, T. A., Dibbern, J., & Hirschheim, R. (2013). A process model of complementarity and substitution of contractual and relational governance in IS outsourcing. *Journal of Management Information Systems, 30*(3), 81-114.

Huberman, A. M., & Miles, M. B. (1998). Data management and analysis methods. In N. K. Denzin & Y. S. Lincoln (Eds.), *Collecting and interpreting qualitative materials* (pp. 179-210). SAGE.

Huebner, B. M. (2003). Administrative determinants of inmate violence: A multilevel analysis. *Journal of Criminal Justice, 31*(2), 107-117.

Jaworski, B. J. (1988). Toward a theory of marketing control: Environmental context, control types, and consequences. *Journal of Marketing, 52*(3), 23-39.

Keil, M., Smith, H. J., Iacovou, C. L., & Thompson, R. L. (2014). The dynamics of IT project status reporting: A self-reinforcing cycle of distrust.

Journal of the Association for Information Systems, 15(12), 879-912.

Kirsch, L. J. (1996). The management of complex tasks in organizations: Controlling the systems development process. *Organization Science, 7*(1), 1-21.

Kirsch, L. J. (1997). Portfolios of control modes and IS project management. *Information Systems Research, 8*(3), 215-239.

Kirsch, L. J., & Cummings, L. L. (1996). Contextual influences on self-control of IS professionals engaged in systems development. *Accounting, Management and Information Technologies, 6*(3), 191-219.

Kirsch, L. J., Ko, D.-G., & Haney, M. H. (2010). Investigating the antecedents of team-based clan control: Adding social capital as a predictor. *Organization Science, 21*(2), 469-489.

Kirsch, L. J., Sambamurthy, V., Ko, D.-G., & Purvis, R. L. (2002). Controlling information systems development projects: The view from the client. *Management Science, 48*(4), 484-498.

Kunda, G. (1992). *Engineering culture: Control and commitment in a high-tech corporation*. Temple University Press.

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

Latham, G. P., & Locke, E. A. (1991). Self-regulation through goal setting. *Organizational Behavior and Human Decision Processes, 50*(2), 212-247.

Lee, A. S. (1989). A scientific methodology for MIS case studies. *MIS Quarterly, 13*(1), 33-50.

Lee, A. S., & Baskerville, R. L. (2003). Generalizing generalizability in information systems research. *Information Systems Research, 14*(3), 221-243.

Lioliou, E., Zimmermann, A., Willcocks, L., & Gao, L. (2014). Formal and relational governance in IT outsourcing: substitution, complementarity and the role of the psychological contract. *Information Systems Journal, 24*(6), 503-535.

Ludwig, T. D., & Goomas, D. T. (2009). Real-time performance monitoring, goal-setting, and feedback for forklift drivers in a distribution centre. *Journal of Occupational and Organizational Psychology, 82*(2), 391-403.

Manz, C. (1986). Self-leadership: Toward an expanded theory of self-influence processes in organizations. *Academy of Management Review, 11*(3), 585-600.

Manz, C., & Sims Jr, H. P. (1987). Leading workers to lead themselves: The external leadership of self-managing work teams. *Administrative Science Quarterly, 32*(1), 106-129.

March, J. G., & Simon, H. A. (1958). *Organizations*. Wiley.

Maruping, L. M., Venkatesh, V., & Agarwal, R. (2009). A control theory perspective on agile methodology use and changing user requirements. *Information Systems Research, 20*(3), 377-399.

Miller, S. M. (1987). Monitoring and blunting: Validation of a questionnaire to assess styles of information seeking under threat. *Journal of Personality and Social Psychology, 52*(2), 345-353.

Mähring, M. (2002). *IT Project Governance*. The Economic Research Institute, Stockholm School of Economics.

Molnar, W. A., Nandhakumar, J., & Stacey, P. (2017). A paradox of progressive saturation: The changing nature of improvisation over time in a systems development project. *Journal of the Association for Information Systems, 18*(11), 814-836.

Muraven, M., & Baumeister, R. F. (2000). Self-regulation and depletion of limited resources: Does self-control resemble a muscle? *Psychological Bulletin, 126*(2), 247-259.

Nidumolu, S. R., & Subramani, M. R. (2003). The matrix of control: Combining process and structure approaches to managing software development. *Journal of Management Information Systems, 20*(3), 159-196.

Orlikowski, W. J., & Iacono, C. S. (Eds.). (2000). The truth is not out there: An enacted view of the "digital economy." In E. Brynjolfsson & B. Kahin (Eds.), *Understanding the Digital Economy* (pp.235-380). MIT Press.

Ouchi, W. G. (1979). A conceptual framework for the design of organizational control mechanisms. *Management Science, 25*(9), 833-848.

Ouchi, W. G. (1980). Markets, bureaucracies, and clans. *Administrative Science Quarterly, 25*(1), 129-141.

Park, C., Im, G., & Keil, M. (2008). Overcoming the mum effect in IT project reporting: Impacts of fault responsibility and time urgency. *Journal of the Association for Information Systems, 9*(7), 409-431.

Parker, S. K., & Collins, C. G. (2010). Taking stock: Integrating and differentiating multiple proactive behaviors. *Journal of Management, 36*(3), 633-662.

Prasad, P., & Prasad, A. (2000). Stretching the iron cage: The constitution and implications of routine workplace resistance. *Organization Science, 11*(4), 387-403.

Ryan, R. M., & Deci, E. L. (2000a). Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary Educational Psychology, 25*(1), 54-67.

Ryan, R. M., & Deci, E. L. (2000b). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist, 55*(1), 68-78.

Soh, C., Chua, C. E. H., & Singh, H. (2010). Managing diverse stakeholders in enterprise systems projects: A control portfolio approach. *Journal of Information Technology, 26*(1), 16-31.

Steinberg, L. (2005). Psychological control: Style or substance? *New Directions for Child and Adolescent Development, 108*, 71-78.

Tarafdar, M., & Tanriverdi, H. (2018). Impact of the information technology unit on information technology-embedded product innovation. *Journal of the Association for Information Systems*, *19*(8), 716-751

Tatikonda, M. V., & Rosenthal, S. R. (2000). Successful execution of product development projects: Balancing firmness and flexibility in the innovation process. *Journal of Operations Management, 18*(4), 401-425.

Tiwana, A. (2010). Systems development ambidexterity: Explaining the complementary and substitutive roles of formal and informal controls. *Journal of Management Information Systems, 27*(2), 87-126.

Tiwana, A., & Keil, M. (2009). Control in internal and outsourced software projects. *Journal of Management Information Systems, 26*(3), 9-44.

Van de Ven, A. H., & Poole, M. S. (1995). Explaining development and change in organizations. *Academy of Management Review, 20*(3), 510-540.

Vogelgesang, G. R., Leroy, H., & Avolio, B. J. (2013). The mediating effects of leader integrity with transparency in communication and work engagement/performance. *The Leadership Quarterly, 24*(3), 405-413.

Vohs, K. D., Baumeister, R. F., Schmeichel, B. J., Twenge, J. M., Nelson, N. M., & Tice, D. M. (2014). Making choices impairs subsequent self-control: A limited-resource account of decision making, self-regulation, and active initiative. *Journal of Personality and Social Psychology, 94*(5), 883-898.

Weick, K. E. (1995). *Sensemaking in Organizations*. SAGE.

Wiener, M., Mähring, M., Remus, U., & Saunders, C. (2016). Control configuration and control enactment in information systems projects: Review and expanded theoretical framework. *MIS Quarterly, 40*(3), 741-774.

Wiener, M., Remus, U., Heumann, J., & Mähring, M. (2015). The effective promotion of informal control in information systems offshoring projects. *European Journal of Information Systems, 24*(6), 569-587.

Wouters, M., & Wilderom, C. (2008). Developing performance-measurement systems as enabling formalization: A longitudinal field study of a logistics department. *Accounting, Organizations and Society, 33*(4), 488-516.

Yin, R. K. (2003). *Case study research: Design and methods*. SAGE

Yun, R., Scupelli, P., Aziz, A., & Loftness, V. (2013). *Sustainability in the workplace: Nine intervention techniques for behavior change.* Paper presented at the International Conference on Persuasive Technology.

.

Appendix

Table A1. Interaction Surrounding Control Enacted

	Category	Empirical evidence
Project Alpha		
E1	**Controller** • transparency • built communication channels • removed impediments	• Set up a shared server for disseminating project- and MRR information (with tasks' temporal connection emphasized) • Discussed the MRR (goals/means/history) with engineers in meetings • Negotiated with other controllers to remove designer oppression and reschedule other projects
	Controlee • understood controller goals • self-monitored	• Understood controller goals (i.e., product auditing) • Used the MRR to acquire product information from designers & started to analyze the product early
E2	**Controller** • repair: allowed independent action	• Allowed individual interpretation/adaptation of the MRR to settle disputes/problems
	Controlee • set self-goals • self-monitored	• Interpreted, applied, and amended the MRR as considered appropriate to situational contingencies • Applied the MRR to identify/analyze design problems
E3	**Controller** • repair: relaxed controls • facilitated collaboration	• Relaxed other formal controls (e.g., tolerance for deviations) • Fostered norms to facilitate collaboration (e.g., respect for expertise, punctuality, information sharing)
	Controlee • set challenging self-goals • self-monitored • conserved personal resources • developed intrinsic motivation: perceived competence	• Collaborated with others to pursue challenging goals • Monitored/solved problems unspecified by the MRR • Used the MRR to memorize details & expedite routine checks • Felt competent at design auditing
Project Beta		
E1	**Controller** • did not remove impediments • hid information	• Accepted structural constraints as given (e.g., designer domination, other controls) • Did not expose engineers to MRR-relevant information
	Controlee • lacked behavioral change	• Were confused about the MRR (i.e., means-ends relationship) and persisted with behavioral patterns that were ineffective for product auditing
E2	**Controller** • used commands to control • imposed interpretation	• Issued commands and demanded responses • Disallowed individual interpretation of the MRR regardless of situational contingencies
	Controlee • worked around control • monitored by others	• Disagreed over responsibilities and performed tasks perfunctorily • Relied on the product engineer for guidance and evaluation
E3	**Controller** • mechanically applied control	• Mechanically applied the MRR and punished engineers for what they had little influence on
	Controlee • developed extrinsic motivation • depleted capacity • a self-protective culture emerged	• Applied the MRR to avoid blame or waive responsibilities • Applied limited energy to respond to frequent review requests • Mistrusted peers (e.g., designers)

About the Authors

Gloria H. W. Liu is a lecturer (assistant professor) in the International Business School Suzhou, Xi'an Jiaotong-Liverpool University. Her research interests include digital technology and business transformation, project control and coordination, knowledge management, and enterprise systems implementation/upgrade.

Cecil Eng Huang Chua received a PhD in information systems from Georgia State University, a master's of business by research from Nanyang Technological University and both a bachelor's of business administration in computer information systems and economics and a masters certificate in telecommunications management from the University of Miami. Cecil has publications in six of the eight AIS Senior Scholars' Basket journals, including papers in *Information Systems Research* and *MIS Quarterly*. He is a senior editor for *AIS Transactions on HCI* and *Pacific Asia Journal of the Association for Information Systems*, desk editor for *Project Management Journal* and an associate editor for both *Information & Management* and *Information Systems Journal*. Cecil has consulted for a range of organizations including Daimler SEAsia, General Motors Singapore, the Singapore Ministry of Defense, and Fonterra, the New Zealand milk cooperative.