Alternative Technologies

Enterprise Integrity: The Art of Mimicry III

Vol. 8, No. 4

Certain aspects of today's software practices reflect traditional, functional organization with its stove-pipe, top-down control structure and profit-center accounting. They are fundamentally misaligned with the principles and value of the business revolution that results in the service-centric business. As noted previously, IT has learned to mimic the vocabulary and appearance, but not the substance, of the service-centric business.

In my last column, I tested my readers' attention, straying from "service-centric" business terminology to IT's weaker "service-oriented" terminology. But the service-oriented business is an oxymoron, "leaning toward" services until it falls over. Service provision must be central to and drive management and operational practices. The service-centric business is goal-driven, supplying services so that customers achieve their goals, and aligning business goals with customer goals.

The service-centric business manager knows that satisfactory (let alone exemplary), competitive (let alone innovative) service delivery requires a well-managed and integrated suite of business processes. Without successful service delivery into the marketplace, nothing else matters. With it, subsidiary goals ranging from improved market share to greater profit margins can become the subject of rational optimization. When this precedence relationship among goals is turned on its head, as when profit-center profitability controls operations, it is possible to optimize the business out of existence.

The cost of a service is the result of the cumulative costs of the distributed set of activities that result in that service (including products), i.e., the cumulative costs of a business process allocated by service. Typically, these activities or "functions" are coupled non-linearly, and so cannot be optimized independently. Activities may also contribute to multiple business processes and therefore multiple services. As such, resources (including time) must be "owned by" the business process that consumes them (as contrasted with profit center, product-centric ownership by function or department) if the business process is to be optimized.

Compare service-centric business organization to its IT counterpart, SOA. SOA evolution has been convoluted, twisting it's way through client/server and object orientation to become a curious hybrid with distributed componentization at its core. SOA software practice is often

object-oriented, a perspective that gives primacy to the identification of objects and relationships among those objects. The closest thing to services is the object's methods which are, in combination, an abstract function. It follows that object design (in particular, establishing the class hierarchy) is merely a high-level, abstract form of functional decomposition. Determining which object class should implement a method binds that method inextricably to a set of abstract resources, and to a *functional* decomposition.

IT services cannot be object-structured methods without imposing rigidity. IT cannot expect practices like extreme or agile programming to maintain pace with a high velocity market. Refactoring does not suffice, tending to perpetuate a rigid class hierarchy even when a mismatch with service utility is obvious. Eventually, classes become so abstract they bear no resemblance to business entities and resources, and maintaining a relationship to business measures requires extreme analytical and development effort. The result? Rigid misalignment.

Functional organizations are necessarily rigid. They assume that functional relationships and priorities will change infrequently if at all, and yet history shows this assumption could not be more false. If a business is to be agile, responsive, and adaptable, flexible service provision must be treated as foremost and composable. Being service-centric makes the goal, rather than some presumed means to achieve that goal, paramount. This gives the business flexibility in delivery and resource management, and ultimately, the loop between provider and consumer is tightened. Services must be understood as implicitly defining products, not the other way round. When products, or their IT corollary "objects", are the focus, we ask foolish questions about the "best" object design in terms of properties rather than utility. Trying to satisfy a diverse market this way necessarily leads to mediocrity and inflexibility.

By contrast, supplying composable services with a guaranteed level of satisfaction (i.e., results) irrespective of the means used to meet the contract, promotes user responsibility for identifying what is wanted, vendor negotiation with the consumer, and cooperative competition among producers over innovative means to meet demand. The composition of business services should define class hierarchies dynamically and implicitly, reflecting the current, ever-changing, and interacting views of management, employees, suppliers, and consumers alike. Only then can service-oriented IT mimic service-centric business, reduce maintenance costs, and maintain agility. Agility is not achieved. A dynamic balancing act among unpredictable economic forces, it is the task of keeping producer and consumer alike on the tightrope of *enterprise integrity*.