INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

A Decomposition and Guided Simulation Methodology for Large-Scale System Design: A Study in QoS-Capable Intranets with Fixed and Mobile Components

Sudip Bhattacharjee, Hong Zhang, R. Ramesh, Dee H. Andrews,

INFORMS Journal on Computing
Vol. 19, No. 3, Summer 2007, pp. 429–442
ISSN 1091-9856 | EISSN 1526-5528 | 07 | 1903 | 0429

DOI 10.1287/ijoc.1050.0173

A Decomposition and Guided Simulation Methodology for Large-Scale System Design: A Study in QoS-Capable Intranets with Fixed and Mobile Components

Sudip Bhattacharjee
School of Business, University of Connecticut, Storrs, Connecticut 06269,
sbhattacharjee@business.uconn.edu

Hong Zhang
College of Business Administration, Missouri State University, Springfield, Missouri 65897,
hongzhang@missouristate.edu

R. Ramesh
Department of Management Science and Systems, School of Management, State University of New York at Buffalo,
Buffalo, New York 14260, rramesh@acsu.buffalo.edu

Dee H. Andrews
Air Force Research Laboratory, Mesa, Arizona 85206, dee.andrews@mesa.afmc.af.mil

Integrated design of a large-scale distributed system is challenging due to multiple conflicting design objectives, a large number of stochastic parameters, and a combinatorially large design space. Current approaches using decomposition and analysis of isolated subproblems could lead to suboptimization. We develop a framework for integrated design that combines analytical modeling with embedded simulations for large-scale infrastructure planning. The framework uses *problem decomposition* and a *guided simulation* methodology, which uses concepts of *extrinsic* and *intrinsic* guidance. These are integrated in a comprehensive framework and used to interlink the subproblems and guide the design process towards achieving overall objectives concurrently. A case study demonstrates viability of the framework. The methodology is efficient and flexible, and can be adapted for other problems where resource sharing and integrated system design is critical.

Key words: multiple-criteria decision analysis; large-scale infrastructure design; decomposition; guided simulation; guidance mechanisms; metaheuristics; simulation optimization; quality of service
History: Accepted by Ramayya Krishnan, former Area Editor for Telecommunications and Electronic Commerce; received September 2002; revised August 2003, July 2004, November 2005; accepted December 2005. Published online in *Articles in Advance* July 20, 2007.

1. Introduction

A large-scale distributed system involves geographically dispersed users supported by many distributed applications and databases. The distributed components must be coordinated to satisfy a diverse set of objectives. These include high performance (e.g., low access delays), high interoperability, persistent availability, security, low costs of deployment and maintenance, and high utilization. Typically, the design of such systems begins with a high-level specification of system components and performance requirements. Next, these specifications are employed in some design framework to derive performance-effective high-level system architectures. These specifications could involve many operational parameters, design options, and performance metrics. Many of these parameters are stochastic and often lead to a combinatorial explosion in the design space. Furthermore, the system objectives are usually conflicting and closely related. Hence, the tradeoffs among all metrics should be simultaneously considered in designing a total system configuration; a piecemeal approach could easily lead to suboptimization. This is usually compounded by difficulty in determining many of the metrics exactly. Hence, completeness and soundness of any exact optimization procedure for comprehensive system configuration could be difficult to establish. As a result, exact optimization could quickly become intractable and may not be viable in large-scale system design. Consequently, heuristics are both preferred and widely used (Ho et al. 1992, 2000).

We develop a novel approach to such design problems, integrating analytical with simulation modeling, with a combination of exact and heuristic techniques.

While simulation modeling yields a robust and practical framework to solve design problems with difficult-to-determine stochastic behavior, analytical modeling decomposes a large problem in a combinatorial design space into identifiable and interlinked subproblems, and guides the simulation models toward effective overall solutions. This typifies simulation-optimization research where tabu search and other analytic and heuristic techniques are integrated with simulation to solve intractable optimization problems (April et al. 2003, Glover et al. 1999, Fu 2002, Fu et al. 2000, Shanthikumar and Sargent 1983). We demonstrate the methodology using a real-world case study configuring a component-based architecture for a large corporate intranet with fixed and mobile users.

We present a decomposition-based design framework that combines analytical modeling with embedded simulations for large-scale infrastructure planning. The design problem is first *decomposed* into a set of interlinked but separate subproblems. Specific techniques are used for the subproblems and the solutions are linked into a comprehensive framework for integrated design. We introduce the notion of *guided simulation* in solving some of the decomposed components of the overall problem, which also arrives at a broad design configuration of the overall environment. Since simulation is expensive, solutions to the decomposed subproblems and results of intermediate simulations guide the scope and thrust of subsequent simulations so that effective design configurations can be efficiently determined. We introduce the concepts of *extrinsic* and *intrinsic* guidance mechanisms. Extrinsic guidance mechanisms provide specific environments for simulations determined analytically from the solutions to some of the subproblems. These solutions constitute a set of parameters that are *exogenous* to the simulation scenarios. Intrinsic guidance mechanisms refine the simulation scenarios using internal system performance evaluations determined from the simulation experiments. The refinements are captured in terms of a set of *endogenous* parameters. Together, the two mechanisms derive cost- and performance-effective system configurations while minimizing simulation effort. While *extrinsic guidance* focuses the search for a best configuration from a combinatorially large design space, *intrinsic guidance* minimizes the complexity in evaluating performance metrics and the tradeoffs among them in a stochastic operational environment. The guided methodology can also be used for system benchmarking.

The unified framework contributes to research and practice on large-scale system design. The literature suggests the need for better integration of advanced optimization and simulation procedures that lie within the extremes of highly specialized algorithms for a particular purpose and very generic ones that are exceedingly time-consuming (April et al. 2003; Fu 1994, 2002; Laguna and Martí 2002). The proposed methodology is a realization of this goal. Using problem decomposition, we exploit solution techniques to specific subproblems. Using the linkages among these solutions, we develop a guided simulation strategy for the overall solution to the design problem that incorporates learning mechanisms derived from both within and outside the simulation environments. To our knowledge, this is a first effort to develop an integrated framework for designing large component-based intranet architectures.

Section 2 presents a brief overview of the design environment, followed by the problem and decomposition strategy. Sections 3 and 4 detail the extrinsic and intrinsic guidance mechanisms, respectively. The guided simulation framework is presented in Section 5. Section 6 presents an implementation of the proposed methodology using a case study, and also develops a set of design guidelines and insights. Finally, Section 7 summarizes and discusses future research directions.

2. Design Environment

Figure 1 presents a brief overview of the enterprise architecture associated with the *distributed system design problem* (DSDP). *User architecture* includes various functional groups spread across geographical locations that access distributed data and application components over a network infrastructure. *Data architecture* involves data partitioning, clustering, and allocation across an enterprise (Umar 2003). *Component architecture* refers to software component model development, identification and aggregation of primitive components into business components, and component allocation across the network infrastructure (Fan et al. 2000, Purao et al. 1998, Umar 2003). *Network infrastructure* defines the physical network model that supports the data and component allocations (Umar 1997).

Optimal design of the data architecture over a distributed enterprise network is NP-hard (Chari 1996) and literature in this is area extensive (Apers 1988, Chen and Akoka 1980, Chu 1969, Ghosh et al. 1992, Hevner and Rao 1988, Jain 1987, Purao et al. 1999). The motivation for the proposed decomposition strategy essentially arises from these studies.

A software component (a *component*) is a self-contained piece of code that captures a business process, e.g., an order-payment system. It has a well-defined interface and can be developed, delivered, installed, and run independently (Herzum and Sims 2000). It is also integrated with other components to fulfill larger enterprise requirements. A key design issue involves locating copies of such components across the network to optimize performance.

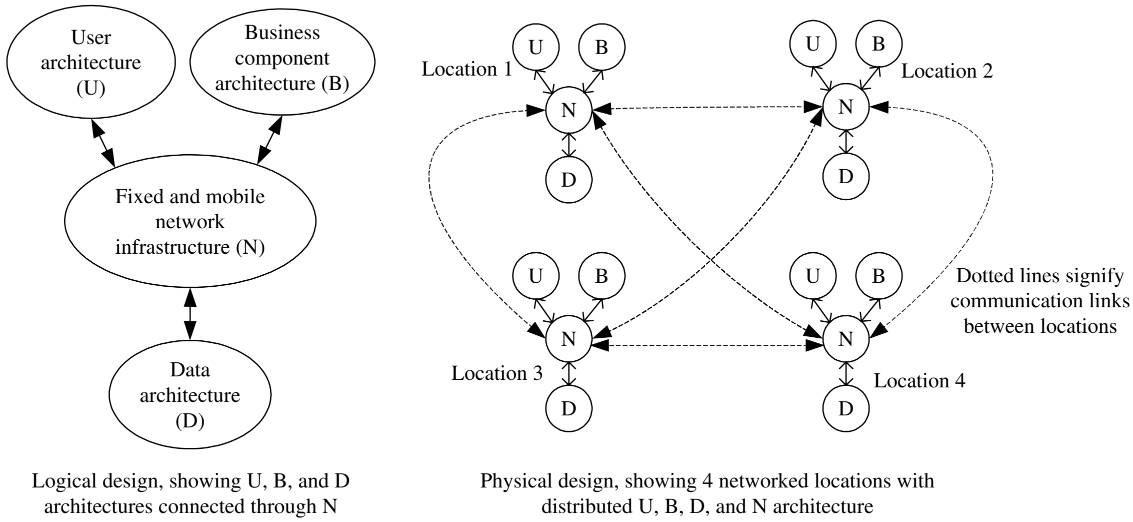

Figure 1 **Enterprise System Architecture: Logical and Physical Design**

In network-infrastructure design, fixed networks link users, components, and data sources at various locations. These are augmented using mobile technologies that inter-connect with fixed networks through specialized interfaces, commonly known as *mobile computing server/switches* (MCSS) (Dhawan 1997). Figure A1 in Appendix A of the Online Supplement to this paper on the journal's website is a conceptual view of the physical infrastructure of a mobile computing system and its inter-connection with a fixed network. An MCSS typically includes communication services, protocol conversion, gateways to legacy systems, and transaction switching (Dhawan 1997, Varshney and Vetter 2002). Designing a fixed network and determining MCSS capacity and access locations are key elements for achieving required performance objectives (Kanter 2003, Ma and Zarki 2002, Varshney and Vetter 2002, Wu 2001).

Quality of service (QoS) is an important performance objective for systems, especially where multimedia-based applications are delivered over the network. Designing a QoS-capable large-scale system for a particular process is itself a nontrivial task. When multiple components, users, processes, and data sources are included, along with various conflicting design objectives, the problem is compounded through their complex interactions. In this study, QoS levels are determined and specified based on user requirements and expectations (Katchabaw et al. 2000, Talukdar et al. 1999), as opposed to low-level specification based on resource requirements (Chen and Akoka 1980, Vogel et al. 1994, Wang et al. 1996). Other determinants in enterprise-system design such as availability, reliability, and scalability are mostly tackled through clusters and other redundant fail-over mechanisms, and are not included here.

2.1. Design Objectives

The design objectives in DSDP are simultaneously to (a) *minimize all relevant system latencies* (QoS metrics), (b) *maximize system utilization* (resource utilization metrics), and (c) *minimize deployment and operational costs* (cost metrics). The QoS metrics are

- *Read latency* (z_1): Time taken to execute a read query, counted from the point at which it was first scheduled until the point when the result is returned to the user, including queueing delays. The maximum allowable read latency is specified for each component.
- *Write latency* (z_2): Specified and determined similarly as in z_1, for write operations, including data insertion, modification, and deletion.
- *Multimedia latency* (z_3): Critical, especially for video applications, measures average frame rates.

The resource utilization metrics are

- *Database utilization* (z_4): Measures average utilization of database systems. A database architecture contains one or more databases over various locations. This metric is derived from the percent of time each database system is active in supporting its underlying business process needs such as queries, updates, analyses, and report generation.
- *Component utilization* (z_5): Measures the average utilization of the application software components in the overall architecture. This metric is derived from the percent of time each component is busy executing requests from users and routing requests to database systems or other components.
- *Network utilization* (z_6): Measures the average utilization of a network. This metric is defined as the ratio of network-resource usage time to the total observation time. This is calculated using packet size and processing and transmission speeds across the

link, and includes the latency across the transmission link.

More fine-grained models of the utilization metrics are possible and can easily be incorporated in the proposed design framework. Finally, z_7 measures the costs of system deployment, operation, maintenance, and remote communication.

DSDP involves multiple conflicting objectives. These objectives are denoted by the criteria set $CS_{DSDP} = \{z_1, z_2, z_3, -z_4, -z_5, -z_6, z_7\}$ where each criterion requires minimization. These criteria conflict: minimizing z_1 and z_2 conflicts with maximizing z_4 and z_5, and minimizing z_3 conflicts with minimizing z_7. In large-scale systems design, these metrics are often complex stochastic functions of decision variables and system parameters. However in practice, target ranges of these metrics are usually available through separate benchmarking studies conducted by systems developers. The developers employ these ranges as formal system specifications and determine the design variables such that a resulting configuration would yield metrics falling in their respective target ranges (Gray 1993). The classical trial-and-error approach is used for this purpose in practice, but with no guarantee of finding a configuration with associated metrics in their desired target ranges. Designers adjust the range specifications as necessary and engage in tradeoffs among the metrics to find a feasible system configuration that best meets their needs.

Classical multicriteria decision making (MCDM) approaches are not appropriate here for several reasons. DSDP does not lend itself to standard methodological frameworks due to its lack of structure. Next, the problem size, stochasticity of problem parameters, and their interactions limit application of classical optimization methods. Furthermore, classical methods introduce several assumptions (e.g., concavity or even continuity of a utility function) that may be hard to establish. Therefore, simulation is used in practice, although with a brute-force trial-and-error approach. However, design practice in industry offers several insights. Using the idea of target ranges, the proposed method uses problem decomposition and guided simulations to derive effective configurations satisfying the target requirements. The decomposition exploits problem structure wherever possible so that they can be appropriately optimized. The guided simulations attempt to find compromise solutions to the overall problem by systematically exploring design configurations obtained from these optimizations. The guidance mechanisms provide progressive learning in this search. The goal is to find a set of configurations that yield the objective metrics in their required target ranges. The solutions thus obtained would all be acceptable, but with different tradeoffs among the objective metrics. Hence, although DSDP is a typical MCDM problem, the proposed solution strategy is different from classical MCDM methodologies.

2.2. Design Variables and Parameters

Appendix A in the Online Supplement provides a full description of the design parameters, decision variables, and models of the decomposed subproblems of DSDP. The parameters and variables are grouped into the four architecture components discussed earlier—user (U), business component (B), data (D), and network (N). A designer may wish to alter the parameter lists as required for a particular case, and this does not affect the proposed design methodology. While all the variable and parameter definitions are fairly straightforward, we briefly explain two of these.

A business component could access data from a local or remote source, and in turn may invoke other component(s) during execution. The access patterns depend on component design. Without loss of generality and for reasons of brevity, we restrict our analysis to only the communication between components and databases. Other different communication scenarios are possible and can easily be incorporated in the design framework. Next, several configurations of transaction processing (TP) ranging from light to heavy exist (Umar 1997). The degrees of transaction control, performance, and cost increase from light to heavy configurations. These options are designated as transaction model types and transaction coordination types in the variable definitions.

2.3. Problem Decomposition and Guidance Strategy

The general structure of DSDP is in Figure 2. The set of design objectives CS_{DSDP} is influenced by direct

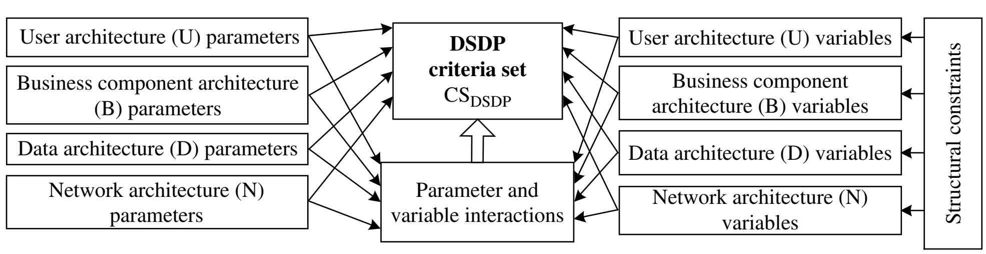

Figure 2 DSDP Model Structure

and interaction effects of various parameters and variables, as well as structural constraints on the decision variables. In practical systems design the exact functional forms of several of these effects will not be analytically determinable, in addition to nonlinearities in many variables, constraints, and objective metrics. Traditional decomposition strategies require a formal and complete statement of DSDP with all these effects included. Therefore, such approaches are not viable here. We adopt a decomposition strategy based on a logical sequence of design decisions as described below.

DSDP is first decomposed into five subproblems based on certain logically independent structures of components U, B, D, and N, and subsequently linked into an iterative solution framework. These subproblems are denoted as *domain formation* (DF), *mobile domain formation* (MDF), *resource allocation* (RA), *fixed network design* (ND), and *mobile network design* (MND). The rationale for this strategy is as follows. First, each subproblem affects others. Second, in many cases, these influences occur and consequently should be considered in certain logical sequences; e.g., while the user-group distribution over the geography of the network should be used to determine domains that affect component and data architectures, the reverse sequence of fitting users and processes to given component and data architectures could be suboptimal or even infeasible. Third, these subproblems have unique characteristics and requirements that should be specifically addressed within their respective scopes; for instance, users with fixed geographical locations differ from mobile ones. Hence we model the user architecture into subproblems DF and MDF where we form domains of fixed and mobile user groups, respectively. This provides a logical operational environment for subsequent resource allocation and network architectures that connect resources and users. Fourth, data architecture over a distributed network is a well-studied problem with a defined structure. Business-component architecture follows a similar structure over a distributed environment, and is closely tied to the data architecture of the organization. Hence, these two components are decomposed into subproblem RA. Finally, fixed and mobile networks have unique characteristics. A mobile network is usually designed over an existing fixed network. Hence these are decomposed into subproblems ND and MND, respectively. This decomposition strategy results in a set of subproblems with a clear domain definition for each problem and a logical sequence in which they need to be solved. This sequence implies that the outputs of a subproblem will be the inputs to all its successors.

Although the domain space for each of these subproblems is well-defined, the direct and interaction effects of their variables, parameters, and constraints on the overall performance metrics are ill-structured. The subproblems DF, RA, and MDF provide a baseline structure of domains and resource allocations. The subproblems ND and MND yield the overall performance metrics for a given baseline structure, network-resource allocation, and stochastic user behavior. The baseline subproblems are modeled analytically and solved accordingly. The problems ND and MND result in final design configurations and are solved through simulation. The proposed *guided simulation* approach employs two forms of guidance: *extrinsic* and *intrinsic*. *Extrinsic guidance* is provided by the analytical solutions to baseline subproblems DF, RA, and MDF. This guidance consists of a set of potentially good baseline structures that should be further explored using simulation. Using each fixed baseline structure, a set of *targeted simulations* is performed by selecting configurations within the domains of problems ND and MND. *Intrinsic guidance* is obtained from these experiments. The results of the targeted simulations empirically yield the direct and interaction effects of the design variables, parameters, and constraints on the performance metrics. The extrinsic and intrinsic guidance mechanisms constitute the learning components of the design framework. This framework presents a systematic search for design configurations that yield performance metrics in their target ranges. Several strategies for using the guidance mechanisms within an iterative scheme are also possible. Finally, the configurations that qualify with respect to the target range specifications are evaluated for dominance in terms of the performance criteria. The set of nondominated solutions is provided to the system designers for tradeoff evaluations and final configuration selection.

We use the following notation. A full system design configuration is $\Phi = \langle \varphi_{\mathrm{DF}}, \varphi_{\mathrm{RA}}, \varphi_{\mathrm{ND}}, \varphi_{\mathrm{MDF}}, \varphi_{\mathrm{MND}} \rangle$, where $\varphi_{\mathrm{DF}} = (d, u_{gd})$, $\varphi_{\mathrm{RA}} = (x_{bd}, y_{sd})$, $\varphi_{\mathrm{ND}} = (v_1, v_2, \ldots, v_6)$, $\varphi_{\mathrm{MDF}} = (p_m, q_{lm})$, and $\varphi_{\mathrm{MND}} = (v_1, v_3, \ldots, v_6)$ (see Appendix A in the Online Supplement for decision-variable definitions). The subproblem structures and guidance mechanisms are presented in the next two sections.

3. Extrinsic Guidance Mechanisms

The extrinsic guidance mechanisms determine the variables that are exogenous to the simulation environment. We develop the models underlying these mechanisms in the following discussion.

3.1. Domain Formation: Subproblem DF

Users are classified into *groups* based on job functions and other requirements, and we form *domains* with similar requirements and geographical coordinates. The inputs are (i) number of users per group in each location, and (ii) communication load between locations, defined as the average data transfer rate between each pair of locations. A solution to subproblem DF provides a domain configuration φ_{DF}, holding other variables and parameters of DSDP constant.

Initially, we assume that a copy of each component and data resource accessed by users in a domain is available in that domain. This assumption is dropped after resource allocations are available. Initial domains are created using a *k*-means nonhierarchical clustering mechanism (Anderberg 1973). However, it is difficult to estimate accurately domain-formation effectiveness solely with cluster analysis. Hence, we develop three measures of domain effectiveness to capture (i) average response time (ART) for different components, (ii) average processor utilization (APU), and (iii) cost of resource allocation and maintenance (CRAM), which are described in detail in Appendix A in the Online Supplement.

Hence the objective of subproblem DF is given by

$$\operatorname{Min} Z_{\mathrm{DF}} = W_1 \times \mathrm{ART} - W_2 \times \mathrm{APU} + W_3 \times \mathrm{CRAM}, \quad (1)$$

where the W_i are user-defined positive weights with $\sum_i W_i = 1$. Minimizing Z_{DF} ensures that multiple users in the same group, or locations with significant traffic among one another, would be members of the same domain, leading to better resource allocation and decreased inter-domain traffic.

We perturb the initial domain composition using the *variable-domain perturbation algorithm* given in Appendix B in the Online Supplement. This may alter the number and composition of domains to improve Z_{DF}.

3.2. Resource Allocation: Subproblem RA

Subproblem RA allocates data sets and component resources to domains identified earlier, with the conflicting objectives of maintaining high performance and utilization, coupled with low operating and remote communications costs. Subproblem RA uses the output from DF and provides a resource allocation configuration φ_{RA}, holding other variables and parameters of DSDP constant. Since it is difficult to estimate accurately the effects of a resource allocation configuration on $\mathrm{CS}_{\mathrm{DSDP}}$, we develop five metrics to mirror closely the effects: (i) domain load balancing (DLB), (ii) maximum domain criticality (MDC), (iii) computing load balancing (CLB), (iv) total cost (TC), and (v) remote communication network load (NL). DLB, MDC, and CLB are minimized under distributed systems, while TC and NL are minimized under centralized systems. We describe these in detail in Appendix A in the Online Supplement.

Subproblem RA is modeled as follows:

$$\begin{aligned} \operatorname{Min} \ Z_{\mathrm{RA}} = (&\mathrm{DLB} \times W_1 + \mathrm{MDC} \times W_2 + \mathrm{CLB} \times W_3 \\ &+ \mathrm{TC} \times W_4 + \mathrm{NL} \times W_5) \quad (2) \end{aligned}$$

$$\text{s.t.} \ \sum_{d \in D} x_{bd} \geq 1 \quad b \in B \quad (3)$$

$$\sum_{d \in D} y_{sd} \geq 1 \quad s \in S, \text{ set of data sources/databases} \quad (4)$$

$$x_{bd}, y_{sd} \in 0, 1 \quad \forall b, s, d. \quad (5)$$

The W_i are user-defined positive weights with $\sum_{i=1}^{5} W_i = 1$. (3) ensures that at least one copy of each component is available on the network, and (4) ensures that at least one copy of each data set is available on the network.

The overall objective function (2) is nonconvex in general as shown in Appendix A in the Online Supplement. We use a tabu-search (Glover and Laguna 1997) based resource-allocation strategy introduced in Bhattacharjee et al. (2001) to explore the solution space efficiently and determine a set of effective allocation schemes. Tabu search has been extensively used to solve practical optimization problems in many fields (Laguna 1994, Skorin-Kapov and Skorin-Kapov 1994, Semet and Taillard 1993), and has been reported to be more efficient than other random-search-based techniques such as simulated annealing (April et al. 2003).

3.3. Domain Formation: Subproblem MDF

Designing mobile systems involves using information on locations in which mobile users operate and components and data reside, the nature of the fixed network, coupled with mobile users' profiles (moving patterns, log-on frequencies, etc.), and MCSS processing capacity. Subproblem MDF uses inputs from DF, RA, and ND and provides a configuration φ_{MDF}, holding other variables and parameters of DSDP constant. This configuration is subsequently used in MND for mobile network design.

The average utilization of an MCSS in location m is $util_{\mathrm{avg}} = \mathrm{TML}_m/(R_m \times 3{,}600{,}000)$, where the total average message load for the MCSS (TML_m) is defined in Appendix A in the Online Supplement. Therefore, the cost-minimization formulation for subproblem MDF is

$$\operatorname{Min} \ Z_{\mathrm{MDF}} = \tfrac{1}{2} \sum_{l \in M} \sum_{m \in M} q_{lm} \times C_{lm} + \sum_{m \in M} (p_m \times F_m) \quad (6)$$

$$\text{s.t.} \ \sum_{m \in M} q_{lm} = 1, \quad \forall\, l \in M \quad (7)$$

$$q_{lm} \leq p_m, \quad \forall\, l, \forall\, m \quad (8)$$

$$LL \times p_m \leq util_{\mathrm{avg}} \leq UL \times p_m, \quad \forall\, m. \quad (9)$$

Equation (7) ensures that any non-MCSS site l must be associated with one and only one MCSS site m, (8) ensures that MCSS site m must have an MCSS assigned to it, and (9) specifies that the utilization

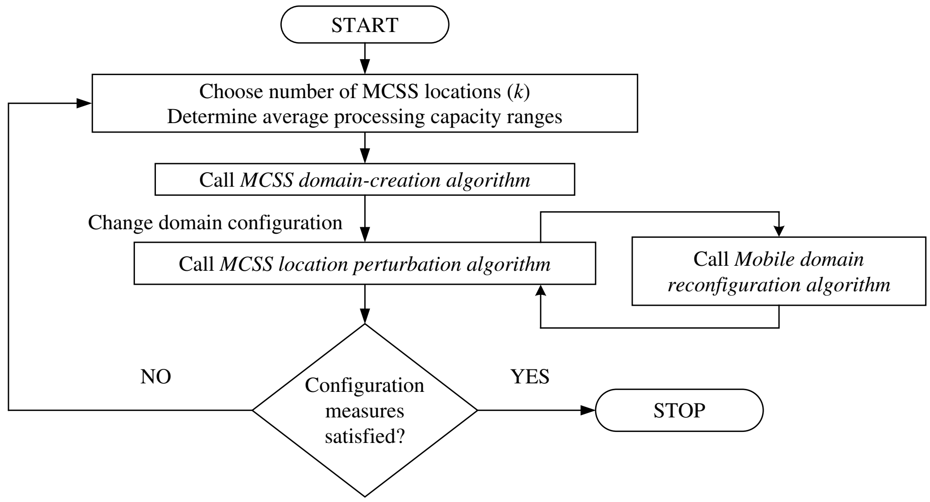

Figure 3 **MCSS Domain Creation Steps**

level of each MCSS ($util_{\text{avg}}$) must be within the user-defined range of LL (lower bound) and UL (upper bound), expressed as a fraction. This ensures a load-balanced network, which reduces congestion and improves performance.

The problem formulation, with (7), (8), and a known number of MCSS's, is similar to the hub-location problem, which itself is NP-hard (O'Kelly 1987). We develop a three-step algorithm to solve MDF, beginning by specifying a fixed number of MCSS locations, determining average MCSS processing capacity required, and creating an initial domain configuration. This configuration is then perturbed to evaluate further improvement in Z_{MDF}.

Given a fixed number k of MCSS locations, the range of average MCSS processing capacity is determined using (9). The total hourly traffic on the mobile network is the sum of traffic from mobile clients and servers, and is expressed as $T = N \times H \times (S^c + S^s)$. Rearranging (9), we have

$$\frac{T}{UL \times k \times 3{,}600{,}000} \leq p_{\text{avg}} \leq \frac{T}{LL \times k \times 3{,}600{,}000}$$

where p_{avg} is the average MCSS processing capacity. This provides an estimate of acceptable cost ranges for MCSS domain formation. For a fixed k, a lower bound of (6) is

$$LB_k = \sum_{l \in \{M^*\}} C_{ll} + \sum_{l \in \{M^*\},\, m \in \{M \backslash M^*\}} \min(C_{lm}) + 200k \times \frac{T}{UL \times k \times 3{,}600{,}000}, \tag{10}$$

where M^* is the current set of MCSS locations, and the set of unselected potential locations is $M \backslash M^*$. The first term is the connection cost within k MCSS locations, the second is the minimum connection cost of connecting $M \backslash M^*$ remote locations, and the last is the minimum fixed cost. The main purpose of the lower bound is to estimate the goodness of the current solution, and may sometimes represent an infeasible configuration.

Figure 3 presents the three-step algorithm to identify the number of MCSS locations and logical domains that connect each MCSS location to several non-MCSS locations at the lowest cost. The *MCSS domain-creation algorithm* uses a greedy approach to construct an initial set of domains with balanced load across domains. After the initial mobile domains have been created, we perturb the domain configuration to determine possible improvements, using (i) the *MCSS location perturbation algorithm*, and (ii) the *mobile domain reconfiguration algorithm*. The *MCSS location perturbation algorithm* is composed of exchanging one pair of locations between M^* and $M \backslash M^*$. Within this, the *mobile domain reconfiguration algorithm* redistributes current non-MCSS locations across domains. The procedures are embedded in a broad tabu-search meta-heuristic, which steers the search procedure to a solution space beyond local optimality. The tabu-search meta-heuristic, along with the *MCSS* domain-formation algorithms, is in Appendix B in the Online Supplement.

4. Intrinsic Guidance Mechanisms

Design configurations from the analytic solutions present effective environments that are candidates for exploration through simulation, where stochastic parameters of subproblems ND and MND are studied. However, simulations are expensive. Hence, intrinsic guidance involves learning from a set of simulation experiments and using this knowledge to guide further simulations. A learning model is embedded within the analysis to guide the simulation strategy successively and increase its efficiency.

A simulation model is run for a given experimental design, and the output data are used to estimate a general multiple linear model, with each DSDP objective measure as a dependent variable, and input factors as independent variables. We then perturb the parameters of the linear model and analyze the sensitivity of the output measures. Some promising sets of parameters are identified and subsequently simulated for detailed investigation. The resultant output data are used to refine the regression model. This constitutes the intrinsic guidance mechanism for the overall design configuration of DSDP.

4.1. Configuration Selection and Analysis: Subproblem ND

The results of solving subproblem ND consist of a set of nondominated solutions among the set of configurations considered within the target range specifications. The criteria set $Z_{\mathrm{ND}} = \{z_1, z_2, z_3, -z_4, -z_5, -z_6\}$ is used in this determination. We denote a typical solution in this set as configuration φ_{ND} and it presents a fixed network infrastructure to link domains and distributed resources. The solution set is arrived at through (a) a set of pilot simulations to enable an initial estimation of the effects of the design variables on the performance criteria; (b) a set of simulations using efficient experimental designs for systematic parametric variations (Law and Kelton 2000); (c) a regression learning model to estimate progressively and refine the effects; and finally, (d) a standard procedure for determining nondominated solutions from the set considered (Chankong and Haimes 1983, Hans 1988). The configurations φ_{DF} and φ_{RA} are held fixed throughout the solution of ND. Sets of simulations and regressions are used iteratively to refine the estimates of the effects progressively and guide subsequent simulations. The *iterative simulation modeling and analysis algorithm* in Appendix B in the Online Supplement presents this strategy.

4.2. Configuration Selection and Analysis: Subproblem MND

Design configurations from other subproblems (φ_{DF}, φ_{RA}, φ_{MDF}, and φ_{ND}) form the environment of subproblem MND, which provides configuration φ_{MND} for a mobile system. The solution strategy is similar to that of subproblem ND and a set of nondominated solutions is determined as before for the criteria set $Z_{\mathrm{MND}} = \{z_1, z_2, z_3, -z_4, -z_5, -z_6\}$. Two important parameters in the MCSS network design are (i) profile management during transactions and (ii) security integration. The *MCSS knowledge level* determines profile management, where an MCSS with full knowledge stores profiles of *all* mobile users (MUs) registered in the system; hence it can authenticate any MU currently within its domain, without contacting any other MCSS. However, when an MCSS has knowledge only about the MU *originally* registered in its own domain, it needs to contact a remote MCSS when it receives a request from an MU registered in other domains. Separately, security-based protocols describe how security requirements are implemented when an MU sends a request and affects system performance, and is detailed in Appendix A in the Online Supplement. These assume critical importance, since wireless systems offer greater chance of signal interception and impersonation.

Table 1 **Variables Used in Subproblem MND Simulation Model**

Type	Parameters	Description
Design	MCSS knowledge level	Degree of MCSS knowledge about mobile user
	Security level	Routing protocol to ensure different levels of security
	Connection bandwidth level	Bandwidth connecting MCSS with other system servers.
Fixed (given)	Mobile user profile (planning estimate /pilot study)	• Number of mobile users • Mobility rate • Log-on frequency • Size of data transferred (Kb)
	MCSS domains	• Number of MCSS's • MCSS domain configuration • MCSS processing capacity • MCSS utilization level
	Fixed network configuration	• Server capacities • Network capacities

Table 1 summarizes the variables studied. We consider two extreme profile scenarios: (i) full knowledge of all MUs, and (ii) knowledge only about locally registered MUs. Subproblem MND uses inputs from all previous subproblems and is analyzed through the *iterative simulation modeling and analysis algorithm* (Appendix B in the Online Supplement).

5. The Guided Simulation Methodology

Figure 4 presents the iterative guided simulation methodology. Given the differing objective measures in $\mathrm{CS}_{\mathrm{DSDP}}$, improvement of individual objective metrics is measured as a percent change in their values. An improvement in $\mathrm{CS}_{\mathrm{DSDP}}$ occurs when (i) a dominant solution is found after an iteration of the methodology, or the average percent improvement in the set of objective measures is positive, (ii) the measures are within their respective target ranges, and (iii) the design configuration is feasible.

During problem decomposition, tractable subproblems are formulated by relaxing some constraints of the overall problem, or fixing values of certain decision variables not specific to the subproblem. Each subproblem has specific objectives that are either the same as $\mathrm{CS}_{\mathrm{DSDP}}$, or surrogates of $\mathrm{CS}_{\mathrm{DSDP}}$ (Table 2). Each subproblem is solved separately, and a key

```
Objective: Solve for nondominated solution of CS_DSDP   // defined in Section 2.1
Inputs: Subproblems DF, RA, ND, MDF, and MND (with associated parameters and decision variables);
        MAXIMUM_ITERATIONS.
i = 1;  // used to identify initial domain configuration
max_iter = 1;
Solve DF (initial solution);  // clustering and perturbation algorithm—Section 3.1
/* solve for nondominated CS_DSDP through interlinked solution approach on decomposed problems */
WHILE max_iter ≤ MAXIMUM_ITERATIONS   // overall algorithm stopping criterion
/* guided simulation, extrinsic guidance from analytically derived DF and RA configurations, intrinsic guidance
   from empirically derived ND configuration. */
  WHILE (improvement in CS_DSDP)   // nondominated solution
    IF i > 1
      Perturb and re-solve DF;   // extrinsic guidance—Section 3.1
    ENDIF;
    Solve RA;   // extrinsic guidance, nondominated solution—Section 3.2
    WHILE (improvement in CS_DSDP AND max_iter ≤ MAXIMUM_ITERATIONS)
      Solve ND;   // intrinsic guidance, iterative solution based on simulations and regression analysis—Section 4.1
      max_iter = max_iter + 1;
    ENDWHILE
    i = i + 1;
  ENDWHILE;
/* guided simulation, extrinsic guidance from DF, RA, ND, and MDF configurations, intrinsic guidance from
   empirically derived MND configuration. */
  Solve MDF;   // extrinsic guidance—Section 3.3
  WHILE (improvement in CS_DSDP AND max_iter ≤ MAXIMUM_ITERATIONS)
    Solve subproblem MND;   // intrinsic guidance, iterative solution based on simulations and regression
                                    analysis—Section 4.2
    max_iter = max_iter + 1;
  ENDWHILE;
  IF (no improvement in CS_DSDP OR max_iter > MAXIMUM_ITERATIONS)
    STOP;
  ENDIF;
  max_iter = max_iter + 1;
ENDWHILE;
STOP;
Outputs: Design configuration Φ = ⟨φ_DF, φ_RA, φ_ND, φ_MDF, φ_MND⟩
```

Figure 4 **Overall Guided Simulation Methodology**

step in the guided simulation methodology involves linking their solutions. At each linking point, the subproblem solutions are used to guide the adjustments in design parameters necessary to solve the subsequent subproblems in the feedback framework. This ensures that the effects of all decision variables and parameters are estimated periodically on the overall objective set.

Subproblem RA requires logical network domains to allocate resources optimally, so an initial domain configuration through subproblem DF acts as an *extrinsic guidance* for the subsequent resource allocation. Next, design configurations from DF and RA provide *extrinsic guidance* for subproblem ND, which interconnects logical domains and allocated resources. The *intrinsic guidance* uses iterative mechanisms—with inner loops of simulation analysis, followed by a regression model—to derive design configurations empirically, until no further improvement in the solution is detected. Given a set of domains, resource allocations, and fixed network infrastructure, the mobile network is designed similarly using both forms of guidance.

The method allows for further improvement of the objective function through perturbation of the solutions to DF and RA while solving ND. When the required level of simulations in solving ND is completed and if it is possible to improve the solution further, the method would backtrack to subproblems DF and RA to improve the extrinsic guidance in solving ND. This implies further exploration of the solution space and could occur in practical systems design. A similar backtracking loop is used to solve subproblem MND. Since MND basically addresses mobile user needs and the required resource allocations, it will mostly be necessary to backtrack to subproblem MDF only. However, if a designer wishes to reconsider the entire domain structure and resource allocations at this stage, then backtracking to subproblems DF and RA can easily be incorporated in the solution framework. When the overall design algorithm terminates, the designer is provided complete system configurations of a set of nondominated solutions from the total set considered with different tradeoffs among the design objectives. This modeling strategy provides a flexible, efficient and systematic approach to the

Table 2 DSDP Decomposition Structure and Solution Strategy

Subproblem (in order of solution)	Surrogate criteria	DSDP criteria represented	Subproblem objective structure	Solution methodology	Guidance mechanism
Domain formation (DF)	ART APU CRAM	z_1, z_2, z_3 z_4, z_5 z_7	Weighted linear combination	Clustering and perturbation algorithm	*Extrinsic*
Resource allocation (RA)	DLB MDC CLB TC NL	z_1, z_2, z_4 z_1, z_2, z_5 z_1, z_2, z_4 z_7 z_7	Weighted linear combination	Tabu-search based algorithm	*Extrinsic*
Network design (ND)	$z_1, z_2, z_3, -z_4, -z_5, -z_6$	Same as the surrogates	Multicriteria	Simulations, regression learning, nondominated solutions in a set	*Intrinsic*
Mobile domain formation (MDF)	Mobile domain costs	z_7	Single criterion	Tabu-search based algorithm	*Extrinsic*
Mobile network design (MND)	$z_1, z_2, z_3, -z_4, -z_5, -z_6$	Same as the surrogates	Multicriteria	Simulations, regression learning, nondominated solutions in a set	*Intrinsic*

exploration of the design space in practical problems, including the design of new systems, upgrades, and system reconfigurations.

6. System Implementation

We implemented our method in a large manufacturing and service-oriented organization in the southern U.S. Manufacturing activity across two fixed locations needed coordination. Sales, customer service, and training with support across remote locations required integration with the core systems. An intranet design was required to link over 600 fixed-location users (composed of five different *groups*) spread over two locations. Additionally, it needed to connect 30 remote locations with mobile users. The design required flexibility to change remote locations periodically to accommodate current clients that the organization supported. Remote clients could authenticate themselves and log in using various methods (PSTN, internet, wireless access points on intranet, etc.).

The core system consisted of 80 subcomponents consolidated into eight component systems for allocation, of which three components were multimedia systems to train and certify support personnel. In addition, four main database systems were used to drive these applications. Query and update operations over fixed and wireless networks were modeled as uniform distributions. *Read operations* were distributed between 5 to 50 per hour, with sizes ranging from 0.1 to 2,000 MB. *Write operations* ranged between 5 to 20 per hour, with sizes between 0.01 to 5 MB. Mobile users' *client messages* were estimated to be between 100 and 500 Kb, while *server messages* were between 100 and 1,000 Kb, with a frequency between 10 and 50 per hour. Component access rates to each other and to databases were also estimated. Targeted average and maximum *read latencies* were 30 s and 120 s respectively (nonmultimedia), while those for *write latencies* were 45 s and 150 s. Multimedia applications displayed 640×480 8-bit video images, with a targeted QoS level of 26 frames/s on average (+/−2 frames/s). This allows for a simple, jitter-free image. Control of other important factors such as standard deviation and maximum frame latencies for each multimedia application were considered, but were not used in the objective. Due to cost and licensing issues, multimedia applications could support up to 20 simultaneous users. The number of users, types of components, and operations conducted over this intranet is fairly typical for small and medium-sized organizations that need to interconnect multiple locations and processes with both wired and wireless systems.

6.1. Performance Results

The solution process follows the overall guided simulation methodology (Figure 4). This intranet was a new system being developed, hence the k-means clustering method uses current information to form initial domains. Z_{DF} is calculated for various values of k, and a four-domain solution is initially chosen with $Z_{DF} = 0.23$ (ART = 0.58, APU = 0.37, CRAM = 0.48). Further configuration improvement is achieved through the *variable-domain perturbation algorithm*, which yielded a five-domain solution with $Z_{DF} = 0.18$ (ART = 0.39, APU = 0.35, CRAM = 0.51). One location contains three domains, while the other location contains the other two, which is chosen as configuration φ_{DF}.

Next, the framework uses φ_{DF} to solve subproblem RA. It first creates a resource-allocation solution pool consisting of a set of nondominated solutions based

Table 3 Fixed Domains and Allocated Resources (φ_{RA})

	Number of users per group					Component system	Database
Domain	1	2	3	4	5	allocation	allocation
1	143	6				5, 6	1, 2
2			130			2, 3 (MM)	
3		82		68		1 (MM), 7	3
4					51	4	4
5	18		38		80	8 (MM)	

Note. (MM)–multimedia application.

on the five metrics of RA, by choosing a different starting point of the resource-allocation algorithm each time. The system-designer-assigned relative weights were $W_1 = 0.25$, $W_2 = 0.10$, $W_3 = 0.25$, $W_4 = 0.25$, and $W_5 = 0.15$. The configuration φ_{RA} with the minimum Z_{RA} (=0.2545) is chosen (where DLB = 0.18, MDC = 0.58, CLB = 0.23, TC = 0.22, and NL = 0.26). Allocation details of this configuration are illustrated in Table 3, where users, components, and databases are distributed across the five domains identified in φ_{DF}. This provides an effective design environment and illustrates the extrinsic guidance for analyzing subproblem ND.

In subproblem ND, each component and database is modeled as hosted on a dedicated server in a multi-tier architecture. This helps us study the effect of each design parameter and decision variable on each resource metric independently. The network-simulation model is implemented in COMNET III™ (CACI Products Company 1998), a discrete-event network design and performance analysis tool. Pilot simulations indicated six factors for detailed investigations: (i) database-server processing speed (X_1), (ii) read-query frequency (X_2), (iii) write-query frequency (X_3), (iv) TP model (X_4), (v) Input/output (I/O) rates (for multimedia systems) (X_5), and (vi) network bandwidth (X_6). Given the low number of factors in this case, a full factorial model with $2^6 = 64$ design points are chosen to study the direct, two-way, and higher-order interaction of these parameters on Z_{ND}. With 30 replications for each design, for a total of 1,920 simulation runs, our parameter estimates have a precision of 0.25 and a confidence level of 90% (Law and Kelton 2000) (see details in *iterative simulation modeling and analysis algorithm* in Appendix B in the Online Supplement). Each replication is initialized and run for 86,400 s of simulation time, allowing sufficient time for warm up to reach steady state.

The output data were normalized and used to estimate general linear models for z_1 through z_6. Interestingly, we find some significant direct and two-way interaction effects, but higher-order interactions are not significant. The linear models are used to provide intrinsic guidance to further simulations

Table 4 Subproblem ND: Intrinsically Guided Design Results

	z_1	z_2	z_3	z_4	z_5	z_6
Initial values:	56 s	196 s	18 frames/s	38%	24%	16%
Iteration	(Percentage improvement in objective value)					
1	21.43	19.39	22.22	−2.63	−16.67	−12.50
2	34.09	32.91	27.27	7.69	−3.57	−27.78
3	−13.79	2.83	14.29	−2.78	13.79	−13.04
4	36.36	45.63	−3.13	5.41	6.06	7.69
5	23.81	14.29	−3.23	−2.86	6.45	8.33
Final values:	16 s	48 s	30 frames/s	36%	29%	22%

efficiently. Table 4 shows the progression of the algorithm through various iterations for one of the solutions analyzed through simulations. The first row of values are from the initial solution, and the next set of rows depict changes in the objective metrics after each iteration. The last row shows Z_{ND} beyond which there is no further improvement, which ends subproblem ND with configuration φ_{ND}.

Next, we iterate between subproblems DF, RA, and ND (see Figure 4), without improvement in CS_{DSDP}. Hence we continue the design process with subproblem MDF. Mobile-user estimates and considerations of future growth suggested modeling 100 to 300 users per location, distributed uniformly. Two significant design issues identified earlier in Section 3.3 are as follows: (i) MCSS utilization ($util_{\text{avg}}$), and (ii) MCSS processing capacity (p_{avg}). The "allowable range" of $util_{\text{avg}}$ was chosen at two levels: between 20%–80% (broad range) and 40%–60% (narrow range). The "change policy" of p_{avg} also had two levels: processing capacities of *all* MCSSs are modified during an iteration if one violates the "allowable range" and only the violator's capacity is modified. The above set of experiments resulted in 20 nondominated solutions with different tradeoffs among them.

The subproblem MDF is solved with k ranging from 1 through 10 domains (for the 30-location problem), and Table 5 lists Z_{MDF} and corresponding deviation from the lower bound (10) for each design point. The results suggest that the tabu-search-based algorithm is efficient and robust across scenarios, deviating between 3% and 9% off the lower bound. Note that the lower bound may represent an infeasible solution, whereas the algorithm always returns a feasible solution, and the small deviation from the lower bound underscores its robustness. The best solution for each scenario is shown in bold in the table. The overall best configuration (1,816.60) uses a five-domain architecture, under a broad allowable utilization level and a policy of modifying only the individual violator's capacity. (This five-domain mobile architecture differs from the DF architecture discussed earlier.) This is within 4% of the lower bound, and is chosen as φ_{MDF}.

Table 5 Subproblem MDF: Mobile-Domain Formation Results

	Processing capacity change policy					
	All capacities modified			Individual violator's capacity modified		
Allowable utilization range	k	Z_{MDF}	% deviation from lower bound (%)	k	Z_{MDF}	% deviation from lower bound (%)
Broad (20%–80%)	1	2,320.22	6.98	1	2,512.68	7.74
	2	2,092.17	5.25	2	2,094.06	5.38
	3	1,996.19	4.88	3	1,923.12	3.61
	4	1,933.04	3.82	4	2,143.77	5.06
	5	1,924.05	3.63	**5**	**1,816.60**	**3.39**
	6	2,087.34	5.95	6	1,869.18	3.07
	7	1,927.13	4.71	7	1,981.32	4.67
	8	**1,841.83**	**3.87**	8	2,032.52	4.36
	9	2,031.87	4.19	9	1,983.49	4.64
	10	2,000.48	4.27	10	1,871.26	3.79
Narrow (40%–60%)	1	2,640.98	8.92	1	2,726.52	9.04
	2	2,548.59	8.02	2	2,425.14	7.16
	3	2,511.79	8.04	3	2,411.35	7.80
	4	2,339.33	6.20	4	2,615.38	8.17
	5	2,447.33	7.60	5	2,330.85	6.15
	6	2,408.10	7.30	**6**	**2,237.84**	**6.91**
	7	2,451.18	7.98	7	2,464.25	6.18
	8	**2,258.32**	**6.37**	8	2,583.26	6.47
	9	2,461.37	8.72	9	2,387.59	6.84
	10	2,321.24	7.89	10	2,327.26	7.33

Lastly, subproblem MND is solved to obtain φ_{MND}. Fixing the set of configurations (φ_{DF}, φ_{RA}, φ_{MDF}, and φ_{ND}) obtained earlier, subproblem MND is solved using the guided simulation procedure. The MCSS knowledge level (X_7), security level (X_8), and connection bandwidth between locations (X_9) are identified as design variables. These are modeled and analyzed at two levels, e.g., X_8 uses scenarios of a single-session authentication protocol (low security) and a per-request authentication protocol (high), while X_9 is modeled as a frame relay link of 128 kbps (low speed) and a point-to-point 1.5 Mbps T1 (high). A full-factorial model with 2^3 design points are chosen to study direct, two-way, and higher-order interactions on Z_{MND}. With 30 replications for each design, for a total of 240 simulation runs, our parameter estimates have a precision of 0.15 and a confidence level of 90% (see details in *iterative simulation modeling and analysis algorithm* in Appendix B in the Online Supplement). Each replication is initialized and run for 86,400 s of simulation time, allowing sufficient time for warm up to reach steady state.

The output data were normalized and used to estimate the initial linear models for z_1 through z_6. Results show that *read latency*, *write latency*, and *network utilization* are significantly affected by the mobile network parameters, while other metrics are not significantly affected. This is expected, as multimedia operations are used on this organization's fixed network only, based on bandwidth constraints. Another interesting finding is that several two-way interactions have a statistically significant effect on the system performance metrics, while further higher-order interactions are not significant. Table 6 presents the progression of Z_{MND} for the mobile network configuration problem using the fixed network design solution described above. The progression shows no improvement after four iterations. Hence this output is chosen as φ_{MND}.

Iterations through the guided simulation framework were continued, but no significant improvement was obtained in CS_{DSDP}. Hence the set of configurations $\Phi = \langle \varphi_{DF}, \varphi_{RA}, \varphi_{ND}, \varphi_{MDF}, \varphi_{MND} \rangle$ determined as above forms a complete Intranet solution for the above fixed network configuration. The overall experiment yielded seven nondominated mobile network solutions for this fixed network configuration.

Table 6 Subproblem MND: Intrinsically Guided Design Results

	z_1	z_2	z_3	z_4	z_5	z_6
Initial values:	28 s	68 s	31 frames/s	37%	31%	32%
Iteration	(Percentage improvement in objective value)					
1	7.14	17.65	6.45	2.70	9.68	9.38
2	23.08	14.29	−3.45	−2.78	−7.14	3.45
3	−5.00	−6.25	3.33	0.00	10.00	7.14
4	9.52	1.96	0.00	2.70	12.12	0.00
Final values:	19 s	50 s	29 frames/s	36%	29%	26%

Table 7 **Fixed Network Design Grid**

Objective metrics	Independent factors													
	X_1	X_2	X_3	X_4	X_5	X_6	$X_1 \times X_2$	$X_1 \times X_3$	$X_1 \times X_4$	$X_2 \times X_3$	$X_2 \times X_5$	$X_2 \times X_6$	$X_3 \times X_4$	$X_4 \times X_6$
z_1	−	+	+	+		−		+	+			+	+	
z_2	−		+	+					−		−		+	
z_3		+			−	−								
z_4	−		+	+									+	
z_5	−						−	−		−				
z_6			+	+		−							+	+

Note. + signifies z increases when X increases; − signifies z decreases when X increases; blank signifies no statistically significant effect.

6.2. Discussion

Insights from the structural relationships are shown in the design grids of Tables 7 and 8. These tables present the significant main and interaction effects observed in the linear model that was estimated using the data generated from the simulation experiments. The interaction effects not shown here were not statistically significant at a 95% level of significance. The database-server processing speed critically affects a majority of QoS performance metrics, including component system utilization (Table 7). Interestingly, database-server processing speed (X_1) affects z_1 negatively, and write frequency (X_3) affects it positively; however their interaction affects z_1 positively. A similar effect is observed between X_1 and X_4. We also observe that data insert/update frequency (write frequency) of users is more critical than is the simple query (read) frequency. Although both determine latency from the database server, write frequency and database-server speed jointly affect system delays. Read frequency assumes more importance in a multimedia system, in conjunction with I/O rates. The interaction between write frequency and the chosen TP model is also a determining factor for system configuration and performance. This suggests that the TP model should be carefully chosen to balance the need for up-to-date data with QoS latency. Network bandwidth affects multimedia latency significantly, and also has an interaction effect with the read frequency and TP model chosen. Cost minimization through analytical models suggests that fixed network infrastructure should be centralized (without violating other constraints).

Similarly, for mobile systems design (Table 8), a high MCSS knowledge level lowers inter-domain traffic and affects the QoS measures, while a high security level increases traffic. However, their interaction affects the measures negatively. Hence MCSS models should use full knowledge of all mobile users, and the level of security should be set through experimentation to avoid any significant adverse effects. Connection bandwidth has a significant effect on system performance, independently and in association with the security level. To minimize costs, models should allow for broad utilization ranges for MCSS design and modify the MCSS processing capacity only for those that are outside the range.

Table 8 **Mobile System Design Grid**

Objective metrics	Independent factors				
	X_7	X_8	X_9	$X_7 \times X_8$	$X_8 \times X_9$
z_1	−	+	−	−	
z_2	−	+		−	
z_3			−		
z_4					
z_5	−				+
z_6	−	+	−		+

Note. + signifies z increases when X increases; − signifies z decreases when X increases; blank signifies no effect.

7. Conclusion

We have developed a *guided simulation* methodology to design large-scale distributed systems. Typically, these have a large set of stochastic design parameters with multiple conflicting yet closely coupled design objectives. The design space for such systems is also combinatorially large. We developed an integrated solution framework by introducing two essential concepts in large-scale systems design: a *problem-decomposition* strategy, and a *guided-simulation* methodology for problem solution. The guided-simulation methodology subsumes two essential concepts: *extrinsic* and *intrinsic* guidance in conducting large-scale simulations. The guidance mechanisms are derived from the solutions of decomposed subproblems (extrinsic), and results from simulation experiments that are internal to the simulation methodology (intrinsic). In this process, the focus of performance investigations is first restricted from a combinatorially large design space to promising design configurations. These are further analyzed using simulation. Since large-scale simulations are usually expensive, this strategy leads to an efficient way of arriving at effective design configurations that meet the designer's requirements on the performance criteria. The framework is integrated, tractable, and efficient, and is illustrated through an intranet design problem.

We derive several insights on large-scale system design from the implementation. These can be used for system benchmarking, sensitivity analysis on current systems, and upgrades to existing architectures. These insights essentially focus on the important attributes and their interactions in component-based systems. Some future research issues are design of web-services architectures, grid-computing configurations, and several other emerging technologies where an integrated approach to resource sharing is essential.

Acknowledgments

The authors thank the editor, area editor, and anonymous referees for many constructive suggestions. The first author also acknowledges support from the XEROX CITI Endowment Fund, the Connecticut Information Technology Institute (CITI), and the Gladstein-Endowed Management Information Systems (MIS) Research Laboratory.

References

Anderberg, M. R. 1973. *Cluster Analysis for Applications*. Academic Press, New York.

Apers, P. M. G. 1988. Data allocation in distributed database systems. *ACM Trans. Database Systems* **13** 263–304.

April, J., F. Glover, J. P. Kelly, M. Laguna. 2003. Practical introduction to simulation optimization. *Proc. Winter Simulation Conf., New Orleans, LA*, 71–78.

Bhattacharjee, S., R. Ramesh, S. Zionts. 2001. A design framework for e-business infrastructure integration and resource management. *IEEE Trans. Systems, Man and Cybernetics* **31** 304–319.

CACI Products Company. 1998. *COMNET III Reference Guide*, Release 2.0. CACI Products Company, La Jolla, CA.

Chankong, V., Y. Y. Haimes. 1983. *Multiobjective Decision Making Theory and Methodology*. North-Holland, New York.

Chari, K. 1996. Resource allocation and capacity assignment in distributed systems. *Comput. Oper. Res.* **23** 1025–1041.

Chen, P., J. Akoka. 1980. Optimal design of distributed information systems. *IEEE Trans. Comput.* **29** 1068–1080.

Chu, W. W. 1969. Optimal file allocation in multiple computer systems. *IEEE Trans. Comput.* **18** 885–889.

Dhawan, C. 1997. *Mobile Computing: A Systems Integrator's Handbook*. McGraw-Hill, New York.

Fan, M., J. Stallaert, A. B. Whinston. 2000. The adoption and design methodologies of component-based enterprise systems. *Eur. J. Inform. Systems* **9** 25–35.

Fu, M. C. 1994. Optimization via simulation: A review. *Ann. Oper. Res.* **53** 199–248.

Fu, M. C. 2002. Optimization for simulation: Theory and practice. *INFORMS J. Comput.* **14** 192–215.

Fu, M. C., S. Andradottir, J. S. Carson, F. Glover, C. R. Harrell, Y. Ho, J. P. Kelly, S. M. Robinson. 2000. Integrating optimization and simulation: Research and practice. *Proc. Winter Simulation Conf., Orlando, FL*, 610–616.

Ghosh, D., I. Murthy, A. Moffett. 1992. File allocation problem: Comparison of models with worst case and average communication delays. *Oper. Res.* **40** 1074–1085.

Glover, F., J. P. Kelly, M. Laguna. 1999. New advances for wedding optimization and simulation. *Proc. Winter Simulation Conf., Phoenix, AZ*, 255–260.

Glover, F., M. Laguna. 1997. *Tabu Search*. Kluwer, Boston, MA.

Gray, J. 1993. *The Benchmark Handbook: For Database and Transaction Processing Systems*. M. Kaufmann Publishers, San Mateo, CA.

Hans, A. E. 1988. Multicriteria optimization for highly accurate systems. W. Stadler, ed. *Multicriteria Optimization in Engineering and Sciences*, Vol. 19. Plenum Press, New York, 309–352.

Herzum, P., O. Sims. 2000. *Business Component Factory*. Wiley, Hoboken, NJ.

Hevner, A., A. Rao. 1988. Distributed data allocation strategies. *Adv. Comput.* **27** 121–155.

Ho, Y. C., C. G. Cassandras, C. H. Chen, L. Y. Dai. 2000. Ordinal optimization and simulation. *J. Oper. Res. Soc.* **51** 490–500.

Ho, Y. C., R. Sreenivas, P. Vakili. 1992. Ordinal optimization of DEDS. *Discrete Event Dynam. Systems: Theory and Appl.* **2** 61–88.

Jain, H. K. 1987. A comprehensive model for the design of distributed computer systems. *IEEE Trans. Software Engrg.* **C-31** 1092–1104.

Kanter, T. G. 2003. Going wireless, enabling an adaptive and extensible environment. *Mobile Networks and Appl.* **8** 37–50.

Katchabaw, M. J., H. L. Lutfiyya, M. A. Bauer. 2000. Driving resource management with application-level quality of service specifications. *Decision Support Systems* **28** 71–87.

Laguna, M. 1994. Clustering for the design of SONET rings in interoffice telecommunications. *Management Sci.* **40** 1533–1541.

Laguna, M., R. Martí. 2002. Neural network prediction in a system for optimizing simulations. *IIE Trans.* **34** 273–282.

Law, A. M., W. D. Kelton. 2000. *Simulation Modeling and Analysis*, 3rd ed. McGraw-Hill, New York.

Ma, H., M. El Zarki. 2002. A new transport protocol for broadcasting/multicasting MPEG-2 video over wireless ATM access networks. *Wireless Networks* **8** 371–380.

O'Kelly, M. E. 1987. A quadratic integer program for the location of interacting hub facilities. *Eur. J. Oper. Res.* **32** 393–404.

Purao, S., H. K. Jain, D. Narareth. 1998. Effective distribution of object-oriented applications. *Comm. ACM* **41** 100–108.

Purao, S., H. K. Jain, D. Nazareth. 1999. Supporting decision making in combinatorially explosive multicriteria situations. *Decision Support Systems* **26** 225–247.

Semet, F., E. Taillard. 1993. Solving real-life vehicle routing problems efficiently using tabu search. *Ann. Oper. Res.* **41** 469–488.

Shanthikumar, J. G., R. G. Sargent. 1983. A unifying view of hybrid simulation/analytic models and modeling. *Oper. Res.* **31** 1030–1052.

Skorin-Kapov, D., J. Skorin-Kapov. 1994. On tabu search for the location of interacting hub facilities. *Eur. J. Oper. Res.* **73** 502–509.

Talukdar, A. K., B. R. Badrinath, A. Acharya. 1999. Integrated services packet networks with mobile hosts: Architecture and performance. *Wireless Networks* **5** 111–124.

Umar, A. 1997. *Object-Oriented Client/Server Internet Environments*. Prentice Hall, Upper Saddle River, NJ.

Umar, A. 2003. *e-Business and Distributed Systems Handbook*. NGE Solutions, Middlesex, NJ, http://www.amjadumar.com.

Varshney, U., R. Vetter. 2002. Mobile commerce: Framework, applications and networking support. *Mobile Networks Appl.* **7** 185–198.

Vogel, A., B. Kerhevre, G. V. Bochmann, J. Gecsei. 1994. Distributed multimedia applications and quality of service: A survey. *Proc. 1994 Centre for Advanced Stud. Conf.* IBM (Canada) Centre for Advanced Studies, Toronto, ON, Canada, 71–87.

Wang, F., C. Wen, C. Cheng, M. Lee, T. Lin, S. Wang, Y. Oyang. 1996. Memory and disk bandwidth management in video-on-demand servers. *Proc. Sixth Internat. ACM Workshop on Network and Operating System Support for Digital Audio and Video, Zushi, Japan*, 69–72.

Wu, J. C. 2001. A real time transport scheme for wireless multimedia communications. *Mobile Networks Appl.* **6** 535–546.