Strategies for global information systems development

By: Murad Akmanligil, Prashant C. Palvia

Akmanligil, M. and Palvia, P. "Strategies for Global Information Systems Development." Information & Management. Vol. 42, issue 1, December 2004, pp. 45-59.

Abstract:

Developing global information systems is a formidable task. Multinational companies operate in regions that are thousands of miles, many time zones, and many cultures away from headquarters. Organizing the activities and aligning the tasks and mindsets of people that are so far apart and to change the way that business is conducted through the use of IS is a major challenge. This study discusses alternative global IS development strategies and the factors that impact their selection. Four systems from a large transportation company are presented as real life examples to demonstrate the viability of these strategies and the accompanying factors.

Keywords: Global information technology IS development strategies; IS success; Contextual factors

Article:

1. Introduction

Information technology (IT) has been advancing at an exponential rate. Cheaper and faster computers and communication devices are introduced to the market every day. Nevertheless, the software cost of building information systems (IS) is significantly higher than that of hardware, the ratio being 80:20 or even higher [20]. Companies spend millions of dollars to automate their processes, improve decision making, and gain strategic advantage. In spite of that, many systems are late and over-budget, and thus, many development efforts fail. Ewusi-Mensah [9] argues that the software industry is in the state of crisis and cites a study showing that 31% of new IS projects are canceled before completion at an estimated combined cost of $81 billion and that 52.7% of the projects completed are 189% over budget at an additional cost of $59 billion. According to Lai [21], improving the effectiveness of software development is one of the top concerns of IS management. Their frustration with IT costs, backlog of system requests, and standards on infrastructure are forcing IT managers to seek alternative solutions to in-house development. Outsourcing and purchasing package software are frequently used as solutions to these problems.

A major factor complicating the issue of IS development and its success is the globalization of companies. Today, this trend is phenomenal, to the extent that some industries such as electronics, computer, and pharmaceuticals are considered predominantly global. In order to compete, companies have to exploit their advantages. Globalization brings forth the challenge of coordinating their activities on a worldwide basis, mostly through using IT. However, globalization drastically increases the complexity of the development process by introducing many new variables and unknowns. As Karimi and Konsynski [18] note, variations in business environments, availability of resources, and technological and regulatory environments are faced by the IS organization as soon as a system crosses national boundaries.

A global information system (GIS) is used across one or more national borders [2]. The IS executive has many system development strategies from which to select in satisfying global system requirements. Alternatives to the classical approaches of in-house development, such as outsourcing, and commercial off-the-shelf software (COTS) now include offshore outsourcing [31,34] and global software development [13]. GIS development poses many challenges. Questions such as: how to get support from subsidiaries, how and who should go about gathering the information requirements from subsidiaries, who should build the system, how to exploit existing applications, what code should be common and what should be local, how many versions of the code are needed, etc. present a wide variety of difficult choices. The choices are not clear-cut and require careful consideration.

2. Global information system development strategies

A major problem facing the firm [25] is to identify the acquisition option/development strategy that will maximize the net present value of software acquisition, subject to organizational considerations (e.g. corporate policy and resource availability). Because many strategies are available, the choice is very difficult.

In the domestic context, insourced development, package acquisition, and outsourcing are the primary strategies available to companies. Additional alternatives are available to global companies. In the literature, eight GIS development strategies are noted [15,17]. We added another. The nine strategies are:

1. Development with a multinational design team (MDT)

2. Parallel development (PD) Central development (CD)

3. Core versus local development (CL)

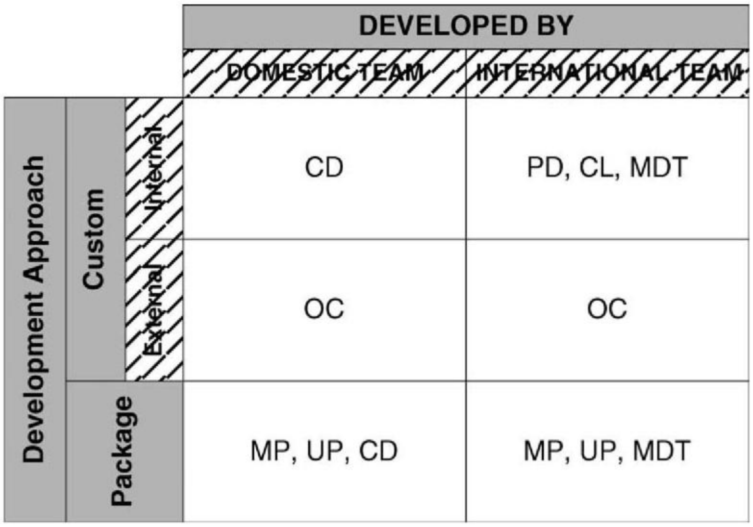
4. Best-in-firm software adoption (BIF)

5. Outsourced custom development (OC)

6. Unmodified package software acquisition (UP)

7. Modified package software acquisition (MP)

8. Joint development with vendor (JDV)



			DEVELOPED BY	
			DOMESTIC TEAM	INTERNATIONAL TEAM
Development Approach	Custom	Internal	CD	PD, CL, MDT
		External	OC	OC
	Package		MP, UP, CD	MP, UP, MDT

Fig. 1. GIS development strategy framework.

A framework is provided in Fig. 1; it categorizes the development strategy by two dimensions: the development approach (custom or package) and the number of countries involved (domestic and international). The custom approach is further divided into two classes to represent whether the system is built internal or external to the firm. The nine development strategies were placed in appropriate cells; some strategies fall in several.

2.1. Development with a multinational design team (MDT)

Here a design team consisting of systems and user personnel from multiple sites is formed. The team members gather at one location for months to work on the system design. It is argued that the participation of the involved parties not only creates a design that meets the requirements of all regions but also increases the likelihood of accepting the software and adhering to international standards. However, the cost of this strategy is reported to be high. Furthermore, the right team leader and the team composition [27] are important for the success of the project.

2.2. Parallel development (PD)

Here the requirements gathering and the construction of the various subsystems are performed locally and then connected through bridges or integrated. Some companies use a variation of this strategy: representatives of teams at different locations, gathering local requirements, then come together at a central site and try to resolve the differences, and thus, define a common structure. At another extreme, the project is broken into multiple components at the start and different components are designed and built at different locations. Coordination and consistency among these systems are provided through common development methodologies, shared software engineering tools, electronic mail, and consistent definitions of data. Herbsleb and Mockus [12] reported that globally distributed development appears to take two and one-half time as long as when work is co-located.

2.3. Central development (CD)

With this, the system is developed at one site (probably the headquarters) and installed at the subsidiaries. The advantages of this strategy are lower costs due to economies of scale, the enforcement of standard operating procedures throughout the company, and better communications among various parties. The disadvantages are potential resistance by the subsidiaries, unsatisfied requirements of some regions, and diminished buy-in from uninvolved regions [4].

2.4. Core versus local development (CL)

Collins and Kirsch [5] observed that project team structures were "typically complex, reflecting the need for centralized effort to create a common global solution, while at the same time understanding and accommodating local needs." Burn and Cheung also pointed to the approach where local IT departments tailor the components created by the headquarters to fit their needs; we call this: Core versus Local Development. While minimizing local conflicts this tends to increase coordination and control requirements.

2.5. Best-in-firm software adoption (BIF)

This strategy was observed to be the most common by Ives and Jarvenpaa. With this, the best software anywhere in the company is selected to be implemented globally. But the roll-out of the software to the regions is gradual. Some modifications may be necessary due to differences in requirements and technological resources. Sometimes, it may be necessary to provide a region with its own versions of the software. This causes duplication of effort and drastically increases the cost of maintenance.

2.6. Outsourced custom development (OC)

Outsourcing has become a popular strategy in software development [23,26]. Many companies are now engaged in global (offshore) outsourcing [24]. This strategy is chosen: to implement and enforce global standards, reduce cycle time, reduce costs, gain access to state-of-the-art technology, shift focus to more strategic systems, etc. [1]. Opponents argue that loss of control and flexibility, loss of qualified IS staff, and loss of competitive advantage are the major risks in outsourcing [30]. With this strategy, the system is developed by an external company (the service provider) who may use any of the other development strategies.

2. 7. Unmodified package software acquisition (UP)

Many companies are purchasing package software rather than developing it themselves. Enterprise resource planning (ERP) systems are examples of such software. This strategy provides a working system to the users relatively soon after the start. The packages are generally parameter driven and are configured extensively. With unmodified packages, vendors are generally able to provide better support and future modifications are likely to be compatible. However, the package may not completely meet the requirements and the firm may have to modify its processes to accommodate the software. Moreover, as Hamel [11] points out, commercial off-the-shelf software (COTS) do not provide unique advantages to companies.

2.8. Modified package software acquisition (MP)

The “vanilla” (unmodified) package does not always meet requirements of all regions of a company, even when the system is parameter driven. In such cases, the software needs to be modified. Ragowsky and Stren [29] suggest only minimal modification, as high development and maintenance costs may be incurred when the vendor upgrades a standard package.

2.9. Joint development with vendors (JDV)

This strategy is also termed a “strategic partnership” with the vendor. The approach is different from just hiring ‘contractors’ to help code or deploy the system; it involves heavy and committed participation of vendors in one or more of the phases. In most cases, the vendor provides guidance or shares the responsibility of the success of the system. As in outsourcing, some of the reasons for involving vendors are: access to new technology, access to project management skills, and the desire to jump-start the project. As a variant, when the package acquisition approach is chosen, the package vendors could be asked to help in any modification. Generally in the JDV approach, the host company desires to take over the development or maintenance in the long-term and wishes to internalize the skills brought by the consultants.

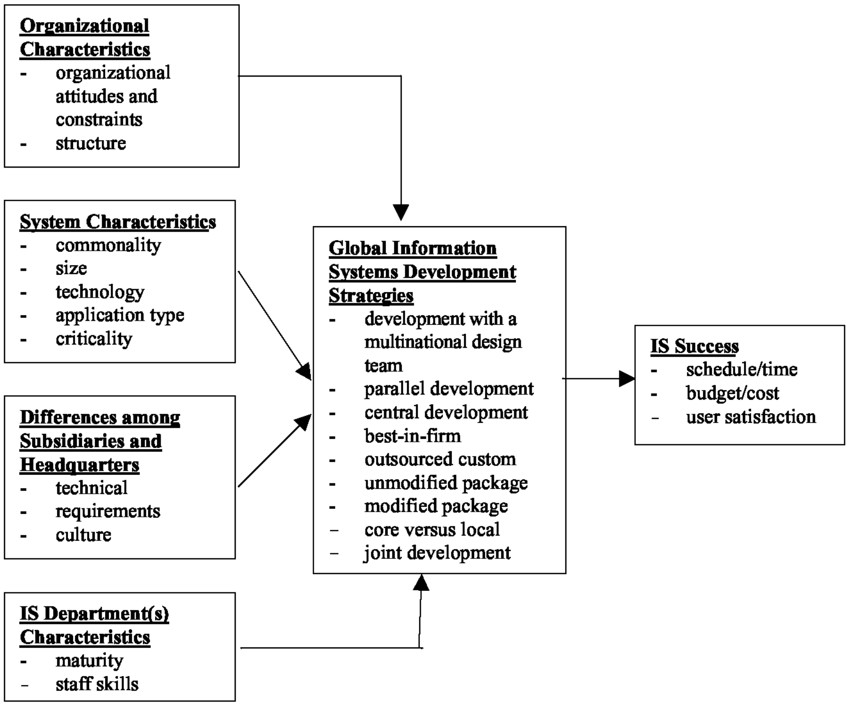


Fig. 2. A guiding framework for GIS development strategy selection.

3. A guiding framework for GIS development strategy selection

The selection of a GIS development strategy is complex and influenced by many factors, both objective and subjective. In order to study the selection process, we developed a framework; Fig. 2 shows the various factors that might need examination. Based on a review of the IS literature, the following factors were identified as impacting the selection of a GIS development strategy: organizational characteristics, desired system characteristics, differences among subsidiaries, and IS department characteristics. The framework was designed primarily to help us study GIS development strategies. No attempt was made to postulate specific hypotheses; the intent was to generate them based on the results.

3.1. Organizational characteristics

The characteristics considered important to software development are: organizational attitudes and constraints [28], and organization structure [10]. For example a corporation due to its polycentric attitude could implement a policy urging the subsidiaries to produce their own solutions to their requirements. In another situation, management frustration over IT costs and responsiveness could affect acquisition decisions in favor of outsourcing.

The organizational structure could also impact development strategy. According to Ghoshal and Nohria, there are four types of structures/environments: international, global, multinational, and transnational. In a multinational environment where the subsidiaries act as domestic companies, the cost of systems development using multinational design teams would be considerably higher than in a transnational company, because there is normally very little coordination among subsidiaries and greater differences in business processes in a multinational structure. As per Cash et al. [3], as firms adopt different organization structures, they need different levels of international IT support and coordination. They also state that for "central development" to be successful, the organization should have "established patterns of technology transfer, strong functional control over their subsidiaries, and some homogeneity in their processes." Therefore, it is less likely to be used in a multinational structure.

3.2. System characteristics

The characteristics play an important role in selecting the development strategy. Some key ones are: commonality, size, technology, type, and criticality. Commonality is the extent to which a particular type of software is used by other companies. There is empirical evidence that common applications are more likely to be outsourced or bought than being custom built. Relative size of the project (compared to others in the company) is a factor that affects its risk. As the size and scope of the project increases, the coordination costs and risk increases. Outsourcing the development of the system could reduce this. Collaboration and exchange among the subsidiaries and the headquarters can be used to reduce risk.

New technology, in general, increases risk. Then, outsourcing should be considered seriously, since a technically skilled outsourcer can alleviate risk. Application type also affects development strategy selection. Systems with relatively low structure (such as decision support applications) are typically bad candidates for outsourcing while highly structured systems have defined and fixed outputs and are easier to outsource. System criticality can also affect development strategy. If a system directly helps the company with its core competencies and is central in implementing a key strategy, the system is "critical." For this, a multinational design team strategy might be more effective, as it can help ensure the commitment of management. To maintain control, insourced development would then be preferred over outsourcing.

3.3. Differences among subsidiaries and headquarters

Foreign subsidiaries and headquarters may have significantly different technologies, information requirements, and culture. Technology variations include stark differences in the availability and quality of both hardware and software. These may force a firm to use different vendor products in different subsidiaries or multiple versions that run on different platforms or necessitate the involvement of an external organization in bridging the platforms.

Variations among the requirements of subsidiaries can be overwhelming. Even seemingly minor differences can necessitate major local tailoring [19]. DiNardo [7] pointed out that parameter driven software his project used in Asia was significantly challenged in accommodating the differences. In order to prevent development failures and increase global ownership and smoother implementation, Palvia and Lee recommend worldwide elicitation of requirements.

It is crucial to consider cultural differences among subsidiaries and headquarters in making development decisions. Shore [33] recommended the inclusion of national culture in IS methods that attempt to create linkages between headquarters and subsidiaries. Hofstede [14] identified four dimensions of culture: power distance, uncertainty avoidance, masculinity, and individualism; as an example, organizations in less risk-taking cultures may find some of the development strategies (e.g. MDT and PD) too risk-prone. While acknowledging the role of national and organization cultures, Robey and Rodriquez-Diaz [32] argue that organizational culture plays a more determining role than national culture.

3.4. IS department characteristics

The maturity of the IS departments and the difference in skills among various IS departments within the organization also has a bearing on the development strategy. The specific technical capabilities of the IS department will directly influence the decision to build the system internally or not. Its maturity and experience in developing GIS also has a role in acquisition decisions. An IS department facing problems in establishing and enforcing global standards could choose to alleviate the problems by outsourcing.

Skill set differences of IS staff is another concern. Lack of technical skills or lack of experience with large projects could influence the level of collaboration among regions and headquarters. Lack of technical skills in the regions could also prevent the use of the parallel development approach and favor outsourcing or mandate extensive knowledge and resource sharing with the headquarters and other regions. On the other hand, headquarters' lack of knowledge of subsidiaries' business practices and cultures could mandate a cross-border solution.

3.5. System success

The choice of an appropriate development strategy will greatly influence the success of the project. However, the choice is dependent on many factors and the relationships are complex. Delone and McLean [6] concluded that IS success is a multidimensional construct whose dimensions are system quality, information quality, use, user satisfaction, individual impact, and organizational impact. Of these, user satisfaction has received considerable attention [8,16]. Many of the dimensions are interrelated [22].

4. Research method

Yin [35] states that the case study is a distinctive form of empirical inquiry and is adequate for all types of investigation (i.e. exploratory, descriptive, and explanatory). It is especially appropriate when trying to understand complex social phenomena, when the investigator has little control over events, when the focus is contemporary events, and when "how" and "why" type questions are asked. Therefore, this approach was selected for our study. Multiple-case design was employed because:

1. The descriptive goals of the study (such as the determination of existing GIS development strategies) cannot be accomplished in a single case.

2. The effect of organizational characteristics and IS department characteristics on GIS development strategy selection cannot be analyzed with one case.

3. The evidence from multiple cases is often more compelling and the overall study is more robust.

4.1. Execution of the study

A semi-structured questionnaire was prepared: qualitative data was gathered in an interview setting from executives in global firms. Free-form exchange of information was encouraged; it was expected that some issues would elicit more information than others and that new issues would emerge during the course of the studies.

In the first phase, a pilot case study was conducted. Its objective was "to refine data collection plans and procedures." The conceptual framework and the adequacy and clarity of the interview questions were re-evaluated based on feedback received during the pilot study: though no changes were made to the conceptual framework, some questions were clarified and a few were deleted.

For the final case studies, the researcher conducted semi-structured interviews with IS and/or user department executives as well as with the project manager. The focus was always on specific global IS projects. The respondents were contacted in advance. If the request for an interview was accepted, available documentation on the project was requested. This was analyzed before the interview, and project specific questions were prepared. The interviewees also identified key individuals to answer questions that they could not. These individuals were then contacted and information was, therefore, gathered from multiple sources: some provided information on the acquisition decisions, some on the development process, some on the technical architecture, etc. Often, new issues arose and were documented. Patton's advice was followed in post-interview and documentation procedures. A report was then written and sent to the interviewee(s) for their review. This ensured that the facts were consistent with those of the interviewees. Their feedback was then incorporated.

5. Cases—global information systems

The four cases threw light on the factors influencing the choice of strategies and their success. These were projects from a major transportation company (MTC). The identity of the company and the people interviewed were disguised per their request.

MTC is a large company headquartered in the US. This multinational was organized into five regions and each was made responsible for its own profits and losses. It is a highly customer oriented company, trying to establish itself as a major player in the global economy by leveraging IT. Its relatively large IT department is generally centralized with the majority of its employees in the US. However, there are some IT personnel in other regions; they are responsible for support, maintenance, and new development in their region.

In the mid 1990s, MTC started to implement an executive initiative to move its systems from mainframes to a client/server architecture. On the server side, software included Unix, C/C++, Sybase/Oracle, while on the client side there were Windows, Java, and Visual Basic; these are becoming company standards. The majority of MTC's applications are custom built in-house; however, the company has been experimenting with package software and domestic, as well as global, outsourcing.

5.1. Case 1: Global Clearance (GC) system

With growing international operations, MTC created several country-specific domestic systems to handle customs clearance operations. However, not all such processes were available in any single system. They were not integrated with their functional systems such as International Credit/Collections, Duty/Tax charges, and Sort Operations. This, compounded with the fact that some processes were not automated and some were outsourced, made maintaining control of data very difficult. There was a need for a single system that provided all clearance related processes. Therefore, a project called *Global Clearance* (GC) was initiated. The system was expected to reduce costs by "eliminating manual processes and paperwork, managing dispute processing, increasing productivity, and reusing detailed customer information."

There was considerable commonality in the existing systems. Eliminating any redundancy by creating a set of core components was, therefore, a major objective. It seemed that this strategy would reduce the development, maintenance, and support costs in addition to reducing the cycle time for delivering new clearance systems. The system was planned to provide a single point of access and automate Import Clearance in major countries. The project plan defined the scope as:

> The scope of Global Clearance system extends from the point at which information and/or goods enter the country of importation to completion and archiving of the associated financial transaction and encompasses all systems and data required for clearance.

The system was to be implemented at every site with more than 500 shipments per day; this was the estimated break-even-point. In September 1996, 31 sites in 15 countries met this criterion. This project originated from another in Europe called Europe Clearance, whose requirements were gathered by a team consisting of UK. IS personnel and user representatives from several other European countries. This was carried out in Brussels using a 4GL tool called Uniface. UK was responsible for building the back end. After about a year, it was decided that the headquarters' IT department could help UK and a project called the International Broker Clearance system was merged with Europe Clearance to become the Global Clearance system.

During the mid 1990s, corporate headquarters mandated that newly developed or bought systems would have multi-tiered software architectures. The application development manager of the project described this strategy as:

> I think we were looking at (outsourcing) to begin with to see how much of anything can be out-sourced. And we always thought the broker piece was the central piece ... Before we really got going very far, we started looking at broker packages ... but we didn't find any client server broker packages and we didn't find any people that were interested in working with us to build one ...

Given this, the system had to be built in house. Since clearance systems built before GC were not three tiered, the team could not use much code from them. Some faxing software from another system called *Japan Broker* was used with modifications and another system was studied to get business rules and ideas.

The decision to involve HQ in the development of *Europe Clearance* was made to expedite development. UK's delay was probably due to an unrealistic schedule: the lack of availability of resources. The IT departments there were much smaller than at HQ. Unanticipated problems with new technology and scope creep also contributed to the delay. Headquarters could help as they had built functionally similar systems and furthermore some parts of *Japan Broker* could be reused.

The HQ asked an external company to perform an audit of several existing systems, including the clearance operations. This resulted in recommendations for changes to the existing systems. The enhancements were incorporated into the requirements of the *Global Clearance* system. Later, the division of labor was changed. The Headquarters IS department was asked to build the core and common components and UK to build the local pieces. According to the project manager, 90% of the code was expected to be core and common and 10% to be local. A decision was made that regional IT groups would develop, install, and maintain the necessary local components and local connectivity. The core and common pieces would be scalable and reusable and the regions would be free to select the components they needed and tailor them to their own needs.

A few months later, the development strategy was changed to give HQ the responsibility of building the entire system. Analysis, design, and coding were to be carried out centrally. The system was unit and assembly tested at the HQ, user acceptance and volume tested in the UK, and training was conducted internally in the UK.

During these changes, the UK and HQ IT departments periodically communicated and visited each other's sites. The UK personnel were able to install the hardware and perform a "proof-of-concept" test. However, they concentrated on hardware, since they had a separate budget for it. The differences in priorities (hardware versus software), politics, limited resources, and the different ways the two groups conducted analysis and design contributed to the shift of the strategy from parallel core-local to central development.

Currently, development of the system has stopped. Only the first phase out of the planned eleven was completed. The partial system is being used at one location in the UK which uses Global Clearance in conjunction with their existing system, though they do not interface with each other.

There were many reasons for discontinuing the Global Clearance system. The first phase was internally funded and a proposal for the remaining 10 phases was to be presented to the Board of Directors (BOD) after completing the first phase. However, delays in both Europe Clearance and GC and the increase in requested funding due to the broadened scope reduced the systems' chances of obtaining BOD approval. Furthermore, according to the development manager, IT executive management started saying that the regions know their needs better than HQ and were fully capable of writing their own local applications.

5.2. Case 2: global accounts receivable (GAR)

MTC started to look at the possibility of a new accounts receivable (AR) system as the current largest one seemed to be reaching its limits in handling the transaction volume. This system was handling the receivables of

US payers and was about 14 years old. It had been patched many times and was losing its flexibility to handle code change. It was also intended to present a global financial picture of the company and of its customers. There were many other AR systems in the company but they did not necessarily interface with one another. This made the preparation of financial documents and balancing different accounts very cumbersome.

In the summer of 1995, with re-engineering in mind, MTC asked a consulting company to conduct an analysis of the US AR system. This company determined that (1) MTC had high transaction processing costs due to many customers receiving EDI invoices but not using EDI for remittances; (2) inefficient and outdated procedures resulted in duplicate handling of customer inquiries, costly research, and low employee morale; and (3) the largest AR system handling the US payers needed to be replaced due to the growth in transactions and its inflexibility. Other AR systems were more than 10 years old.

A decision was made to examine AR packages available in the market: there was a general impression that an accounts receivable system is much the same from one company to the next and there was a directive from upper management that package solutions should be considered first. After a potential package was selected, a pilot was conducted for performance evaluation. Some team members also visited companies in the UK and New York to observe real life implementations of the package. A high-level requirement document was prepared to bridge the gap between the capabilities of the package and MTC requirements. The vendor agreed to implement the enhancements these as standard package upgrades.

A project team consisting of members from MTC, the vendor (ARC–AR Company), and a consulting company ("CC") was formed. ARC was to help expedite the implementation and CC was to provide project management expertise. The team's objective was to detail requirements for enhancements and build the business model for the US and then other regions. It was a complicated process due to the highly parameterized nature of the package. A development team was also formed to help interface the package with existing systems and customize the package. A decision was made that the core functionality of the system was to be modeled at the HQ and that the regions would be responsible for local tailoring. This distribution of workload was made to speed the pace of implementation. However, the company had problems "in pulling all the requirements and the organizations together in the international community." The representative of International Air Freight operations underlined some of the difficulties:

> The main problem in working with the regions is when to bring them in. If it is too soon, they complain that we are wasting their time. If it is too late, they complain that we have already decided what we are going to do so why ask them now. (Due to their limited resources,) they have difficulty in providing dedicated support for the project.

GAR was implemented in the US and is being used for airfreight and special non-transportation receivables. Implementation in other regions was stopped: a primary reason was that few enhancements were developed because they were relatively large and the vendor had limited resources, plus other customers did not want the enhancements. The managing director of IT stressed another reason:

> During the evaluation of packages we did not demand international representation. It was voluntary for regions to participate and we informed them about the progress but that is not enough.

5.3. Case 3: operations service level system

The operations service level system (OSL) provides near real time information of the service (e.g. timely delivery) for each shipped package. This information enables the analysis of MTC's service to its customers and helps identify systemic problems. It provides drill-down capabilities and Intranet based reporting capabilities. The system was intended to replace its counterpart on the mainframe. The company had established a service quality index based on different types of service failures and all employee bonuses were affected by this index. The leader of business team of the project summarized the major motivations as:

> ... The new system is supposed to be the single point of source for all service level information. Some departments needed information, they needed a query based system and the old system did not have this capability. Using it was difficult too. They had to deal with Focus program tapes and not everybody could do that ...

Many requests could not be satisfied by the old system; e.g. determining the source of service failure at intermediate points and the analysis of service levels by customer. The old system had reaching its limit of growth and flexibility and the corporate initiative required moving systems from the mainframe.

Purchasing package software for OSL was never considered seriously, due to its unique nature. The OSL was part of another project, Service Squared, that began in 1995. Its objective was to provide an enhanced version of the mainframe service-level program and consisted of two parts: the scan processor and OSL. The first was for gathering raw shipment data; it contained the logic to determine the success of a service. Both parts had been outsourced to a single company. Corporate guidelines required them to be client/server based and the company did not have enough experience in building such systems. However, outsourcing was not successful. The managing director explained:

> The system was developed in a remote location to MTC ... Dynamics of MTC change rapidly. This group was detached from MTC and stayed isolated ... If you cannot work with them face to face, the project can lose focus and not stay current with the business demand.... In every outsourced project that I am aware of, which is about half a dozen, the ones that were successful worked with MTC on-site.

The director added that the service provider had problems due to its rapid growth. One problem was that the two components were designed to be highly integrated and that this reduced flexibility. Another was the performance related technical problems due to the hardware platform. The project leader argued that MTC was not ready to take over the system from the vendor because too much training would be required. She added that technical requirements were not well defined due to MTC’s inexperience with client/server and object-oriented technologies and inadequate requirements, which had been gathered by the vendor.

In December 1997, the systems were redesigned and the components decoupled. The project scope was also expanded to include reengineering the way that shipment information is processed. This new project, called Collection, was a set of infrastructure projects that supported down-line shipment management applications. The redesign and other phases of development were conducted centrally at one site in the US. Some requirements were gathered from the existing system through code diving. A central committee (consisting of representatives from departments such as operations, sales, legal, and international and located at another US site) was formed to gather requirements. The international business representative coordinated with subject matter experts from the various regions. During the development process, this committee and the development team worked closely. A prototype was built and benchmarked for performance and tests in parallel to the existing system were conducted to ensure proper data handling.

5.4. Case 4: global revenue system

The global revenue system (GRS) was planned to be the single source of revenue information for MTC. It was viewed as a strategic project, supporting quick introduction of new services. The vision statement defined GRS as “an end to end redesign process that focuses on the revenue cycle from the point of origin, package processing and final payment.” The project aimed at reorganizing the processes increase controls on revenue from all lines of business and to reduce the cost of processing. Providing common processes around the world, flexible billing options, and the capability to quickly add new services and change existing ones were among the major objectives of the system. GRS was designed to be a point-of-sale based solution; its focus was on getting accurate and complete information about the customer and the shipment at or before the point of initial contact. Downstream automated controls were being implemented to prevent and detect errors and anomalies in processing. The revenue data was also reconciled with operational data to make sure that all services to customers were invoiced.

GRS is a very large, multi-hundred million dollar system that employs a few hundred developers and user representatives. The transition from existing systems was carried out gradually. The vision of GRS was created in 1996 and the first release to production took place in mid 1997. It was developed using an incremental approach, where an increment was released to production every 3 months. The managing director, the advisory Committee, and the project management team oversaw the project and established the scope for each increment. A waterfall methodology was used. Requirements gathering and analysis were conducted by a business liaison group. After the development teams completed their design and coding, a separate testing team tested the code to be released to production.

MTC formed a temporary partnership with a consulting company, Business Solutions (BS), to develop GRS. BS provided the great majority of the developers at the start while MTC hired and trained employees. A partnership was formed with the vendors whose products (e.g. servers, middleware, database, etc.) were used. Through the assistance of these vendors, a technical benchmark was conducted. The objective was to investigate whether the technical architecture could support the volume and performance requirements and to determine the required hardware for production.

The partnership with BS was by far the largest and longest. BS had people with the required technical skills and had developed similar large distributed systems. Their involvement was expected to reduce risks of failure associated with new technology and the size of the project. The plan was to phase out employees of BS as MTC took over the project. MTC provided BS with requirements of deliverables to be developed. When a problem occurred in production, the party responsible for the problem was expected to fix it using its own resources. The phase out of BS' employees started one and a half years after the first release of GRS and was completed by mid 2000.

GRS used a "core versus local" development strategy. Core components were common to all regions. The location of development for functionality that crosses regions but was not core was to be determined on a case-by-case basis. The maintenance of the local code was to be performed in the region that owned the code. One managing director stated:

> The biggest challenge is getting agreement globally. We would like to come up with 80% core and 20% local functionality. [Getting an agreement] is difficult even in the US but much more difficult globally due to communication problems, time differences, cultural, legal and other issues... .

The European and Canadian regions were involved more than a year after the project started. They had dedicated people who worked on gathering requirements, designing, and coding the local functionality. The headquarters business liaison group had a separate team that interfaced with these regions. It had individuals that focused on a single region to help in all phases.

The managing director of the user liaison group indicated that purchasing a package was dismissed due to stringent performance requirements of GRS and that the packages could not come close to supporting them. Further, the functional requirements (such as rating of a shipment) were unique and complex. Completely outsourcing the system was not possible, said the manager of the software architecture team, because the company wanted to internalize expertise in the new technology.

6. Discussion

We came up with 32 salient findings which are numbered in this section.

6.1. Package software

Purchasing package software was one of the first development strategy examined in all projects. Cost effectiveness, achieved through economies of scale by the vendor, is the major underlying reason for acquiring them (1). There was a consensus that package software is cheaper than all other alternatives and maintenance costs are perceived to be less with them (2). They are also faster to implement (especially when unmodified) (3). The commonality of the requirements across the organization seemed to be a prerequisite for package software (4). Nevertheless, it is practically impossible to find packages that meet all of the requirements of a company. To be successful in using a package software strategy, change and expectation management often becomes crucial.

Another option is to ask the vendor to modify the software. However, this generally involves high cost (5). The existence of enhancements not incorporated in the standard product would also decrease the level of support provided by the vendor or require additional contract maintenance (6).

6.2. Outsourcing

When package software is not an option, custom building is the only choice, involving a software vendors or in-house effort. Increasing management frustration with the IT department provides one reason for the rise of outsourcing (7). Another explanation of the increase in outsourcing is its increasing popularity in the business world (8).

OSL management chose outsourcing over internal development because they did not have access to required skills in new technologies (9). In cases where system development is outsourced but maintenance is not, it is crucial that the customer understands the design and code developed by the vendor.

Requirements gathering and preparing the outsourcing agreement are important challenges. The system developed by a vendor does not generally allow easy addition and deletion of services and does not have the performance required to support large transaction volumes. Since MTC frequently added, changed, and deleted services, both of these drawbacks were major problems with outsourcing (10).

6.3. Parallel development

Parallel development strategy is often selected because the region that starts the project does not have enough resources to complete it (11). One of the major reasons that a co-development attempt is fruitless is lack of accountability (12); according to the technical lead of the project:

> ... You really need to have one person that would be the sponsor and the delegator ... We had our own little teams and there wasn't one person running both teams, saying "you guys are going to do this and you guys are going to do this ...

Problems of accountability were exacerbated when complete responsibility of developing the system was taken from the UK team and the reasons given were not convincing. The resentment not only created a negative working environment but also the UK to shift their priority to other tasks. Further complicating the communications issue were cultural differences (13). It was suggested that the differences in time and development methodologies contributed to the frustration (14).

6.4. Core versus local development

After parallel development, GCS management decided to implement a core versus local strategy. The team manager argued that it separated the responsibilities of the regions more distinctly and clearly (15). Less coupling was later expected to expedite development, since there would be less need for communication (16). This was expected to decrease cost of coordination. The differences in development methodologies was another reason that loose coupling was welcomed (17).

The managers also considered using core versus local strategy. The core part of the system was expected to be much larger than the local parts (18). However, determining what is core what is local is not always easy (19) and sometimes regions do not have resources for local tailoring (20).

6.5. Central development

MTC had a history of developing systems centrally, because regions did not have adequate resources to build systems (21). Also some requirements were relatively uniform across the international regions; thus, this strategy was appropriate (22). The preferred strategy of some project members was single site development because they believed that face-to-face communication improved the effectiveness of the team and increased collaboration among team members (23).

6.6. Joint development with vendor

Some systems used joint development with vendor strategy. GAR is a package solution and management of the project involved the package vendor to increase the probability of success and to expedite its implementation (24). It allowed access to additional resources not available internally (25). Since the vendor knew the product very well, it could help smooth the transition to the new system by providing product expertise, training users and developers, and helping develop custom reports and software exits.

Joint development with vendors was found to be relatively challenging due to difficulties in the assignment of accountability (26). Large amount of time was wasted in determining whether the defects in production software were caused by vendors or the host company. Another disadvantage of this strategy is its cost (27).

6.7. Multinational design team (MDT), unmodified package software acquisition (UP), and best-in-firm software adoption (BIF)

Cost and availability of regional resources were major obstacles to utilize the MDT strategy (28). Getting resources away from their daily jobs for an extended period of time is a challenge and a cause for high turnover. Based on the projects, it is unlikely that a package software would be available or can be used, as is, for large-scale multi-country global projects (29).

None of the cases had systems in place that would meet the needs of the entire worldwide operations. In some cases, a BIF application provided the starting point for a global application. Even when processes were similar, there were still major differences. Thus, it appears that BIF is not a viable strategy for the development of complex and large-scale global systems (30).

6.8. Some overall observations

Depending on its complexity, global organizations may need multiple strategies for system development (31). Different strategies may be suitable for different phases of the life cycle of the same project. Also strategy selection is not necessarily a single static decision made at a single point in time. Organizations may need to continually reevaluate their resources and problems, and may have to dynamically readjust the development strategy (32).

7. Conclusions

Classifying global system development projects as simply larger versions of their domestic counterparts is an oversimplification. They are huge and complex, involving issues from technical to behavioral. Their degree of difficulty seems to multiply with size. Delays and budget overruns are commonplace. Eight strategies for developing GIS were identified in the literature and one was added during the investigation. A guiding framework was developed representing the underlying factors that may affect the choice of a strategy. The framework provided the backdrop for four qualitative studies that provided contextual results about development strategies used in global organizations. Thirty-two findings were explicated from the case studies. We understand that our research suffers from the limitations of case and qualitative research; while we found many interesting results, there may be relationships that were not present in the cases.

References

[1] U. Apte, Globalization of information systems outsourcing; opportunities and managerial challenges. Edwin L. Cox School of Business, Southern Methodist University, in: S.C. Deans, J. Jurison (Eds.), Information Technology in a Global Business Environment, Boyd and Fraser Publishing Company, Danvers, 1996.

[2] J.M. Burn, H.K. Cheung, Information systems resources structure and management in multinational organizations, in: P. Palvia, S. Palvia, E. Roche (Eds.), Global Information Technology and Systems Management: Key Issues and Trends, Ivy League Publishing, Nashua, 1996, pp. 239–322.

[3] J.F. Cash, F.W. McFarlan., J. McKenney, Corporate Information Systems Management: the Issues Facing Senior Executives, third ed., Irwin, Homewood, IL, 1992.

[4] J.F. Cash, F.W. McFarlan, J. McKenney, L. Applegate, Corporate Information Systems Management: Text and Cases, third ed., Irwin, Homewood, IL, 1992.

[5] R.W. Collins, L. Kirsch, Crossing Boundaries: the Deployment of Global IT Solutions, Pinnaflex Educational Resources Inc., 1999.

[6] W. Delone, E. McLean, Information system success: the quest for dependent variable, Information Systems Research 3 (1) (1992) 60–95.

[7] G. DiNardo, Regional banking and credit card processing at Citibank Consumer Banking and its global implications, in: P. Palvia, S. Palvia, E. Roche (Eds.), Global Information Technology and Systems Management: Key Issues and Trends, Ivy League Publishing, Nashua, 1996, pp. 577– 591.

[8] W.J. Doll, G. Torkzadeh, The measurement of end-user computing satisfaction, MIS Quarterly 12 (2), 1988, pp. 259– 274.

[9] K. Ewusi-Mensah, Critical issues in abandoned IS development projects, Communications of the ACM 40 (9), 1997, pp. 74–80.

[10] S. Ghoshal, N. Nohria, Horses for courses: organizational forms for multinational corporations, Sloan Management Review 34 (2), 1993, pp. 23–35.

[11] G. Hamel, Leading the Revolution, Harvard Business School Press, Boston, MA, 2000.

[12] J.D. Herbsleb, A. Mockus, An empirical study of speed and communication in globally-distributed software development, IEEE Transactions on Software Engineering 29 (3), 2003, pp. 1–14.

[13] J.D. Herbsleb, D. Moitra, Global Software Development, IEEE Software, 2001, pp. 16–20.

[14] G. Hofstede, Culture’s Consequences: International Differences in Work-Related Values, Sage, Newbury Park, CA, 1980.

[15] B. Ives, S. Jarvenpaa, Applications of global information technology: key issues for management, MIS Quarterly 15 (1), 1991, pp. 33–49.

[16] B. Ives, M. Olson, M.S. Baroudi, The measurement of user information satisfaction, Communications of the ACM 26 (10) (1983) 785–793.

[17] B.D. Janz, N.P. Vitalari, J.C. Wetherbe, Emerging best practices in global systems development, in: P. Palvia, S. Palvia, E. Roche (Eds.), Global Information Technology and Electronic Commerce, Ivy League Publishing, Marietta, GA, 2002, pp. 332–355.

[18] J. Karimi, B. Konsynski, Globalization and information management strategies, Journal of MIS 7 (4), 1991, pp. 7–26.

[19] P.G.W. Keen, G.S. Bronsema, S. Zuboff, Implementing common systems: one organization’s experience, Systems, Objectives, and Solutions 2 (1982) 125–142.

[20] C. Kim, S. Westin, N. Dholakia, Globalization of the software industry: trends and strategies, Information and Management 17 (4), 1989, pp. 197–206.

[21] V.S. Lai, Issues of International Information Systems Management: a perspective of affliates, Information and Management 38 (4), 2001, pp. 253–264.

[22] L. Markus, D. Robey, Information technology and organization change: causal structure in theory and research, Management Science 34 (5), 1998, pp. 583–598.

[23] W. McFarlan, Issues in global outsourcing, in: P. Palvia, S. Palvia, E. Roche (Eds.), Global Information Technology and Systems Management: Key Issues and Trends, Ivy League Publishing, Nashua, 1996, pp. 352–364.

[24] C.J. Meadows, Globalizing software development, Journal of Global Information Management 4 (1) (1996) 5–14.

[25] P. Nelson, W. Richmond, A. Seidmann, Two dimensions of software acquisition, Communications of the ACM 39 (7), 1996, pp. 29–35.

[26] P. Palvia, A dialectic view of information systems outsourcing: pros and cons, Information and Management 29, 1995, pp. 265–275.

[27] S. Palvia, K. Lee, Developing and implementing global IS: lessons from Seagate technology, in: P. Palvia, S. Palvia, E. Roche (Eds.), Global Information Technology and Systems Management: Key Issues and Trends, Ivy League Publishing, Nashua, 1996.

[28] M.Q. Patton, How to Use Qualitative Methods in Evaluation, Sage, Newbury Park, CA, 1987.

[29] A. Ragowsky, M. Stren, How to select application software, Journal of Systems Management 46 (5), 1995, pp. 50–54.

[30] H.R. Rao, K. Nam, A. Chaudhury, Information systems outsourcing, Communications of the ACM 39 (7), 1996, pp. 27–28.

[31] R. Ravichandran, U. Nazim, Offshore systems development, Information and Management 24 (1), 1993, pp. 33–40.

[32] D. Robey, A. Rodriquez-Diaz, The organizational and cultural context of systems implementation: case experience from Latin America, Information and Management 17, 1989, pp. 229–239.

[33] B. Shore, Cultural Asymmetries between Headquarters and Foreign Subsidiaries and their Consequence on the Integrative Role of Information Technology, in: P. Palvia, S. Palvia, E. Roche (Eds.), Global Information Technology and Electronic Commerce, Ivy League Publishing, Marietta, GA, 2002, pp. 161–177.

[34] M.A. Smith, S. Mitra, S. Narasimhan, Offshore outsourcing of software development and maintenance: a framework for issues, Information and Management 31 (3),1996, pp. 165–175.

[35] R.K. Yin, Case Study Research: Design and Methods, second ed., Applied Social Research Methods Series, Sage, 1994.