Designing Complex Software Implementation Programs

Piet M.A. Ribbers
Tilburg University
Department of Information Management
Tilburg, The Netherlands
E-mail: P.M.A.Ribbers@KUB.nl

Klaus-Clemens Schoo
PriceWaterhouseCoopers
Frankfurt, Germany
E-mail: klaus-clemens.schoo@de.pwcglobal.com

Abstract

The central question of this paper is: How can design of Program Management contribute to the success of complex software implementations? Incomplete goal specifications, lack of communication, and underestimation of project complexity are signs of insufficient program and project management. To avoid these pitfalls, we will propose ways that have worked well in recent complex multi-project ERP implementations ("programs"). Answering the above question raises the problem of adequately defining and measuring the level of complexity of an implementation program. We operationalize complexity through three dimensions: variety, variability, and integration. A conceptual model was developed, which identifies program management factors that are proposed to have an impact on implementation success. Fifteen cases were studied to find support for the proposed framework.

Key words: program management, project management, standard software, software implementation, success factors, complexity.

1. Introduction and Problem Specification

1.1. Program Management

Current competitive agendas in many industries require timely and flexible responses to changing market conditions. New business models based on process orientation [11], dynamic partnership relationships with trading partners [20], and concentration on core competencies [26] [27] emerge. Faster, more accurate and integrated information is a cornerstone in this effort. The implementation of an Enterprise Resource Planning (ERP) software package can help to enable or even drive the required changes. ERP software attempts to integrate all departments and functions across a company into a single computer system. It has the potential to integrate databases, data flows and systems, and to streamline operations and reporting (e.g. through implementing the same package at all sites). ERP packages offer a great variety of standard solutions for individual business problems.

However, in practice, a lack of effectiveness and efficiency in large package implementations is reported. Failures are attributed to unsatisfactory project management and control, incomplete goal specifications, lack of communication, and underestimation of project complexity [4]. These problems will affect any software implementation, they become the more severe, the larger and more integrated the implementation effort is as a whole.

Despite the popularity of ERP, broad-based empirical research on the success factors that impact implementation is still very limited [32] . Most of the literature has focused on project management and implementation [5]. Projects are a structured set of activities concerned with delivering a defined capability to the organization based on agreed schedule and budget. However, complex package implementations consist of a set of (sub-) projects, which are related regarding contents, resources, and goals, and pursue a common (strategic) goal, e.g. to improve the supply chain. The set of (sub-) projects are cast into a program. A program consists of a portfolio of projects, defining a set of related activities, both for the IT and the business side, that have defined goals and benefits, and need to be controlled as a whole. Implementations of ERP packages can be perceived as change processes that transform the initial organization into an envisaged, improved state, supported by the new IT as enabling factor. Program Management can be conceptualized as a controlling instance for a transformation process (i.e. the design, development, and deployment of changes to the organization and IT, following a result path [29]) that in turn is governed by projects. The implementation has been successful when the projects in the transition phase have been a success, i.e. when the envisaged changes not only to the IT but also to business processes have been achieved.

1.2. The research question

This paper is about making complex package implementations a success. Success is not only to technically implement or replace a system, but also to support the desired business changes.
The central question is: How can design of Program Management contribute to the success of complex software implementations?

Generic advice on program management exists in the literature on project management, e.g. [23] or on Information Management (under de heading of portfolio management, as in [17], and [6]). Case based advice exists in, e.g. [8] and [2]. This paper aims at bringing the conceptual and the case-based approaches together, and developing indications for how to organize program management for complex implementations.

1.3. Implementation Complexity

We are particularly interested in how implementation complexity affects program management. Is there a specific design, or are there specific design characteristics for program management to respond adequately to requirements posed by the complexity of ERP implementation programs? Answering this question raises the problem of adequately defining and measuring the level of complexity of an implementation program.

Based on [34], [3] and [15], we propose three measures of complexity: Variety, variability and integration. Variety refers to the number of different states a system can take. More concretely it reflects the number of elements and their interrelations in a given situation or system. In an implementation program, variety will increase with e.g. the number of sites affected, or the functions of a package implemented. Variability of a system relates to the dynamics over time of its elements and the interrelations between them. Examples of variability in an implementation program are scope changes, lack of resources, or dependencies on other programs that are competing for resources. Integration characterizes the planned changes to be realized through the implementation program in terms of integration of IT systems and across business processes. This measure describes the degree of innovation in IT and business processes that is to be implemented through the program. Innovation factors are, for instance, new co-operations with customers and suppliers, and the sophistication and newness of the technology to be introduced.

1.4. Research Approach

The research that formed the basis for this paper was conducted in two phases [30]. We reviewed relevant literature, interviewed experts and analyzed case studies. The literature review focused on Systems theory and Program Management. Systems theory contributed to the understanding of the effects of complexity, e.g. [22] [18]; Program Management literature contributed to the understanding of determinants of success of complex implementations. The main source for the latter was literature on Supply Chain Management implementation e.g. [5] [11], Software Package implementation e.g. [29] [14] [2], Multi-project management, e.g. [28] [23] [33], and Project control and IT efficiency studies e.g. [16] [25]. The literature study and expert interviews have resulted in a conceptual model, which identifies program management factors that are proposed to have an impact on implementation success. Fifteen cases were studied to find support for the proposed framework. Case studies were used to support analytical rather than statistical generalization [35], and because they enable “the capturing of reality in considerable greater detail and the analysis of considerable greater number of variables” [9].

All the case studies we analyzed were perceived as being successful implementations. The cases were chosen in accordance with our conceptual framework (see below). As a consequence, the cases do not represent a random sample. Despite a strong effort, we found it impossible to obtain cooperation of a sufficient number of complex cases representing a failure. Consequently, we have not been able to study whether the absence of a proposed program management factor contributed to failure. The empirical material gathered has to be seen in the light of offering support for the proposed program management factors. From a methodological point of view, it offers the possibility of falsification [24]: can a project be fully successful without the support of a proposed program management factor?

The outline of the paper is as follows. Section 2 discusses the conceptual framework. Section 3 provides a discussion on empirical evidence. Section 4 closes with conclusions.

2. Conceptual Framework

The conceptual model consists of three analytical aspects: Program Success, Implementation Complexity, and Program Management. We build on the Environmental School within Organization theory which focuses on the relationship between management and the environment. In particular we follow contingency theory, which explores the relationship between (independent assumed) situational or environmental variables and dependent structural variables. Besides uncertainty, complexity is very often discussed as a characteristic of organizational environment [21]. We capture complexity from the size and structure of the company in our measure variety. Dynamic complexity is captured in variability in that it measures the unplanned, uncontrollable variations. In case of high variability there is high uncertainty in decision making because of lack of knowledge about what the next situation will be like, e.g. on how stable the new technology will be. The planned variation is captured in integration. We expect implementation
complexity to have an effect on program management through each of the complexity dimensions, and we expect adequate program management to contribute to program success. This follows from the "congruency hypothesis" [21]. It states that those organizations whose structures match more closely the requirements of the environment will be more effective than those which do not.

In the following paragraphs we will develop operational measures for the three analytical aspects and more refined propositions regarding program management under different types of implementation complexity.

2.1. Program Success

Success of a package implementation pertains to the implementation process (project success – in particular delivery of the new system and procedures on time and within budget), and to its results (product success – in particular performance improvements achieved). We measured success through the following indicators, see e.g. [31] [19] [29]:

- Level of use of the new system and procedures;
- Level of contribution of the program deliverable to the company;
- Level of adherence to the time plan;
- Level of adherence to budget.

2.2. Implementation Complexity

As discussed in section 1, complexity of package implementations can be characterized through variety, variability and integration. We are interested both in the overall complexity of an implementation program, and its complexity level related to each of the individual dimensions. On the basis of literature and expert interviews we identified complexity indicators and relative weights of each indicator. Relative weights have been introduced because from the interviews it became clear that the relative importance of all the indicators is not the same. The proposed weights [on a scale 1 – 5] thus reflect the views of interviewed program managers on the relative impact of complexity variables.

For Variety we chose the following indicators:

- Number of affected locations (weight factor 3);
- Readiness in terms of organizational differences between locations, and the level of negative predisposition from experiences in earlier, similar programs (weight factor 1);
- Conversion effort in terms of level of data misfit and number of systems to be replaced (weight factor 2).

For Variability the following indicators were used:

- Level of availability of resources in terms of availability of adequately trained and experienced project staff (weight factor 4);
- Level of concurrent similarly complex programs (weight factor 2);
- Extent of system redesign after pilot (weight factor 1);
- Extent of goal and scope changes (weight factor 3).

Integration characterizes the planned changes to be realized through the implementation program in terms of the integration across business processes and IT systems as defined in section 2. Keen’s differentiation of reach and range for IT platforms provides an adequate framework for description [15]. IT integration primarily pertains to IT infrastructure. For organizational integration, the dimensions of reach and range find their counterparts in the organizational boundaries and processes in the organization. An increased reach and range goes together with (and supports an intended) crossing of organizational boundaries as business processes span more organizational functions. Thus we can describe technical and process integration on four axes:

- Reach, ranging from ‘within location’ to ‘all over the world’
- Range, ranging from ‘single, local support’ to ‘cooperative transactions’
- Process, ranging from ‘internal process’ to ‘external process’
- Organizational boundary, from ‘team internal’ to ‘external partners’

The axes characterize different levels of process flow capabilities within or between organizations (organizational networking):

- 1 intra-team
- 2 inter-team
- 3 intra-organizational (between organizational units, formed by teams)
- 4 inter-organizational.

Similar categories are used to differentiate levels of networking in CSCW [13].

Figure 1 illustrates the categories (used as ordinal scales) for the four axes.

The complexity variable integration is computed as the average deltas between the levels of integration on these axes before and after the program. It is transposed onto a scale between 1 and 5, to give the variable the same scale as the other two complexity variables.
2.3. Program Management

Program Management contains the elements [18]:

- Program Organization, as the structural organization of a team that plans and controls projects in the program and the related resources;
- Policies, that guide program management to perform within given budgets, deadlines, required acceptance levels and goal adherence;
- Plans, that take the implementation goals and drive the projects within the program;
- Communication, as a means of information to teams and users, and to solve problems within the teams;
- Alignment, the process of adapting the system to the organization or vice versa, in line with the business direction.

2.3.1. Program Organization. On the basis of the Law of Requisite Variety [18] we propose that the design of the implementation program organization should support sensing and controlling the complexity in the program environment:

Proposition 1: A complete program organization in complex program environments contributes to program success.

In particular we suggest that teams of successful complex implementation programs should be structured such that they cover all of the following roles: Program manager, Steering committee, Program Sponsor, User representative, Coordinator across projects ("Global Process Owner"), Coordinator with external suppliers, Coordinator for an efficient implementation process ("Site Implementation Manager"), independent Quality Assurance.

2.3.2. Policies. In particular, we look at budget and time restrictions. Several authors suggest that few restrictions should be impinged on the program in its innovation phases (visioning, software selection, test and pilot), while tighter restrictions are called for in the later rollout phase [6] [12]. The reasons given are that in the innovation phase one should strive for effectiveness (taking time for innovative thinking to find the right solution through design work and extensive trial runs), and in the rollout phase for efficiency (implementing it right). We propose:

Proposition 2: A loose budget policy, in terms of time and cost restrictions, in the innovation phase (visioning, software selection, test and pilot) contributes to program success.

2.3.3. Planning. In particular, we look at the extent to which implementation projects run in parallel and the technology or organizational emphasis chosen in the implementation route. When the number of implementation projects is large, the program manager has to consider parallel implementation projects, in order to complete the program on time. A “Big Bang” strategy would visit each affected site only once, and would introduce all technical and organizational innovations at once. However, due to e.g. limited resources, this may...
not be possible once the variety exceeds a certain number of locations. Depending on the number of sites affected, the number of parallel projects will differ. We propose:

**Proposition 3:** Adjusting the number of parallel implementation projects to the level of variety in the program environment (number of elements and their interrelationships, e.g. number of sites etc.) will contribute to program success.

If the implementation strategy is not “Big Bang”, the question is which innovations to concentrate on in the first round of implementation projects. If the program introduces IT-changes first, leaving the existing operational procedures largely intact, we speak of a technology-driven implementation route. If the program introduces organizational changes first, based on the current application systems, we call it a business-driven route. In most of our case studies, it was a mixture of both, in order to complete the program in a shorter time. For programs with little innovation (little integration), the new system will be a replacement of existing systems. Increased integration will introduce the need for more organizational changes. We propose:

**Proposition 4:** Adjustment of the implementation route to the degree of integration will contribute to program success.

2.3.4. Communication. In particular we looked at management sponsorship as being affected by the complexity of the program environment. Sponsorship should penetrate all levels within an organization (“cascading sponsorship” [7]). The more locations are affected in a program, and the more they differ, the more difficult it will be for the sponsor to make his/her influence felt, heard, seen. We propose:

**Proposition 5:** Program environment variety negatively affects the level of management sponsorship penetration.

2.3.5. Alignment. To assess the level of alignment we looked at the use of alignment mechanisms and tools. Program reviews, steering committees and other efforts can be installed in order to achieve alignment between the current goals and requirements on one hand, and the implemented systems and procedures on the other. We expect that a successful program of high variability will use more organizational means for alignment than programs with low variability. We propose:

**Proposition 6:** More prevalent use of alignment mechanisms in high variability programs will contribute to implementation success.

3. Empirical Analysis

3.1. Focus on successful programs

We focus on successful programs. We attempt to find the ways in which they set themselves apart from similarly successful, but less complex programs. Consequently in the evaluation design, we controlled the relationship between program management and program success, and kept the relationship between environment complexity and program management variable. Figure 2 is a representation of our conceptual model and illustrates the evaluation design.

3.2. Case studies

Fifteen cases were studied to show empirical evidence for the proposed framework. Most of them ended in 1998/99. They were lead by Western-European teams and included national or worldwide rollouts. The following industry sectors are included in our sample:

- **Services:** postal service, transportation, utility, and education
- **Consumer products:** automotive, electrical appliances, and garment
- **Industrial products:** pumps, packaging, and electrical engines
- **Process industry:** chemicals, pharmaceuticals, and healthcare product.

The companies from the services industry are very large, national enterprises. Their rollouts were national and affected many locations (average: 104, maximum: 248). The manufacturers of consumer goods are large organizations, with an international range of affected locations (plants and sales offices). The producers of industry products are mid-sized companies that maintain an international presence primarily through sales offices. The companies from the process industry show a variety of sizes and rollout scopes. In our final set SAP R/3 was the package of choice in 6 out of 15 companies; other packages were Baan, BPCS, Oracle Applications and Catalyst.

In each case, interviews with key officials were held. Relevant reports were analyzed and archival data were
obtained. The interviewees were identified by contacting consulting companies (PwC, KPMG, CSC Ploentzke). About half of the program managers identified were from a consultancy service or from the software supplier; the other half was in-house staff. The interviews were semi-structured; they all followed a questionnaire, leaving room for additional remarks. An EXCEL database contains the coded answers from all interviews; during the interviews, the program managers were shown how their answer was recorded on the questionnaire form.

The discussion in the following sections will be structured according to the analytical model. We will discuss the level of implementation success in the 15 cases in section 3.3. Section 3.4 focuses on the complexity of the program environment. The propositions with regard to program management will be evaluated in section 3.5.

3.3. Program Success

The programs were all successful; none had been aborted or, for those still running, were in danger of being stopped. But were there differences in their levels of success? Figure 4 shows the result for the 13 cases that had fully completed the success assessment; in two cases, data for one success criterion was missing, three fall together on one data point (80;95). Because, however, during the interviews the case descriptions indicated success, we chose to include them in the set. We have also included 3 programs where the project success was below 70%. They all showed heavy cost overruns. In one case, the project was completed very efficiently, but the product success was only 70%, because even after a "handholding period" after Go-Live, the users were not fully conversant with the system. We still included these four cases in our study, because the programs finished with a new ERP system and processes in place.

3.4. Program Environment Complexity

Table 1 shows the computed complexity indicators and an overall median for complexity. We are using the median as the computed complexity score rather than the sum, because we want the scores to be comparable on the same scale (1 to 5). The median is appropriate (rather than the mean), because all variables are on an ordinal scale. For cases 2 and 7 the integration indicator could not be computed, because of insufficient information from the interviewees. These two cases have been eliminated from any assessment regarding integration complexity.
Table 1: Complexity indicators for 15 cases.

<table>
<thead>
<tr>
<th>Case no.</th>
<th>Variety</th>
<th>Variability</th>
<th>Integration</th>
<th>Median overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>8</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>14</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 2: computed versus subjective rating for complexity.

<table>
<thead>
<tr>
<th>Subjective rating</th>
<th>Computed rating</th>
<th>High (&gt; Median)</th>
<th>Low (median)</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>High (&gt; Median)</td>
<td></td>
<td>6</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>Low (median)</td>
<td></td>
<td>0</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Sum</td>
<td></td>
<td>6</td>
<td>9</td>
<td>15</td>
</tr>
</tbody>
</table>

3.5. Program Management

3.5.1. Organization

Proposition 1: Completeness of program organization in complex program environments contributes to program success.

All programs were structured with clear responsibilities in the team. Figure 5 shows responsibilities which we had proposed as being important, and percentages to which they were found in the programs (n = 15). Most program organizations were very complete, except for some coordinating roles and the external Quality Assurance role. One case had no coordination functions at all, two others were lacking coordination across projects. Incidentally, two of these three cases were the least successful projects, with a project success of 60% and below, and a product success of 80%; the new system was pushed in almost on time, but with expensive external help. Several program managers stated that they themselves carried out the coordinating roles. However, in more complex programs, due to reasons of span of control, appointment of a separate coordinator is necessary to assure sufficient attention. We conclude that proposition 1 is supported.

Fig. 5: Roles in program organization.

3.5.2. Policies

Proposition 2: A loose budget policy, in terms of time and cost restrictions, in the innovation phase (visioning, software selection, test and pilot) contributes to program success.

We analyzed the cost and time restrictions for both the innovation and implementation stage of the programs, by asking for a score from the ordinal scales as shown in Table 3.

Table 3: Scales applied for cost and time restrictions

<table>
<thead>
<tr>
<th>Cost</th>
<th>Success at all costs</th>
<th>Limited budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Restricted budget; restrictive</td>
<td>87%</td>
</tr>
<tr>
<td>4</td>
<td>Tight budget; dictates approach</td>
<td>80%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>Whenever you are ready</th>
<th>Limited time</th>
<th>External deadlines restrict</th>
<th>Deadline dictates very fast</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 4: Cost and time restriction by phase
Table 4 shows the median values for both the innovation and implementation phase. All the programs had clearly defined budgets; the innovation budget was least limited. We validated the proposition by comparing the two samples of scores using the sign test [8]. The null hypothesis is that cost and time restrictions are equal in both phases; the alternate hypothesis is that in the implementation phase the restrictions are higher than in the innovation phase. For cost restrictions we conclude that the case studies support the proposition; also for deadlines the proposition is fully put into practice by the programs. We conclude that the results from the case studies support proposition 2, because the restrictions in the implementation phase are tighter by one level.

3.5.3. Planning

Proposition 3: Adjustment of the number of parallel implementation projects to the level of variety of the program environment will contribute to program success.

The maximum number of parallel projects is 12. The number of parallel projects increases with the number of locations affected, it also increases with variety in general, as Figure 6 illustrates.

The three exceptions (variety of 3 or greater, with 2 or fewer parallel projects, see lower right corner of Figure 6) were programs, which used a sequential implementation approach despite a high variety. They were not very successful in their implementation efficiency; they were the three programs with the lowest score for "project success" (see Figure 3).

3.5.4 Communication

Proposition 5: Program environment variety negatively affects the level of management sponsorship penetration.

In more than half of the cases, sponsorship was very active at all company levels. On the other hand, in a third of the cases, it was "lip service" only. 6 out of 8 high-variety programs show complaints about lack of sponsorship. In contrast, in programs with low variety, only 1 complained about sponsorship. We conclude that the extent of sponsorship given in our case study set was sufficient only in the low-variety cases. High-variety implementation programs needed more active organizational changes. Managers of the most complex programs commented that because of the complexity they had to go the middle route.

Figure 8 illustrates that this effect is indeed present in our case studies: only programs with little integration needs went a mainly or purely technical route. The case data support proposition 4. We conclude that with increasing integration complexity, the organizational changes were implemented together with the technical changes.
sponsorship than currently given. Table 5 shows details. The case results support proposition 5.

<table>
<thead>
<tr>
<th>Sponsorship Variety</th>
<th>Active</th>
<th>Lip service only</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>High (&gt; 3)</td>
<td>2</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>Low (&lt;= 3)</td>
<td>6</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>Sum</td>
<td>8</td>
<td>7</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 5: Contingency table for sponsorship versus variety

3.5.5 Alignment

Proposition 6: More prevalent use of alignment mechanisms in high variability programs will contribute to implementation success.

Alignment was organized as a mix of information gathering (reviews) and direction setting (steering committees, budget restrictions). Overall, there was little institutionalization of alignment (see figure 9).

Our case-study set did not support proposition 6. Computed for integration, it appeared to support the proposition for integration complexity. We conclude that the more challenging the program goals (here, high innovation in terms of integration), the more alignment mechanisms were to be put in place. A reason for that could be that increased integration complexity goes together with a higher degree of technical and/or business innovation. This requires a higher degree of prototyping and a greater need for aligning the program goals and scope with what turns out to be technically and operationally feasible

<table>
<thead>
<tr>
<th>Align.mech.Integration</th>
<th>High (&gt;median)</th>
<th>Low</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>High (&gt;3)</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Low (&lt;=3)</td>
<td>2</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>Sum</td>
<td>4</td>
<td>9</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 6: contingency table for alignment mechanisms versus integration complexity.

4. Conclusions

E-business has lead to increased need for program management. New business models and faster innovation cycles demand close coordination of a number of business process reengineering and change management projects, together with projects that implement the enabling new information technologies. In comparison with standard ERP implementations, complexity in managing programs with an e-business component is shifting. Integration increases because value chain integration requires wider integration of IT architectures as well as business processes. Also variability rises because the new business models and technologies are not as proven and stable as ERP. Our framework and propositions can be used to manage such programs more effectively in that it

- contributes to a better understanding of the different components of the complexity that program managers will be facing, and
- highlights limitations and strategies that earlier successful programs have adopted in similar situations.

4.1. Managing increased integration complexity

The complexity variable integration can be used in program management to

- indicate the number of innovation steps in IT interoperability that have successfully been handled in other programs. In our set of 15 cases, most companies took only one or at the most 2 steps at a time in a project
- decide on the implementation strategy. Should Program Management introduce the new IT architectures first and afterwards the new business models, or both at the same time? Our survey of cases shows that when integration is increased, it is conducive to program success that both types of changes are introduced at the same time (proposition 4)
- organize for alignment with the business direction: mechanisms such as project reviews or a Steering Committee should be installed in order to align the technical solution with the reengineered processes and the program goals (proposition 6).

The increased integration complexity will generally lead to more complicated changes per end user, requiring extended change management efforts. Our survey showed that program managers were keenly aware of potential resistance from users, and that they pushed for management sponsorship, but the success varied. It turns out that sponsorship effectiveness is not so much limited by integration complexity, but by variety (proposition 5).
The risk for program success is less the extent of change per person, but the localization efforts required. Program organizations adapt to this, for instance by introducing a Coordinator across projects (e.g. a "Global Process Owner"), by not running more than 12 rollout projects in parallel (implied in proposition 3), and by providing an escalation instance like the Steering Committee (proposition 1).

4.2. Managing increased variability complexity

When the desired level of integration requires leading edge technology such as B2B marketplaces or internet banking, it will increase variability, because it involves piloting an evolving solution. Experienced resources are in short supply, and the supporting technology is often not fully stable yet. Our survey of case studies suggests to coordinate closely with vendors and with other (technology pilot) projects (proposition 1), and to give more budgetary latitude during the design, piloting and test phases as compared to the rollout phase (proposition 2).

The above hints should be useful making decisions of how to organize a program ("forewarned is forearmed"), but they must not be taken as rules that will guarantee program success. The limitations of the underlying contingency theory would not allow to go that far. More research is required on recent ERP and e-business programs, in particular on failed ones. The research presented here provides a foundation and indications for success factors to look out for.

5. Bibliography

22. Mitroff, I., Betz, F., Pondy, L., Sagasti, F. On managing science in the systems age: two schemes for the study of science as a whole systems phenomenon. TIMS Interfaces, Volume 4, Number 3 (May 1974).
30. Schoo, K.-C. Engineering Complex Software Implementation Programmes. Berlin: VDE Verlag, 1999