8-15-1997

Globally Disaggregated Software Development: Value Chain Perspective and Effort Estimation

Sanjay Gosain
University of Southern California

Globally Disaggregated Software Development: Value Chain Perspective and Effort Estimation

Sanjay Gosain
Information and Operations Management Department
Marshall School of Business
University of Southern California

Introduction

Software is one of the enabling factors for many businesses and accounts for a large share of information system costs. Outsourcing has emerged as an institutional arrangement aimed at achieving efficiency in operations and has been applied to software development as well. Computer programming services are the most frequent type of outsourcing after management services [Dbisna, 1996]. A large number of firms have already outsourced application development. Clearly, there is pressure on firms to reorganize their information systems and software development activities. At the same time, problems with software systems are common and well-publicized and software projects are prone to cost and time overruns.

Economic and Technological Trends

We note three main trends that have led to the globalization of the software development:

Personal Computing

Globalization of consumer preferences has been predicted due to economic reasons, with lower cost and technologically superior products triumphing over regional and national barriers. Levitt [1983] has argued for companies to move from customized products to globally standardized products that are advanced, functional, reliable and low-priced. In the case of software, the process of globalization has been aided by convergence of standards towards common hardware platforms. Jones [1994] notes the differences in the economics of software development between mainframe and personal computers. Kim, Westin & Dholakia [1989] note that the falling hardware/software ratio implies a decline in capital intensity and a rise in human capital intensity.

Software Components (objects) and reuse

Software reuse has long been felt to be the way to increase software productivity. The basic principles of systematic reuse are that components be designed for a range of products, components be designed with reuse in mind and common parts and customizable parts of components be distinguished [Cleveland, Fertig & Newsome, 1996]. It has been suggested that software component reuse-based development will lead to changes in organization of software development. Component developer and application developer roles will be segregated and a firm could make choices on developing components (labor-based approach) or acquiring them (capital-based approach).

Communication networks and standardized environments

The emergence of global communication networks such as the Internet and standardized computing environments favors the globalization of the software development industry. The client-server computing paradigm allows for computing capabilities to be partitioned over space and across organizations. The emergence of the Internet has provided a seemingly universal computing medium.

Software Development Paradigms

The nature of software development is constantly being altered by technological changes. Some dimensions of likely change are highlighted in the table:

Dimension	Past	Future Scenario
scope	functional area automation	enterprise-wide integration
development	decomposition	composition
life cycle	waterfall	iterative
change over time	static	dynamic reconfiguration, reuse
architecture	stand-alone customized	distributed universal client-server
programming	large teams	tool-augmented small teams
sharing	code	application components
critical resources	skilled manpower, hardware	skilled manpower, component libraries
core goal	cost cutting/competitive edge	enabling business
disaggregation	hierarchical	network
methods	structured	object-oriented
coordination	communication, documentation	standards, collaborative

Disaggregation of the Value Chain

The advantages of disaggregation are cost reduction, access to skilled professionals, faster development and access to growing markets while the disadvantages are problems of coordination, intellectual property rights violation, lack of control, government policy changes, difficulty in managing cultural diversity and unstable economic, social and political environments [Apte & Mason, 1995].

Using the Indian software industry as an example, Heeks[1991] points out to the lower skill level programmer-heavy nature of tasks delegated to Indian firms, and the predominantly on-site location of programmers, as evidence that labor costs are not the only factor that influence disaggregation and the role of factors like trust and risk is important.

Conventional research has viewed disaggregation as a hierarchical handing down of tasks to organizational entities that can perform them in a more efficient manner. We view disaggregation as being the constitution of a value chain across many entities, with maximum value addition being the key goal.

Theoretical Perspectives

Figure 1 shows the theoretical perspectives that can be used to explain the structuring of value chains. These perspectives are used to arrive at various independent variables and their likely impact.

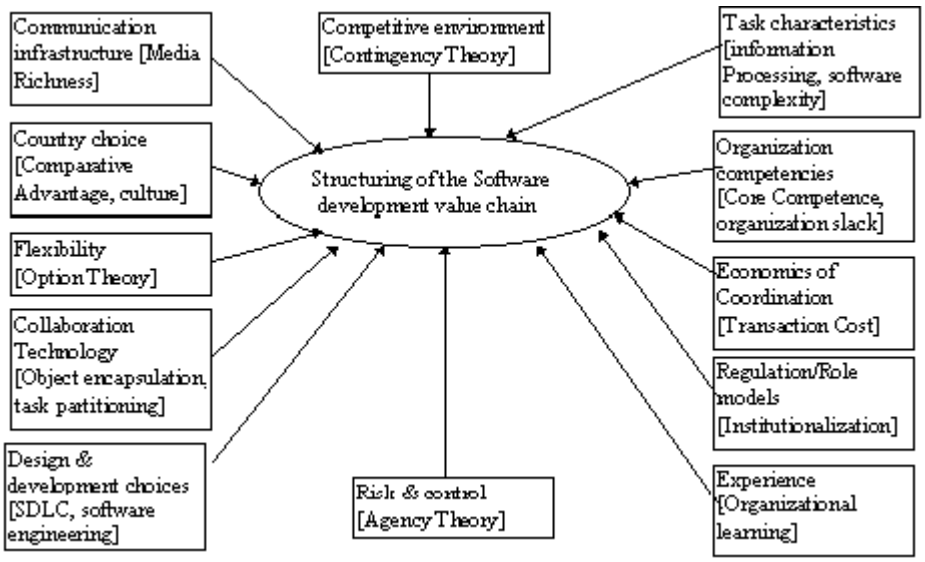

Fig 1 : Factors that impact the choice of collaborative model and associated theories

A Causal Model for Value Chain Structure

We utilize the above perspectives to explain choice of value chain structuring using the following elements (Fig 2.):

Organization Behavioral - Trust, learning, adaptation

Task factors - Complexity, methodology

Economic Factors - Scale, country, transaction costs

Coordination Factors - Media, task partitioning, structuring

Strategic Considerations - Competencies, Resources, risk & control, flexibility

Environment - Institutions

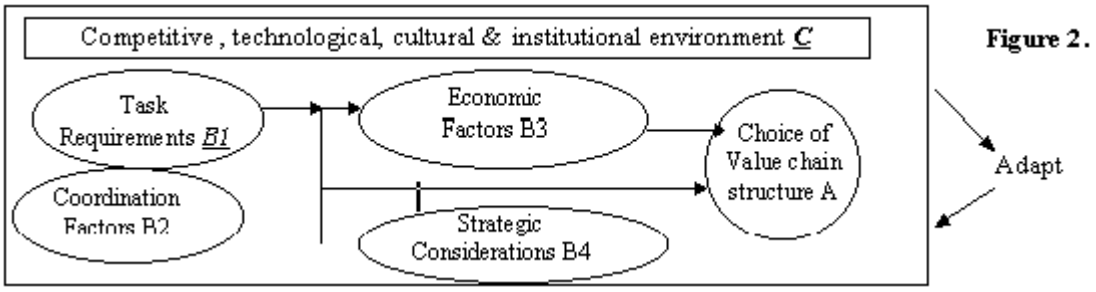

Figure 2.

Specification for Effort Estimation

The reason why we need to extend the well-known cocomo model [Boehm, 1981] is that a number of different factors apply in the disaggregated context that may not be important in conventional development:

1. **Country differences**

When software development is carried out across several countries, we need to take country differences into account in order to develop models that provide greater explanatory power. Differences between nations that affect software development cost estimates may take the form of:

- differences in cost structure
- differences in software programmer or analyst productivity
- differences in infrastructure availability.
- cultural differences

1. **Configuration of software development activity**

The software development life cycle may be configured in several alternate ways. The allocation of development activity to different countries may be done horizontally, that is, allocating functional modules to different development teams in different countries or vertically, by allocating different development phases to different development teams. Sometimes a composite approach may be followed, by assigning specific lifecycle stages in specific modules to specific development teams.

1. **Coordination Overhead**

When software development activity is distributed across large physical distances between different teams, coordination of the development process is extremely vital. This factor plays a role in stand-alone development as well but since the overhead placed by communication requirements is not very significant or variable by situations it has not been explicitly been included in the Cocomo model. In a multi-country scenario and especially with cultural and time-zone differences this is very significant.

The first term in the proposed specification reflects the development effort as a summation of the individual development effort for the components produced in each country, while the second term represents the added communication overhead.

Man-month effort = [f_i(dev_mode) KDSI-eq^(g_i(dev_mode) * h_i(prod_attrib, comp_attrib, pers_attrib, project_attrib, country_attrib] + q_{ij}[prod_attrib, KDSI-eq, dev_mode, pers_attrib, comm_media, $country\_attrib_i$, $country\_attrib_j$]	
where,	
dev_mode	software development mode chosen (organic or embedded)
KDSI-eq	equivalent number of delivered source instructions(in thousands)
prod_attrib	a vector of parameters reflecting software product attributes
comp_attrib	a vector of parameters reflecting computer attributes
pers_attrib	a vector of parameters reflecting personnel attributes
project_attrib	a vector of parameters reflecting project attributes
f_i, g_i and h_i	specifications of functional forms for country I
q_{ij}	specification of functional form for communication overhead between countries i & j
comm_media	richness of communication media used

$country\_attrib_i$	vector of parameters reflecting country attributes

Estimation of the Model

The extension proposed requires that the basic Cocomo model (the first term) be estimated for each country and the coordination overhead factor (second term) be estimated for each pair of countries involved in the development activity. In order to estimate the parameters, data from disaggregated software projects needs to be collected and statistically analyzed.

There is some evidence to expect that the parameter estimates vary across countries. Marouane & Mili [1991] found, for instance that Tunisian software projects provided different parameter estimates than the original Cocomo estimates which are based on US data.

Definition of Country Vector

The impact of country factors mentioned earlier manifests itself in two ways in the model - through differences in productivity that impact the first term and through added communication overhead that inflates the second term. By adding country_attrib as one of the inputs to the h_i function and also giving a country subscript to the function we provide a general form of the model implying that functional form could different.

The country attribute vector has the following parts:

- *Communication and computing-related Infrastructure*

The availability of a reliable communication network and computing facilities enable software production. Indirectly, the availability of a trained pool of software developers also depends on infrastructure such as training institutions.

- *Manpower productivity*

Classical economics has emphasized the importance of factors of production in analyzing national competitiveness. Given, the labor intensive nature of software development it is apparent that nations with low labor costs will be attractive as sites for development, given similar skill and productivity levels. Porter [1990] argues that factor conditions, demand conditions, related and supporting industries and firm strategy, structure and rivalry are determinants of national competitive advantage. Apart from economic reasons that favor a shift to developing and newly industrialized countries because of lower wages and the availability of human capital, there are barriers in the form of cultural differences, limited local markets, insufficient research and development, inadequate communication facilities and delayed access to new technology [Kim, Westin & Dholakia, 1989]. In addition, firms operating in a foreign country run the risk of skills being acquired by domestic firms or changes in government policies. The emergence of countries in South America, Asia and Eastern Europe as sources of manpower has been recognized [Press, 1991]. However, opportunities arising for this reason will be qualified where a high level of skill and capital-intensity is required [OECD, 1989].

- *Cultural factors*

Cultural differences are postulated to play a role in the structuring of global software development value chains. Hofstede [1983] identified four independent dimensions of differences among national value systems - power distance (large vs. small), uncertainty avoidance (strong vs. weak), individualism vs. collectivism and masculinity vs. femininity. A later study added a fifth dimension of a short term vs. a long term orientation of life and work. Differences in national culture attributes are likely to affect positions of

the negotiating parties on how alliances will be structured. In order to reduce the possibility of culture clashes and dissonance, firms will try to ally with partners from compatible cultures. For countries with large cultural differences, full acquisitions would be avoided. Also, the value chain structuring will tend to reflect cultural traits - High uncertainty avoidance could imply more communication and formalization of work procedures. Small power distance tolerance could lead to more delegation of authority.

An empirical study, "Worldwide Benchmark Project" [CIO, 1997] has demonstrated differences at corporate, regional and national levels in terms of productivity, quality and capability.

The various factors considered in the model have been derived from the Cocomo specification or proposed based on normative expectations from theory, and need to be further refined through an examination of cross-national software development projects.

[References available on request]