From Environmental Structure to Service Systems Thinking: Wholeness with Centers Described with a Generative Pattern Language

DAVID ING, INTERNATIONAL SOCIETY FOR THE SYSTEMS SCIENCES AND AALTO UNIVERSITY

In the early 1990s, pattern languages originating in architecture began the foundation for development of work in object-oriented design and methods. Since that time, the work of Christopher Alexander has continued to develop, with new emphases only hinted upon in the research of the 1990s. The SSMED (Service Science, Management, Engineering and Design) vision originating circa 2003 is now being more thoroughly founded on systems thinking, leading to a new perspective labeled as Service Systems Thinking. Advances in generative pattern languages and collaborative Internet technologies are proposed to enhance the development of an emerging pattern language for service systems.

Categories and Subject Descriptors: **H.1.1 [Models and Principles]**: Systems and Information Theory—*General Systems Theory*; **H.5.3 [Information Interfaces and Presentation]**: Group and Organization Interfaces—*Organization Design;* ***J.4 [Social and Behavioral Sciences]****: Economics, Psychology, Sociology;* ***K.4.3 [Computers and Society]****: Computer-supported cooperative work*

General Terms: Service systems, systems thinking, pattern language

Additional Key Words and Phrases: Design thinking,

ACM Reference Format:

Ing, D. 2014. From Environmental Structure to Service Systems Thinking: Wholeness with Centers Described with a Generative Pattern Language. ACM Trans. Pattern Languages of Programming. n, n, Article n (October 2014), nn pages.

1. INTRODUCTION: SERVICE SYSTEMS THINKING AIMS TO BUILD ON CHRISTOPHER ALEXANDER'S APPROACH AS A FOUNDATION

Service systems thinking is proffered as a label for an emerging body of work that: (i) builds on systems thinking extending social systems science (i.e. socio-psychological, socio-technical and socio-ecological systems perspectives) into service systems science; (ii) advances a transdisciplinary appreciation of service science, management, engineering and design (SSMED); (iii) explores the practices of architectural design in Christopher Alexander's work on generative pattern languages; and (iv) collaborates through a multiple perspectives inquiring system with the new federated wiki platform. This endeavour is seen as a community activity that could take ten years to mature.

This article aspires to engage the pattern language community not only to repurpose the broad range of pattern catalogs already developed across the broad range of domains, but also to more deeply appreciate Christopher Alexander's clearer articulation of generative pattern languages in his later writings.

In brief, service systems thinking can be described both with an intentional representation and an object-process representation.

Author's address: D. Ing, 151 Booth Avenue, Toronto, Ontario, Canada M4M 2M5; email: coevolving@gmail.com

In an *intentional representation*, service systems thinking is a resource that can be applied by service scientists, managers, engineers and designers.

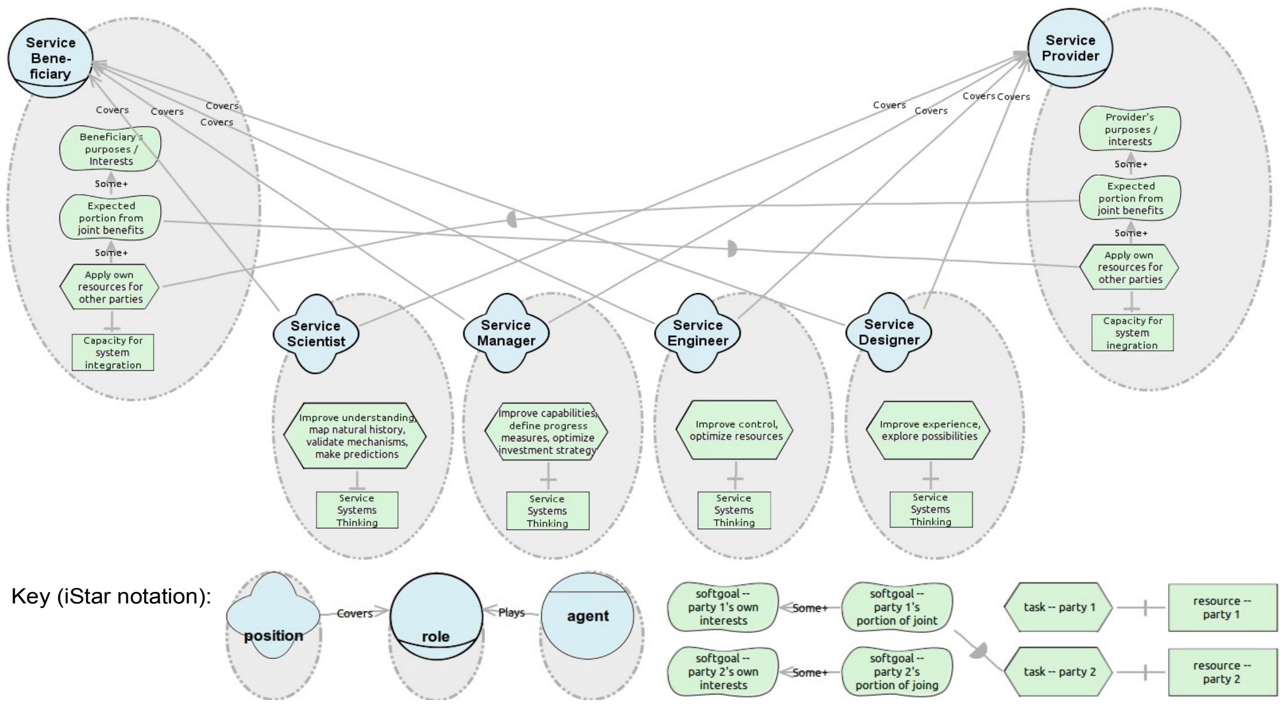


Illustration 1: Service Systems Thinking -- An intentional perspective

Illustration 1 depicts a service system with two roles: a beneficiary and a provider, using an i* (pronounced eye-Star) notation (Horkoff and Yu 2006). Each role has its own softgoals of purposes and interests. The expected portion of joint benefits from the relationship depends on the combination of resources (as hardgoals) that are applied by the other parties and itself. Among the resources at hand for each role is the capacity for system integration

Each of the service beneficiary and service provider roles may be covered by a position. A service scientist position has hardgoals to improve understanding, map natural history, validate mechanisms and make predictions; a service manager position has hardgoals to improve capabilities, define progress measures and optimize investment strategy; a service engineer position has hardgoals to improve control and optimize resources; a service designer position has hardgoals to improve experience and explore possibilities (Spohrer and Kwan 2009).

Service systems thinking could be a resource that supports the hardgoals for all of these positions, as a cross-disciplinary platform for communicating.

In an *object-process representation*, service systems thinking (as a process) is related to a service systems thinking community (as an object). Illustration 2 depicts that service systems thinking is handled by the service systems thinking community, using OPM notation (Dori 2006). Service systems thinking exhibits systems thinking (a process), SSMED (an object), generative pattern language (an object) and multiple perspectives open collaboration (a process).

The services systems thinking community handles four processes: conversations for orientation, conversations for possibilities, conversations for action, and conversations for clarification (Winograd 1986).

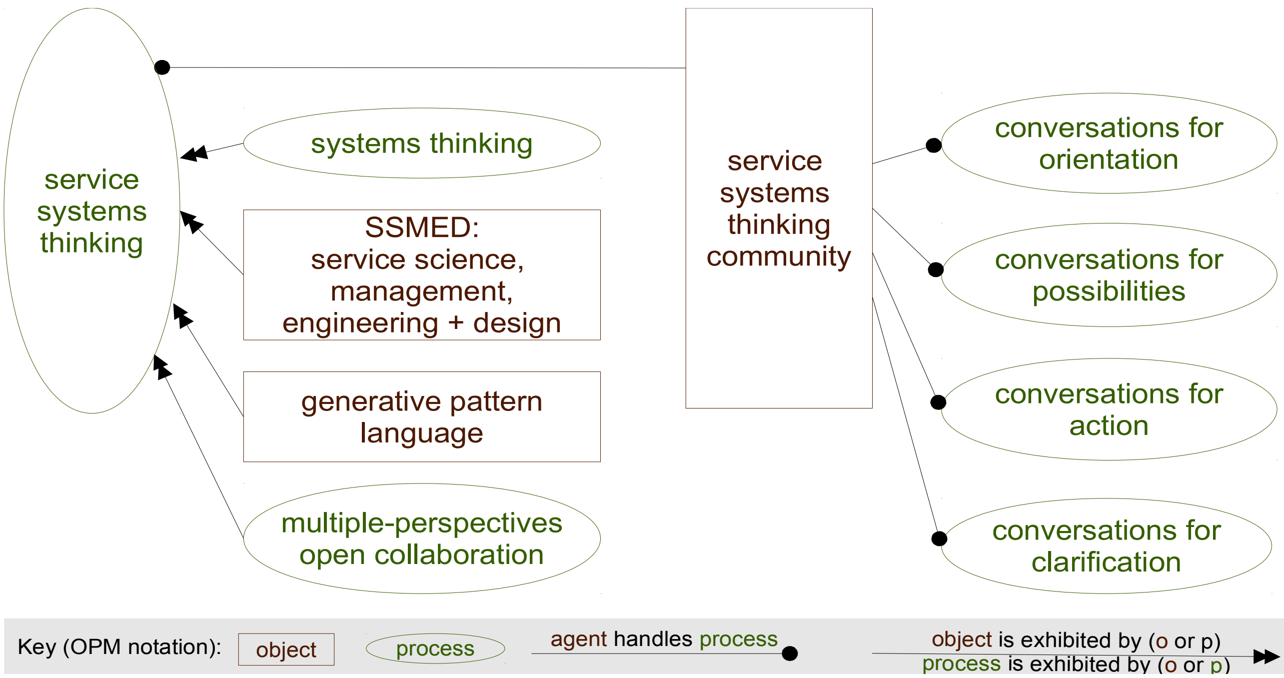


Illustration 2: Service Systems Thinking: An object-process representation

The service systems thinking community is still in a formative phase. This article provides orientations on SSMED and generative pattern language. Content on advances in systems thinking in the 21st century can be covered through alternative orientations (Ing 2013). Multiple perspectives open collaboration has been implemented in a new federated wiki technology where orientations are better presented through web video and hands-on learning (Cunningham 2012).

Section 2 of this article describes key features in the science of services systems that may reframe the approach to a generative pattern language. Section 3 traces the development of ideas by Christopher Alexander over 50 years, and highlights writings where his worldview is clarified.

Section 4 explores possibilities for service systems thinking, as questions in which alternative paths forward warrant collaboration. This article concludes in Section 5, recounting the activities which have taken place to date.

2. ORIENTATION: DISTINCT FEATURES IN SERVICE SYSTEMS INCLUDE COPRODUCTION, OFFERINGS, VALUE AND RESOURCES

The centrality of services in human activity was recognized in the 20th century with *service management* (Normann 1984), but the call for a science of service systems did not come until the 21st century. This idea was introduced to the service science community in 2005 (Spohrer 2005).

> Over the past three decades, services have become the largest part of most industrialized nations' economies. Yet there's still no widely accepted definition of service, and service productivity, quality, compliance, and innovation all remain hard to measure. Few

> researchers have studied service, and institutions have paid little attention to educating students in this area (Spohrer et al. 2007).

In a concise orientation to some key features in service systems, the content for appreciating the domain is described in section 2.1. Coproduction is outlined in section 2.2; offerings are defined in section 2.3; inquiry into value in service science is described in section 2.4; resources are analyzed as operand and operant in section 2.5; and actors and intentions in service systems are introduced in section 2.6. In section 2.7, the progress on a science of service systems is compared to the development of computer science from its origins.

2.1 Service systems dominate human activity in more developed countries

Our everyday lives have service systems omnipresent in technical, organizational and socio-political forms. We are immersed in service systems, so developing a greater appreciation just requires drawing attention to them. A proposed curriculum for primary and secondary schoolchildren illustrates how much of civilization we take for granted.

- Systems that *move, store, harvest and process* include transportation; water and waste management; food and global supply chains; energy and energy grids; and information and communication technology (ICT) infrastructure.
- Systems that *enable healthy, wealthy and wise people* include building and construction; banking and finance; retail and hospitality; healthcare; and education (including universities).
- Systems that *govern* include cities; regions and states; and nations (Spohrer and Maglio 2010).

Table 1: Types of service systems (adapted from Spohrer and Maglio 2010)

Systems that move, store, harvest, process	Transportation	K
	Water and waste management	1
	Food and global supply chain	2
	Energy and energy grid	3
	Information and communications (ICT) infrastructure	4
Systems that enable healthy, wealthy and wise people	Building and construction	5
	Banking and finance	6
	Retail and hospitality	7
	Healthcare	8
	Education (including universities)	9
Systems that govern	Government (cities)	10
	Government (regions / states)	11
	Government (nations)	12

The order of these service systems ranges roughly from the more concrete to the more abstract. Kindergarten children could learn about transportation systems as they travel from home to school. Grade 1 students could visit a water treatment plant. By Grade 2, students could learn how food reaches their dinner tables. The most abstract service systems are provided by governments, better explored in later high school.

While defining "service" has been approached by a wide variety of perspectives, describing a "service system" compatible with a systems thinking worldview is rarer. A publication oriented towards innovation for education, research, business and government by the University of Cambridge proposed a concise wording:

> A service system can be defined as a dynamic configuration of resources (people, technology, organisations and shared information) that creates and delivers value between the provider

> and the customer through service.
>
> In many cases, a service system is a complex system in that configurations of resources interact in a non-linear way. Primary interactions take place at the interface between the provider and the customer. However, with the advent of ICT, customer-to-customer and supplier-to-supplier interactions have also become prevalent. These complex interactions create a system whose behaviour is difficult to explain and predict (IfM and IBM 2008).

In the $54 trillion system of systems in our world, improvement is seen as a $4 billion challenge (IBM 2010). This challenge could be taken up by a variety of disciplinary professions. Service scientists could aim to improve that basic understanding of service systems, mapping their natural history, and validating mechanisms so that better predictions could be produced. Service managers might then have a better foundation on which to improve capabilities, define progress measures, and optimize investment strategies. Service engineers would have an applied science in which they could improve control and optimize resources. Service designers might take a lead in improving service experiences, and exploring the possibilities for better value propositions and government mechanisms (Spohrer and Kwan 2009). Service systems thinking could serve as a crosswalk to bridge disciplinary mindsets and language for more effective collaboration.

2.2 Service providers help customers create value for themselves, as coproducers

A service system, by definition, has multiple parties in interaction. Mechanistic conceptions of systems as producer-product, e.g. economic depictions of value chains, or engineering depictions of supply chains, tend to emphasize parts as independent with low-intensity interactions as handoffs. Interactive concepts of systems see parts (in nature) or roles (in human interactions) as coproducers. Coproduction is expressed as "*the* most critical concept" in purposeful systems (Ackoff and Emery 1972, 23). Richard Normann grounded much his work in systems theory.

> What is new is not co-production, but the way it now expresses itself in terms of role patterns and modes of interactivity. The characteristics of today's economy naturally reshape co-productive roles and patterns. The distinction between "producer" and "consumer", or "provider" and "customer" is ever less clear as the business landscape takes more of a "service" mode (Normann 2001, 96).

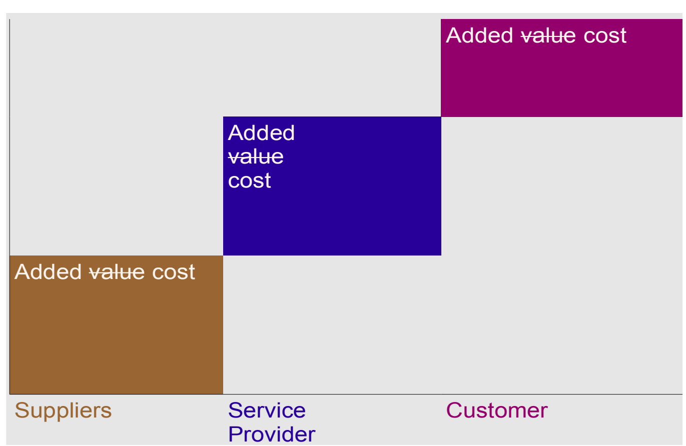


Illustration 3: Not added value; added cost

A production system can produce with only a producer. A service system presumes at least two parties, and may serve not only the customer who consummates the transaction, but potentially also additional downstream beneficiaries and upstream suppliers. Rather than analytically focusing on bilateral relations, a *value constellation* approach draws a more inclusive boundary around a larger set of involved parties.

With multiple interactions between parties taking place within a value constellation, the idea of a "value chain" with "added value" at each stage shown in Illustration 3 is dissolved into a representation of added costs accumulated sequentially in interactions.

> Our traditional about value is grounded in the assumptions and the models of an industrial economy. According to this view, every company occupies a position on the value chain. Upstream, suppliers provide inputs. The company then adds values to these inputs, before passing them downstream to then next actor in the chain [whether another business or the final consumer] (Normann and Ramirez 1993, 65).

This "assembly line" mindset is more appropriate in a world where demand exceeds supply, so that production lines are optimized for greatest efficiency, and the variety available to customers is low. In a world where supply exceeds demands, the interactions between parties can have higher variety.

Let's flesh out the Ikea example that is commonly presented as an example. A mechanistic value chain perspective "follows the money" with the provider signatory (e.g. Ikea) providing an output, and the customer signatory (e.g. the father of a family as purchaser) paying an additional profit for acquisition.

Alternatively, in an interactive value constellation perspective depicted in Illustration 4, let's recognize four parties: (i) the suppliers (e.g. foresters, furniture makers); (ii) the provider signatory (e.g. Ikea, as the prime mover orchestrating the design, manufacturing and distribution); (iii) the customer signatory (e.g. the father who foots the bill for the purchase); and (iv) the beneficiary stakeholders (e.g. other family members in the home who enjoy the furniture). All four parties can be seen as coproducers in the service system. The interactive value of primary interest should be value in use, i.e. by family members enjoying the furnishings for many years after the father has executed on the transaction of purchase. That interactive value is a distinct from the profits that the provider signatory (e.g. Ikea) gains.

> IKEA is able to keep costs and prices down because it has systematically redefined the roles, relationships and organizational practices of the furniture business. [….]
>
> IKEA wants its customers to understand that their role is not to consume value, but to create it. […] IKEA's goal is not to relieve customers of doing certain things but to mobilize them to do easily certain things they have never done before. Put another way, IKEA invents value by enabling customers' own value-creating activities. … Wealth is [the ability] to realize your own ideas (Normann and Ramirez 1993, 66–67).

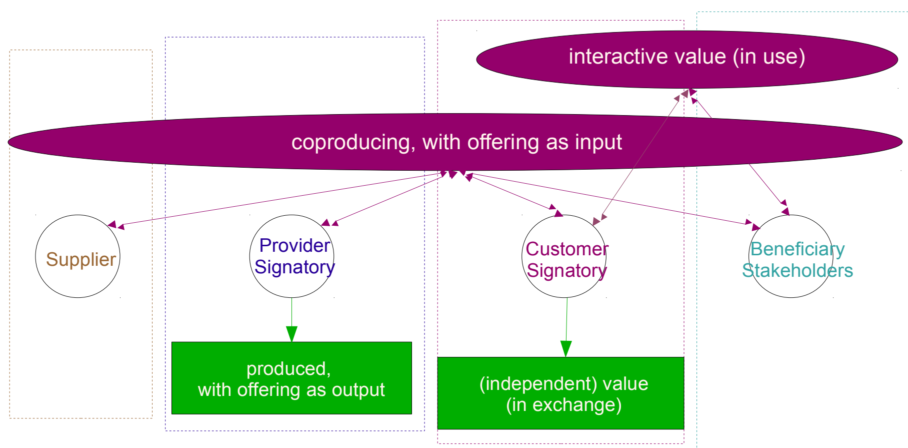


Illustration 4: Enabling interactive value creation

In the illustration, interactive value is depicted as a process where enjoyment takes place over a period of time, as compared to the value in exchange that occurs at only a point in time. In the larger service system, independent transactions are deemphasized relative to the ongoing relationship in the context of mutually changing environments

> From [the] value constellation perspective, value is co-produced by actors who interface with each other. They allocate the tasks involved in value creation among themselves and to others, in time and space, explicitly or implicitly. This opens up many opportunities for defining relationships between actors and reassigning activities. If we look at a single relationship in a co-productive system (for example, that between customer and supplier) this view implies that the customer is not only a passive orderer / buyer / user of the offering, but also participates in many other ways of consuming it, for instance in its delivery. Etymologically, consumption means value creation, not value destruction; this sense of consumption is inherent in the "value constellation" point of view. Furthermore, as actors participate in ways that vary from one offering to the next, and from one customer / supplier relationship to the next, it is not possible to take given characteristics for granted: co-producers constantly reassess each other, and reallocate tasks according to their new values of the comparative advantage each other to have (Normann and Ramirez 1994, 54).

With foundations in systems theory, coproduction is a concept that can be appreciated across the disciplines of science, management, engineering and design, as a common foundation for service systems thinking.

2.3 Offerings are three-dimensional packages either as outputs to, or inputs for, customers

The rise of research into services has led to some confusion of that term. In definitions that emphasize activities or processes with ties between service provision and economic exchange, an implication could be that “everything is a service” (Vargo and Lusch 2004b). This is an unfortunate semantic overloading.

In a clearer definition of a service system, the label of offering is introduced to describe a delivery package in three dimensions, as shown in Illustration 5: physical product content, service and infrastructure content, and interpersonal relationship (people) content. Since any offering coproduced by a value constellation – that could include subcontract, supplier, customer and beneficiary roles – involves contributions by each of the parties, the shape of the delivery package could be different in every interaction.

> … it is useful to examine the offering in terms of a three-dimensional activity package [Illustration 3]. The three axes are *hardware* (or the 'physical product content' of the offering), *software* (the 'service and infrastructure content'), and '*peopleware*' (the interpersonal relationship or 'people content').
>
> - The physical content of the offering consists of elements such as the core product, the packaging, the quality and dependability of the good and its material components, the product range, etc.

- The service content includes distribution, technical support, product modifications, customer training, on-line advice, troubleshooting, warranties and other trust-supporting insurance aspects, information brochures, brand reputation, complaint handling, invoicing, integrated information systems, etc.

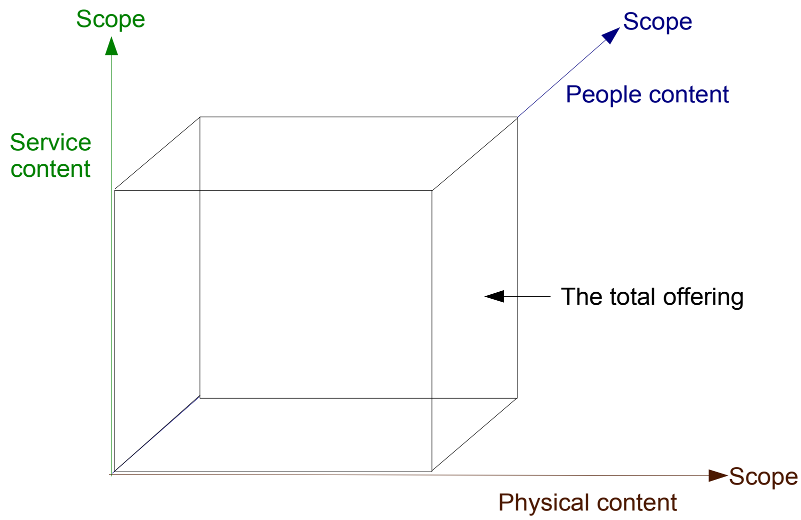


Illustration 5: The three-dimensional offering (Ramirez and Wallin 2000)

- The people content covers issues like long-term partnerships, interpersonal trust, reputation, human resource co-development, etc.

> In keeping with Levitt's view that a product only has meaning from the viewpoint of the customer, different customers will emphasize different axes of the offering.
>
> In co-production terms, the value-creating potential along each of the dimensions of the offering – physical, service or people content – depends on the value-creating system of the customer (Ramirez and Wallin 2000, 58–59).

In this definition of a service system, there are non-service parts to the offering. The way that the customer uses the offering frames its value.

> Offerings are the *output* produced by one (or several) actor(s) creating value -- the 'producer' or 'supplier' -- that becomes an *input* to another actor (or actors) creating value - the 'customer' (Ramirez and Wallin 2000, 47).

Some customers are interested in engaging with a provider for an offering more as an output that requires little or no additional processing, while others want the offering more as an input to be processed with other inputs towards a result with greater value. Customer value can either be derived through transactions or through relationship. The cross of those two dimension leads to the matrix in Illustration 6 (Ramirez and Wallin 2000, 141–145).

- In an *industrial logic* (e.g. 1920s automobile mass production), production cost reductions enable the offering as an output to create value was primarily through an more affordable transaction.

- In a *service logic* (e.g. branded automobiles with models following the customer's age), ensuring continuing customer satisfaction enables an offering as an output to create value primarily through relationship.
- In a *self-service logic* (e.g. do-it-yourself packages), independence and convenience maximization enables an offering as an input to create value through an affordable transaction.
- In a partnership logic (e.g. anticipatory personalization capabilities), value co-development enables an offering as input to create value through an enduring relationship.

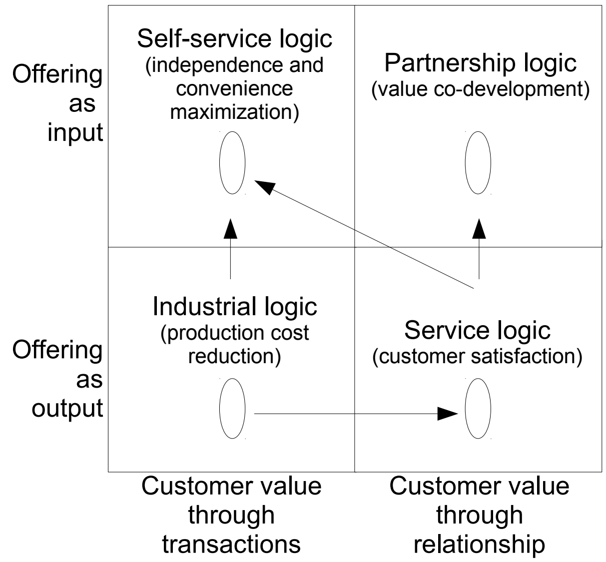


Illustration 6: Alternative views on how offerings and customer relationships interact

The party who designs the offering may be described as the orchestrator or prime mover or the service system capabilities. With an offering as an output, the orchestrator is generally the provider. With an offering as an input, any of the coproducers may rise into a role as orchestrator.

2.4 Value is appreciated interactively by each party in exchange, in use, and in context

Reviewing the academic literature on value, six themes of understanding can be appreciated and mapped into an integrative value framework (Ng and Smith 2012).

From *philosophical* foundations dating back to Plato (360 B.C.E.), value was described as intrinsic (i.e. good to have for itself) and/or extrinsic (i.e. good to have as instrumental to achieve or obtain something else that is good. By 1927, Heidegger proposed an existential philosophy where individuals give meaning to existence in terms of their actions or projects. In 1939, Husserl proposed a phenomenological concept of object conceived in the experience of it. Through Giddens (1979), Chandler and Vargo (2011) argue that individuals and their contexts are mutually constitutive, whereby a context could be simultaneously be a resource for one actor and a deterrent for another actor. All of these views can be labeled as “use-value”.

From *economic* foundations with Adam Smith in 2014, “value in exchange” (i.e. as the power to purchase other goods) was presented as distinct from “value in use” (i.e. as the utility of a particular object. Endowed with invariant properties of goodness and contexts presumed to similarly perceived by all, homogeneity led to a goods-centric focus where products were manufactured in seek of target markets who would perceive value. The experience of use-value after the purchase informing future transactions led to the discipline of marketing.

From *management* foundations, the “selling value” of products circa 1957 evolved by marketers to become exchange value that was superior to competitors. Two firm-centric approaches emerged as (i) the *economic worth of the customer* (EW) in lifetime purchases; and (ii) the *perceived satisfaction of the firm's offerings* (PS) in a stream of repeat purchases. Two preferential judgements of the customer were expressed as (iii) *net benefit* (NB), i.e., the evaluation of outcomes as net difference between the benefits and costs associated with acquiring and consuming an offering, and (iv) *means-end* (ME), i.e.

the evaluation of attributes offering as means towards a goal in the customer's use situations. Evaluating v*alue at the point of choice* can be different from the evaluation at the point of use.

The *modern conceptualization* led by Holbrook (1994) sees value as residing not in an object, a product or possession, but as an "interactive, relativistic preference experience", where the customer is an active participant in its creation. This view was extended in Service-Dominant Logic (Vargo and Lusch 2004a; Vargo and Lusch 2008), with a recapturing of value-in-use. Thus, firms cannot provide value, but only offer propositions of value, with the customer determining the value and the cocreation with the company at a given time and context. Customers are always co-creators of value-in-use contexts, but my not always be co-producers of a firm's offerings.

As a new contribution to service science, P-C-value and A-C-value are presented as a reconciliation and an integration of the preceding conceptualizations. The value being created may sit in different levels of consciousness at different times.

> Block (1977) describes consciousness as being of two types – phenomenal consciousness (P-consciousness) and access consciousness (A-consciousness). P-consciousness is the raw experience of movement, forms, sounds, sensations, emotions and feelings, while A-consciousness is perception, introspection, reflection, in a sense, a more heightened awareness of a phenomenon. This suggests that if we understand value creation as creating something 'good' as an outcome, the consciousness of that goodness during the phenomenological experience may be different from the consciousness of that goodness imagined before, or evaluated after, the phenomenon. One can even argue that within the phenomenon, the actor is merely 'in practice' of resource integrating, with a lower level consciousness of what is 'good', or what is of 'value', from the resources being integrated within the value-creating phenomenon. In other words, even if value is uniquely created within a phenomenon, there could possibly be two levels of consciousness of that value that could exist at different times: P-consciousness of value (P-C-value) or A-consciousness of value (A-C-value) (Ng and Smith 2012, 227–228).

This integration sees that value is not necessarily static, but dynamic according to time (i.e. before, during and/or after the experience).

P-C-value is the creation of value in context that is phenomenal, integrating (i) the existence of the offering, (ii) the affordance of the offering; (iii) the context of the offering in use situations, (iv) agency as the capacity of an actor or entity to act in the world;

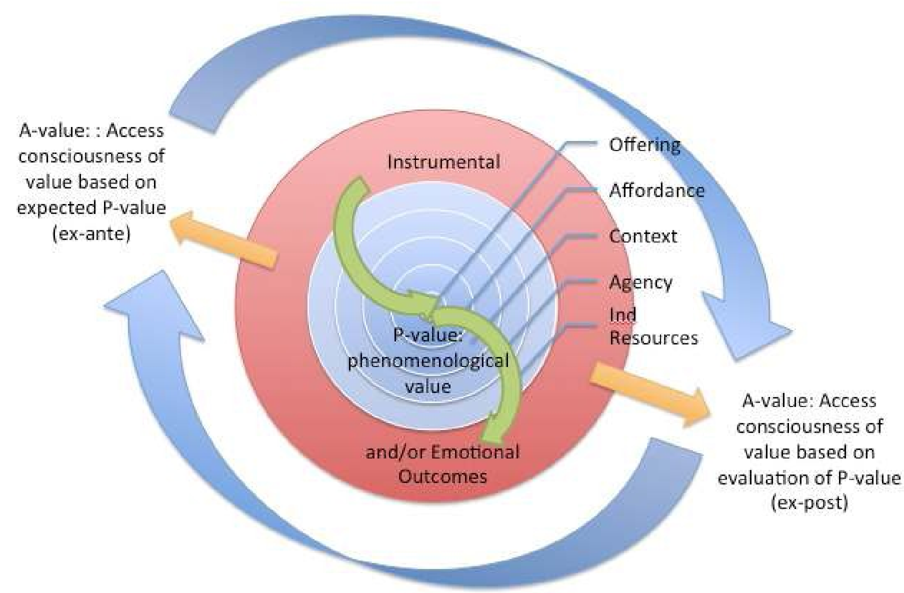


Illustration 7: The Integrated Value Framework (from Ng and Smith 2012)

and (v) actor resources of skills and competencies required to create the P-value of the offering in context.

A-C-value is argued as the perception of goodness that drives choice ex ante and valuation ex post. It is an awareness of goodness at the point of exchange.

The degree of A-C-value ex ante may be related to the P-C-value, which is related to the A-C-value ex post. These relationships have been left for future research.

2.5 While resources were previously considered only operand, service science sees operant resources

The contemporary view on service systems is that they operate in a world where resource not only include "natural resources" that are tangible, but also human ingenuity that is not tangible. This is marked by a shift from Goods Dominant (G-D) Logic to Service Dominant (S-D) Logic.

> In his analysis of world resources, Thomas Malthus (1798) concluded that with continued geometric population growth, society would soon run out of resources. In a Malthusian world, "resources" means natural resources that humans draw on for support. Resources are essentially "stuff" that is static and to be captured for advantage. In Malthus's time, much of the political and economic activity involved individual people, organizations, and nations working toward and struggling and fighting over acquiring this stuff. [….] As we discuss, this change in perspective on resources helps provide a framework for viewing the new dominant logic of marketing.
>
> Constantin and Lusch (1994) define *operand resources* as resources on which an operation or act is performed to produce an effect, and they compare operand resources with operant resources, which are employed to act on operand resources (and other operant recourses). During most of civilization, human activity has been concerned largely with acting on the land, animal life, plant life, minerals, and other natural resources. Because these resources are finite, nations, clans, tribes, or other groups that possessed natural resources were considered wealthy. A goods-centered dominant logic developed in which the operand resources were considered primary. A firm (or nation) had factors of production (largely operand resources) and a technology (an operant resource), which had value to the extent that the firm could convert its operand resources into outputs at a low cost. Customers, like resources, became something to be captured or acted on, as English vocabulary would eventually suggest; we "segment" the market, "penetrate" the market, and "promote to" the market all in hope of attracting customers. Share of operand resources and share of (an operand) market was the key to success.
>
> Operant resources are resources that produce effects (Constantin and Lusch 1994). The relative role of operant resources began to shift in the late twentieth century as humans began to realize that skills and knowledge were the most important types of resources. [….]
>
> Operant resources are often invisible and intangible; often they are core competences or organizational processes. They are likely to be dynamic and infinite and not static and finite, as is usually the case with operand resources. Because operant resources produce effects, they enable humans both to multiply the value of natural resources and to create additional operant resources. A well-known illustration of operant resources is the microprocessor: Human ingenuity and skills took one of the most plentiful natural resources on Earth (silica) and embedded it with knowledge. [….] The service-centered

dominant logic perceives operant resources as primary, because they are the producers of effects. This shift in the primacy of resources has implications for how exchange processes, markets, and customers are perceived and approached (Vargo and Lusch 2004a, 2–3).

This rethinking about focus on resources changes the perspective on how service systems should be considered.

S-D logic implies that "producing" should be transformed into "resourcing." Resourcing allows value creation through collaborative value cocreation, not only involving the provider and the beneficiary but all parties in a value-creation network. Goods remain important in S-D logic, but they are seen as vehicles for resource transmission (what some call appliances or tools), rather than containers of value. [….]

This resourcing conceptualization of service connects well with the concept of service systems as market-facing complex systems [….]

Conceptual Foundations for Service Science

S-D logic, with its process and resourcing orientation, offers a perspective for a conceptual foundation of service science, management, and engineering (SSME), as illustrated in [Table 2]. A critical element of S-D logic involves rethinking the meaning and role of resources. The key distinction is between operand and operant resources (Lusch, Vargo, and Wessels 2008, 7).

Table 2: G-D logic versus S-D logic: A change in perspective (Lusch, Vargo, Wessels 2008)

From G-D Logic	**To S-D Logic**
Operand resources	Operant resources
Resource acquisition	Resourcing (creating and integrating resources and removing resistances)
Goods and services	Servicing and experiencing
Price	Value proposing
Promotion	Dialog
Supply chain	Value-creation network
Maximizing behavior	Learning via exchange
"Marketing to"	Collaborative marketing ("marketing with")

The surfacing of S-D logic perspective, originally developed by Vargo and Lusch, has led to many practitioners reflecting on their preconceptions based on G-D logic, as well as a series of refinements by service researchers (Lusch and Vargo 2006; Vargo and Lusch 2008). For the purposes of service systems thinking, compatibility of S-D logic with systems theory was not as high as with the original concept of offerings by Normann and Ramirez, but academic inquiry continues to work out details.

2.6 Including actors and intentions in service systems models can complement objects and processes

When the word "systems" gets appended to "services", many are predisposed to think about processes. However, services also involve social relationships, where parties coordinate to provide outcomes.

Recent research into service systems has proposed that service system entities – people, organizations and/or partnerships – be represented as intentional agents, to account for intentional and strategic dimensions.

Our notion of intentional agent is drawn from agent-oriented modeling, where agents are viewed as social entities that depend on one another to reach their goals; they thus intentionally enter in relationships with one another to improve their well-being (Yu, 2009).

> *i** (short for distributed intentionality) is an agent-oriented modeling approach that has been developed to support the analysis and design of sociotechnical systems where multiple actors create networks of interdependencies; *i** enables the representation of such a system, as well as the evaluation of different alternatives that could best satisfy actors' goals (Yu, 2002). The use of *i** enables us to represent and analyze service systems at different levels of granularity. It also enables us to design and analyze service system interactions in terms of each entity's motivations. This can complement current process-based design approaches ..., whose focus on sequence of activities and information flows can help to understand *how* value is cocreated in time but do not account for *why* it is so (Lessard and Yu 2013, 69).

The i* modeling framework has been used in requirements engineering, business process design, organization modeling, software development methodologies and evolution. With the Seventh International i* Workshop being held in 2014, the body of knowledge and community has become well-developed. The basic i* notation represents actors and their associations, elements (of resources, tasks, (hard)goals, softgoals and beliefs); and links of dependencies (e.g. strategic, goal, task, resource) (Horkoff and Yu 2006).

> We focus here on mechanisms that emphasize the intentional dimension of service engagements in this domain. Core to such engagements are the benefits that each participating entity expects to gain, in exchange for which it is willing to offer something of value to another entity. Since the other entity will only accept the value proposition if it is beneficial from its own perspective, service system interactions are established in the context of perceived mutual benefits (Vargo, 2009). We have also observed that entities come into relationships with high-level interests, to which the specific benefit that can be obtained from a service engagement contributes. [….] The benefit(s) expected by each entity may then become realized values if the results of the service engagement are evaluated positively, but different determinations of value by each system can lead one system to experience higher value than other systems. At any level of granularity, a service system can thus be understood in terms of the following concepts:
>
> - *High-level interests.* General interests or objectives pursued by a service system.
> - *Expected benefits.* Specific benefits that a service system expects to gain from its *collaboration with another service system.*

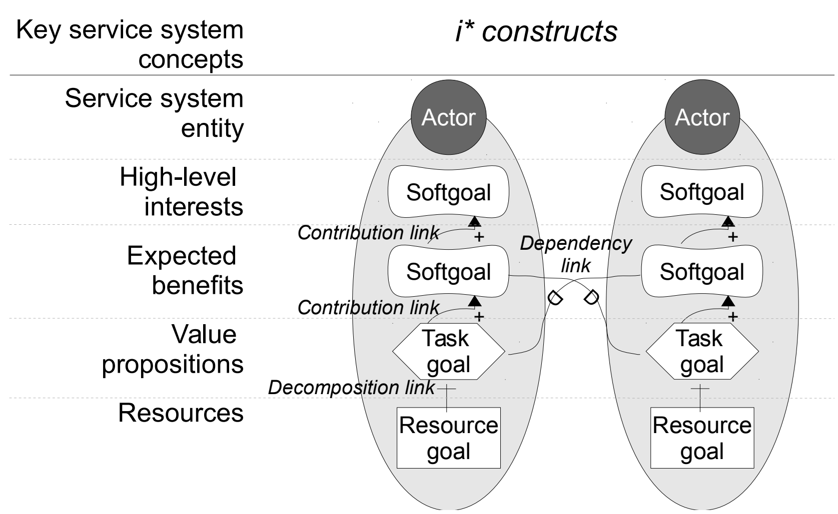


Illustration 8: Express of key concepts of value cocreation through i* modeling constructs (Lessard and Yu 2012)

- *Value proposition.* A service system's proposition to apply its knowledge, skills, and other required resources to produce something of potential benefit to another service system (Lusch et al., 2008).
- *Resources.* Operant and operand resources that can be integrated by a service system to form a value proposition (Vargo & Lusch, 2008).
- *Perceived value*. Positively evaluated outputs and outcomes of a service engagement.

> This understanding of service system value cocreation is in line with current literature but emphasizes a key dimension that has not received attention up to now: the *intentionality* of service systems. Indeed, service system entities are not only composed of resources but also of interests, desires, and needs (Lessard and Yu 2013, 71).

This intentional view represented through i* can complement more traditional modeling of entities and processes. The modelling of software systems conventionally uses UML; the modeling of hardware systems has moved towards SysML.

For conceptual modeling, a simpler alternative may be found in approach consistent with the basic concepts in systems thinking: Object-Process Methodology. OPM takes a strong stance on the fundamentals of systems.

> **Function, Structure, and Behavior: The Three Major System Aspects**
>
> All systems are characterized by three major aspects: function, structure, and behavior. The **function** of an artificial system is its value-providing process, as perceived by the beneficiary, i.e., the person or group of people who gain value from using the system. For example, the function of the organization called hospital is patients' *health level improving*. Each patient is a beneficiary of this system, the customer may be a government or a private entity, and the medical staff constitutes the group of users.
>
> Function, structure, and behavior are the three main aspects that systems exhibit. Function is the top-level utility that the system provides its beneficiaries who use it or are affected by it, either directly or indirectly. The system's function is enabled by its architecture -- the combination of structure and behavior. The system's architecture is what enables it to function so as to benefit its users.
>
> Most interesting, useful, and challenging systems are those in which structure and behavior are highly intertwined and hard to separate. For example, in a manufacturing system, the manufacturing process cannot be contemplated in isolation from its inputs -- the raw materials, the model, machines, and operators -- and its output -- the resulting product. The inputs and the output are objects, some of which are transformed by the manufacturing process, while others just enable it. Due to the intimate relation between structure and behavior, it only makes sense to model them concurrently rather than try to construct separate models for structure and behavior, which is the common practice of current modeling languages like UML and SysML. The observation that there is great benefit in concurrently modeling the systems structure and behavior in a single model is a major principle of OPM.

> **Structure** of a system is its form -- the assembly of its physical and logical components along with the persistent, long-lasting relations among them. Structure is the static, time-independent aspect of the system. The **behavior** of a system is its varying, time-dependent aspect, its dynamics -- the way the system changes over time by transforming objects. In this context, transforming means creating (generating, yielding) a new object, consuming (destructing, eliminating) an existing object, or changing the state of an existing object.
>
> With the understanding of what structure and behavior are, we can define a system's architecture.
>
> > **Architecture** of a system is the combination of the system's structure and behavior which enables it to perform its **function**.
>
> Following this definition, it becomes clear why codesign of the system's structure and behavior is imperative: they go hand in hand, as a certain structure provides for a corresponding set of system behaviors, and this, in turn, is what enables the system to function and provide value. Therefore, any attempt to separate the design of a system, and hence its conceptual modeling, into distinct structure and behavior models is bound to hamper the effort to get close to an optimal design. One cannot design the system to behave in a certain way and execute its anticipated function unless the ensemble of its interacting parts of the system -- its structure -- is such that the expected behavior is made possible and deliver the desired value to the beneficiary (Dori 2011, 216–217).

The entities in OPM include two things, (i) *objects* and (ii) *processes*, which are modeled as first class citizens in an *object-process equality principle*. The third entity in OPM is a *state*, defined as a situation in which an object can be at some point in time. Links are used to connect the three entities in Object Process Diagrams.

In formal definitions:

> An **object** is a thing that exists or can exist physically or informatically (Dori 2011, 223).

This is a structural, timeless view of the world at moment of time. This definition is more general that that normally used for object-oriented development of information systems.

For the temporal perspective, a definition of *transformation* is invoked so that time-dependent relationships amongst things are representable.

> **Transformation** is the generation (construction, creation) or consumption (destruction, elimination) or change (effect, state transition) of an object (Dori 2011, 224).

The existence of an object could be changed through a transformation, or some of its attributes could be changed over time. Thus,

> A **process** is a transformation that an object undergoes (Dori 2011, 225).

This definition of a process requires the existence of at least one object. An object can have states; a process can have subprocesses.

In the English language, a noun can sometimes mean either an object or a process. While the default is to assume a noun is an object, the object-process distinction says to classify a given noun as a process if an only four process criteria are met: (i) object involvement; (ii) object transformation; (iii) association with time; and (iv) association with verb (Dori 2011, 227).

OPM employs both graphical and text to reduce the cognitive load of interpreting a model. Software tools can map from the graphical Object-Process Diagram (OPD) to the textual Object-Process Language (OPL). Illustration 9 show an example constructed in the Opcat tool.

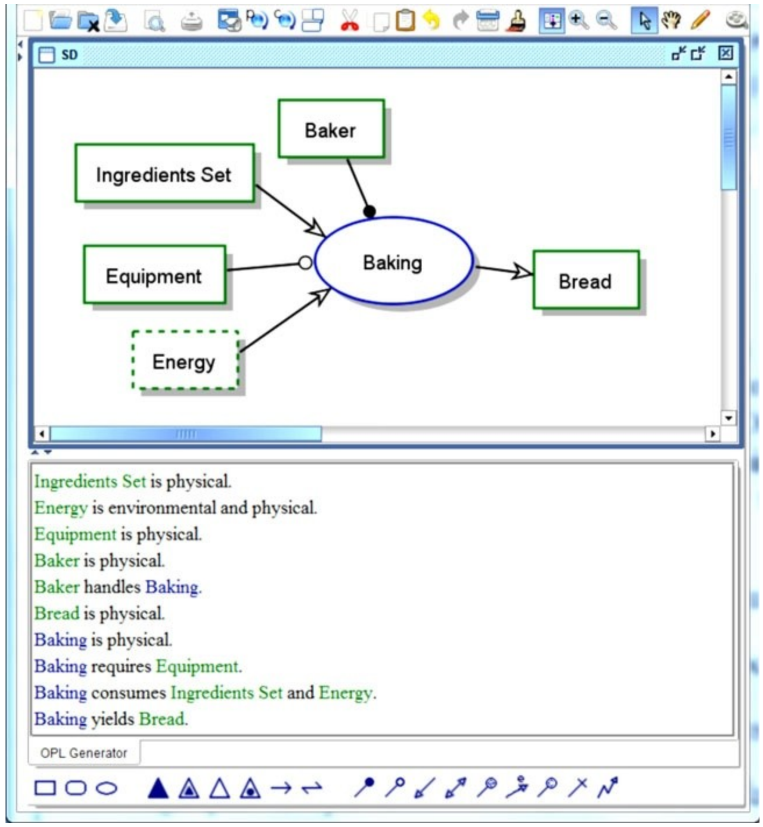


Illustration 9: A baking system, with the Object-Process Diagram (OPD) above and Object-Process Language (OPL) below (from Dori 2011, 212)

> For example, **Baking**, the central system's process, is the ellipse in [Illustration 9]. The remaining five things are objects (the rectangles) that enable or are transformed by **Baking**. **Baker** and **Equipment** are the *enablers* of **Baking**, while **Ingredients Set**, **Energy**, and Bread are its *transformees* -- the **objects** that are transformed by Baking. As the direction of the arrows indicates, **Ingredients Set** and **Energy** are the *consumees* -- they are consumed by **Baking**, while **Bread** is the resultee -- the object created as a result of Baking. As soon as the modeler starts depicting and joining things on the graphics screen, OPL sentences start being created in response to these inputs. They accumulate in the OPL pane at the bottom of [Illustration 9], creating the corresponding OPL paragraph, which tells in text the exact same story that the OPD does graphically.
>
> As the example shows, the OPL syntax is designed to generate sentences in plain natural, albeit restricted, English, with sentences like "**Baking** yields **Bread**." This sentence is the bottom line in Fig. 7.1. An English subset, OPL is accessible to nontechnical stakeholders, and other languages can serve as the target OPL. Unlike programming languages, OPL names can be phrases like **Ingredients Set** (Dori 2011, 212–213).

To progress communications in service systems thinking, making a distinction between (i) the intention-oriented perspective through i*, and (ii) the function-structure-behavior perspective in OPM is worth consideration. Although interests, benefits, value propositions and resources could be represented in OPM as well as in i*, their primacy of these elements in a service system calls for ways to increase their salience.

2.7 Service systems science has a promise to synthesizes disciplines, as did computer science

Service systems thinking, as a new field, will draw heavily on a foundational service science that has its origins only as recently as 2005. The prior experiences of IBM in the emergence of a new science of computing are a parallel. In the 1970s, the IBM Research organization was composed of physicists, chemists, electrical engineers and mathematicians. To respond to business changes requiring software systems research, new Ph.D.s joined the organization in large numbers.

> Some colleagues in IBM and in academia advocated a bold approach– creating a new academic discipline called service science (Chesbrough 2004, 2005; Horn 2005), which aims theories and methods from many different disciplines at problems that are unique to the service sector. At the start, the particular disciplines (including some engineering, social science, and management disciplines) and the particular problems (e.g., improving service innovation and service productivity) were not clear. However, this idea of an integrated service science was particularly appealing to us, as we found that the number of separate PhDs required to form a suitable services research organization had grown to nearly a dozen! We had recruited PhDs in anthropology, cognitive psychology, computer science, cognitive science, education, human factors, industrial engineering, and organizational psychology, among others. The communication challenge alone of getting such a diverse population of scientists to speak a common language around "service innovation" required training everyone in each others' disciplines to some extent, as well as injecting new, practical concepts fresh from the front lines of our own services business (Spohrer and Maglio 2008, 239).

The feature of coproduction, offerings, values and resources described above in Section 2 have cross business strategy, marketing, psychology, economics, computer science and philosophy. Improving communications across and amongst the disciplines into a new field is a challenge that may require a generation to new scholars to become fully institutionalized.

To the disciplines described above, Section 3 explores contributions from the architecting and design of built environments that had previously been cross-appropriated to computer science. Wisdom from decades of practice in those fields can inform the development of service systems thinking.

3. ORIENTATION: THE HISTORY OF ARTICULATIONS BY CHRISTOPHER ALEXANDER ARE SALIENT TO SERVICE SYSTEMS

Christopher Alexander is best known for this 1977 book, *A Pattern Language*. That work, however, was only one point in a evolving body of work, with major publications ranging from the *Pattern Manual* in 1967 to *The Battle for Life and Beauty* in 2012. Alexander's progress as a theorist was formed throughout his activity as a practicing architect and as builder of dozens structures. Ideas became clarified with time, and Alexander's articulations sharpened. "Diagrams" became "patterns". "Quality without a name" became "unfolding wholeness". "Structure extending transformations" became "wholeness-extending transformations". "Living centers" became "systems of centers".

This evolution of articulations can be examined through a historical retrospective of the context and content of Alexander's work. In section 3.1, the emergence of architectural programming as problem seeking in the late 1960s reflects the way in which leading architects were practicing, and new architects were being trained. In section 3.2 between 1964 and 1971, the design problem of fit between the form of the built environment and its containing context was related to “diagrams of forces” that became patterns. Section 3.2 describes the 1967 formation of the Center for Environment Structure with the earliest description of the pattern format. Section 3.3 cites a 1968 publication where the feature of generativity in the pattern language was made explicit, with systems thinking foundations. Section 3.5 reviews the work on multi-service centers around the same time in 1968, where a pattern language and ranges of contexts for prototypes of sites in eight cities were demonstrated. In section 3.6, the well known 1975-1979 publications of the popular *The Oregon Experiment*, *A Pattern Language* and *The Timeless Way of Building* becomes the context for the 2001-2004 *The Nature of Order* volumes, where examples of living structure and processes for creating them were illustrated and theorized. Section 3.7 reviews the 2012 Battle for Life and Beauty of the Earth where unfolding wholeness through local adaption was demonstrated in the building of the Eishin campus circa 1985.

The philosophy for an Alexandrian design process has been described as ateleological, in contrast to the teleological style followed by most architects. This perspective in presented in section 3.8.

3.1 Circa 1969, architectural programming was envisioned as problem seeking, preceding design as problem solving

The context for a generative pattern language has its roots in architectural programming. While Christopher Alexander was appointed as a research professor at U.C. Berkeley in 1965, and the Center for Environmental Structure was formed in 1967, the idea of architectural programming was documented in 1969 by practitioners from *Caudill Rowlett Scott*, Architects, Planners, Engineers in Houston Texas. The distinction between seek problems in architectural programming and problem solving in design is often confused. A clarification is depicted in Illustration 10, with its supporting text.

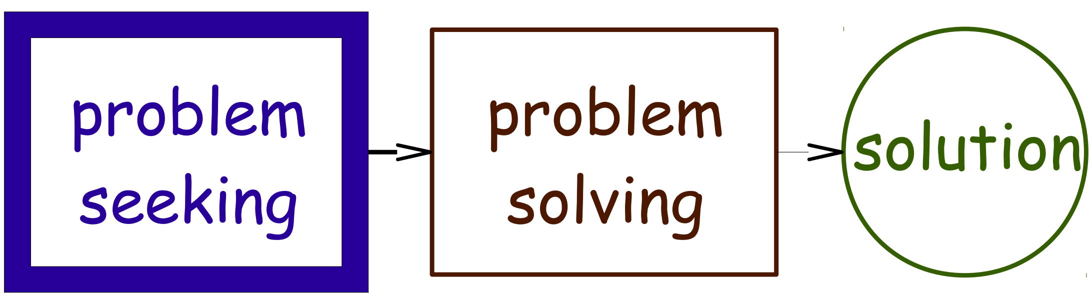


Illustration 10: Programming is problem seeking, design is problem solving (Pena and Focke 1969)

Programming is a specialized and often misunderstood term. It is “*a statement of an architectural problem* and the requirements to be met in offering a solution. While the term is used with other descriptive adjectives such as *computer* programming, *educational* programming, *functional* programming, etc., in this report, programming is used to refer only to architectural programming.

Why programming? The client has a project with many unidentified sub-problems. The architect must define the client's total problem.

Design is problem solving; programming is problem seeking. The end of the programming process is a statement of the total problem; such a statement is the element that joins programming and design. The "total problem" then serves to point up constituent problems, in terms of four considerations, those of form, function, economy and time. The aim of the programming is to provide a sound basis for effective design. The State of the Problem represents the essense and the uniqueness of the project. Furthermore, it suggests the solution to the problem by defining the main issues and giving direction to the designer (Pena and Focke 1969, 3).

Architects that rush into problem solving without adequate exploration of problem seeking constrain the resulting design prematurely. Alternative ways of bounding choices of the site and structure could preempt design challenges later. The description of site and structure shearing layers that change more slowly than the enclosed services, space plan and stuff illustrates constraints in design placed through architectural programming (Brand 1994; Brand and Runice 1997). The label of "shearing layers" has been subsequently generalized beyond built environments to a broader variety of systems as "pacing layers" (Brand 1999).

Architectural programming is a negotiation of constraints with the sponsoring client and/or program beneficiaries, either explicitly or implicitly led by the architectural team. The program beneficiaries are the long-term occupants and/or users of the results, and the sponsoring clients should act on their behalf to engage an architectural team to facilitate programming. The elicitation and capture of both unarticulated and explicitly articulated values, wants and needs rubs up against the presentation of conceptual alternative programs made more tangible by drawings, scale models and/or site visits. In the 1969, Caudill Rowlett Scott staff described their analytical framework as an exemplary way to collect information that would lead to a good architectural program.

How Much Information is Enough?

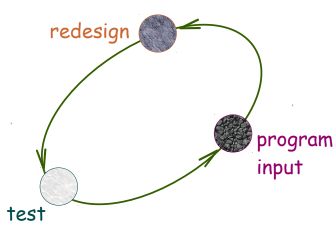


Illustration 11: Programming through design, testing and redesign is inefficient (Pena and Focke 1969)

If a client approaches the architect with very little information, the architect may have to respond by programming through design. He could produce sketch after sketch and plan after plan trying to satisfy undefined requirements. Programming through design can involve misuse of talent and, indeed, risks of creating a "solution" to the wrong problem. [See Illustration 11].

On the other hand, a client may present the architect with too much information but involving mostly irrelevant details. The risk here is that the architect's solution will be based on details rather than major ideas. In this case, the architect must plough through an abundance of information and discriminate between major ideas and details. [See Illustration 12].

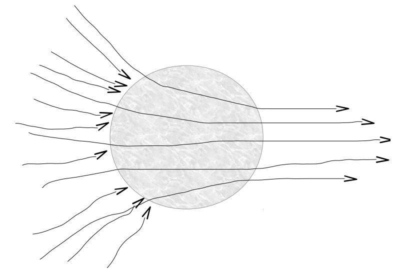


Illustration 12: Discrimination between major ideas and details is necessary to avoid confusion in problem solving (Pena and Focke 1969)

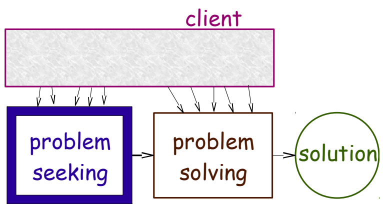


Illustration 13: The client is involved in the process (Pena and Focke 1969)

The analytical procedure used by CRS provides a framework for decision making. Within it the architect help the client identify and make decisions that need to be made prior to design. Within it, the architect can suggest alternatives and other information to bring about decisions. There are times when the architect must evaluate the gains and risks in order to stimulate a decision. Yet, note the emphasis on client decisions; the architect merely participates and at most, recommends. [See Illustration 13]

The new sophisticated client wants to know how his project will be processed and when he will be involved. He wants to remove the mystique associated with the programming and design of his project (Pena and Focke 1969, 4–6).

Architectural programming is prescribed as an engagement between the client and the architectural team. It's the client that is supposed to make the decisions, with the architect facilitating the process. There's a fine line between the architect guiding the client to be clear about the wants and needs of the program beneficiaries, and recommending with professional knowledge on ways that the bounding of the program at early conceptual phases will constrain later design decisions.

The separation of programming from design should be clear. In this architectural practice, the roles of the programmer and the designer are distinct.

Two terms need to be understood and added to the glossary of architectural practice: "Programmatic concepts" and "design concepts." Programmatic concepts refer to the ideas intended mainly as solutions to the client's own management problems so far as they concern function and organization. Design concepts, on the other hand, refer to ideas intended as physical solutions to architectural problems.

Programmatic concepts and design concepts are so closely related that one is mistaken for the other. Design concepts are the physical response to programmatic concepts. For example, *open planning* is the physical response to *integration* of activities. In practice the confusion is compounded because most architects and some clients tend to think more easily in physical terms.

Programmatic concepts must be stated abstractly so as not to inhibit design alternatives unnecessarily. For example, the programmatic concept of *decentralization* may find a design response in either *compactness* (vertical or horizontal) or *dispersion* (varying degrees) (Pena and Focke 1969, 6–7).

Architects have the challenge of expressing abstract concepts in a way that are concrete to sponsoring clients and the eventual beneficiaries. There's an analogy in the eye examinations given by optometrists. While optometrists today can approximate an assessment of optical fitness with laser instrumentation, the final choices are made with an eye chart and pairwise comparison of lenses with a dialogue of “which is better, A or B”?

Architectural programming is balance of function, form, economy and time. Here, the layman who has not experienced a full construction program and project will be handicapped. The architectural team should have more experience to be able to describe how program constraints set today will impact the design that will follow. The architectural program will be budgeted in time, with function, form and economy as considerations.

> **The Four Basic Considerations**
>
> If design of the facility is to solve problems of function, form, economy and time, then programming must treat these as basic considerations by which to classify information. [See Illustration 14].
>
> The first of these, *function*, deals with the functional implication of the client's aims, methods to be used to meet them, and numbers and types of people. It deals with social and functional organization. Contributions to the client could be by management consultants, behavioral scientists, and architects with intuitive insights into social values.

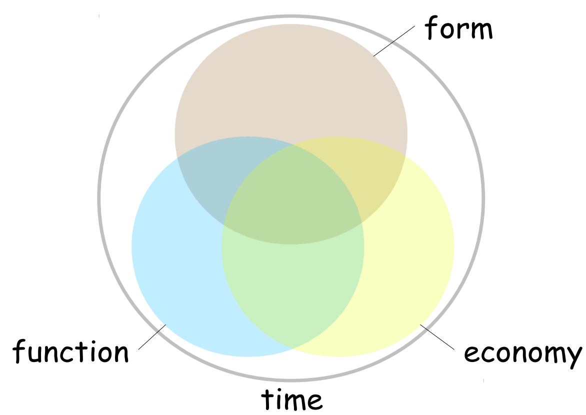


Illustration 14: The whole problem consists of the consideration for form, function, economy and time (Pena and Focke 1969)

> *Form*, the second consideration, is used by CRS to evoke questions regarding the physical and psychological environment to be provided, the quality of construction and the conditions of the site. The physical environment involves physical needs such as illumination, heating, ventilating, air-conditioning and acoustics. The psychological environment raises values which might affect user behavior; the architect must inject these intuitively until such time as analytical means are developed.
>
> The third consideration, *economy*, emphasizes the need for early cost control and brings up for consideration by the programming team the initial budget, the operating cost and long term cost which may be affected by initial quality of construction.
>
> Consideration four, *time*, brings out the factors of change and growth, which affect function, form and economy (Pena and Focke 1969, 14–16).

Time is the ultimate constraint, as some building materials (e.g. concrete) can only be accelerated so much. The mapping between form and function is not one-to-one, but many-to-many. This many-to-many mapping is a reason while architectural programming should be decoupled from the design of slower-changing and faster-changing pacing layers. Economy is associated most concretely with choices on form, which are influenced by the prioritizations on function.

When programming is done properly, the wants and needs of the client are appreciated not as static functional specifications, but instead as goals that may evolve. An individual inexperienced with constructions projects may only be thinking about the first day of occupancy, rather than the longer term phenomenon of living in a built environment where modifications and adjustments may be implemented over the span of many years or decades. A systemic approach could be evident in "negotiable programming".

> Building Systems and "Negotiable Programming"
>
> The expanding trend to system building affects the entire building project delivery process. In programming terms, a resolve to use building systems is a goals-oriented decision which is tested at the first (goals formulation) step in programming, and, if verified, will affect program content.
>
> The use of system building makes possible a more general, flexible form of programming conveniently referred to as "negotiable programming". Negotiable programming presupposes that the building has been developed from user requirements and performance criteria, and that it will produce the kind of flexibility that will make net space requirements "negotiable" within a fixed gross area. The aim is to make the end product a building with the flexibility to change as user requirements change.
>
> Through recourse to system building every program requirement remains negotiable throughout the design and building process, and because of inherent flexibility the functional organization of the interior remains always negotiable (Pena and Focke 1969, 36–37).

The negotiation in an effective architectural program is not the engagement between the client and the architectural team, but instead an engagement between the occupants and/or beneficiaries of the built environment and that completed construction (Parhankangas et al. 2005). The finished building becomes a constraint to socio-technical and socio-ecological interactions amongst human beings in a physical environment. In the pacing layers framework, it's easy to move stuff such as furniture, changing the space plan requires carpenters, and more extensive renovations that impact services will require plumbers and electricians.

While the challenges of problem seeking were framed by Caudill Rowlett Scott staff for buildings, the ideas are clearly applicable to larger scale built environments such as neighbourhoods and cities. Landscape features such as rivers and hills bound decisions on choosing a site. Once rails and streets, water and sewers, and electrical infrastructure is put in place, subsequent architectural programming is constrained. Beyond built environments, the isomorphies promised in systems thinking may aid clearer appreciation of boundaries, function and form.

3.2 Circa 1964 to 1971, the design problem of fit between a form and its context was related to "diagrams of forces" that later became known as patterns

In Christopher Alexander's 1964 publication of *Notes on the Synthesis of Form*, the label "pattern" had not yet been introduced. In the preface to the paperback edition published in 1971, the change in label became explicit.

… diagrams, which, in my more recent work, I have been calling *patterns*, are the key to the process of creating form. [….]

The idea of a diagram, or pattern, is very simple. It is an abstract pattern of physical relationships which resolves a small system of interacting and conflicting forces, and is independent of all other forces, and of all other possible diagrams. The idea that it is possible to create such abstract relationships one at a time, and to create designs which are whole by fusing these relationships …. (Alexander 1964, i)

To be clear, the diagram is not of a single pattern, but a "diagram of forces" that includes the relations between patterns that make up a language.

For architects, the ultimate end for their efforts is form, where abstract ideas become reality. In the chapter on "Goodness of Fit" between "the form in question and its context", systems thinking shows through. More appreciation of the early ideas comes with inclusion of footnotes to the text.

The ultimate object of design is form.

[….] If the world were totally regular and homogeneous, there would be no forces, and no forms. Everything would be amorphous. But an irregular world tries to compensate for its own irregularities by fitting itself to them, and thereby takes on form.[1] D' Arcy Thompson has even called form the "diagram of forces" for the irregularities.[2] More usually we speak of these irregularities as the functional origins of the form.

1. The source of form actually lies in the fact that the world tries to compensate for its irregularities as economically as possible. This principle, sometimes called the principle of least action, has been noted in various fields: notably by Le Chatelier, who observed that chemical systems tend to react to external forces in such a way as to neutralize the forces; also in mechanics as Newton's law, as Lenz's law in electricity, again as Volterra's theory of populations. See Adolph Mayer, *Geschichte des Prinzips der kleinsten Action* (Leipzig, 1877) .
2. D'Arcy Wentworth Thompson, *On Growth and Form*, 2nd ed. (Cambridge, 1 959) , p. 16.

The functional origins of the form of built environments should ideally come from the beneficiaries who occupy it, but may be (mis-)interpreted through the voices of the sponsoring client, the architectural programming team, and the design team. The forces are parts of the form that work against each other, or could be whole forms working against each other. Alexander continues:

The following argument is based on the assumption that physical clarity cannot be achieved in a form until there is first some programmatic clarity in the designer's mind and actions; and that for this to be possible, in turn, the designer must first trace his design problem to its earliest functional origins and be able to find some sort of pattern in them.[3] I shall try to outline a general way of stating design problems which draws attention to these functional origins, and makes their pattern reasonably easy to see. [p. 15]

3. This old idea is at least as old as Plato: see, e.g., *Gorgias* 474-75.

Programmatic clarity, in the architectural context presented by Pena and Focke, is about problem seeking prior to problem solving. The patterns of interest are in the problems, and not in the solutions. Alexander continues:

It is based on the idea that every design problem begins with an effort to achieve fitness between two entities: the form in question and its context.[4] The form is the solution to the

> problem; the context defines the problem. In other words, when we speak of design, the real object of discussion is not the form alone, but the ensemble comprising the form and its context. Good fit is a desired property of this ensemble which relates to some particular division of the ensemble into form and context.[5] [pp. 15-16]
>
> 4. The symmetry of this situation (i.e., the fact that adaptation is a mutual phenomenon referring to the context's adaptation to the form as much as to the form's adaptation to its context) is very important. See L. J. Henderson, *The Fitness of the Environment* (New York, 1913), page v: "Darwinian fitness is compounded of a mutual relationship between the organism and the environment." Also E. H. Starling's remark, " Organism and environment form a whole, and must be viewed as such." For a beautifully concise description of the concept "form," see Albert M. Dalcq, "Form and Modern Embryology, " in *Aspects of Form*, ed. Lancelot Whyte (London, 1951), pp. 91-116, and other articles in the same symposium
>
> 5. At later points in the text where I use the word "system," this always refers to the whole ensemble. However, some care is required here, since many writers refer to that part of the ensemble which is held constant as the environment, and call only the part under adjustment the "system." For these writers my form, not my ensemble, would be the system.

Apply systems definitions, the fitness between “the form in question and its context” is in the relation between a system and its containing whole. This is not be confused with the relation between a system as a whole and its parts. The context is in the environment for the system of interest, which is part of a larger containing whole. Alexander continues:

> There is a wide variety of ensembles which we can talk about like this. The biological ensemble made up of a natural organism and its physical environment is the most familiar in this case we are used to describing the fit between the two as well-adaptedness.[6] But the same kind of objective aptness is to be found in many other situations.
>
> 6. In essence this is a very old idea. It was the first clearly formulated by Darwin in *The Origin of Species*, and has since been highly developed by such writers as W. B. Cannon, *The Wisdom of the Body* (London, 1932), and W. Ross Ashby, *Design for a Brain*, 2nd ed. (New York, 1960).

The references to Walter Cannon and Ross Ashby indicate that that pattern language – as a “diagram of forces” – of goodness of fit between a form and its context is a cross-appropriation from biology. This provides hope that pattern language may be reappropriated back from built environments to use in other domains, such as service systems thinking.

The call for architectural programming to be conducted before design is implicit in Alexander's 1966 article on “A City of Not A Tree”. Alexander was critical that design problems did not come in a tree-like structure, which would suggest that there's a overarching problem statement that could lead to subproblems in a purely hierarchical form. The problems are not in a completely non-hierarchical organization of a network, but could be described as a semi-lattice.

> Too many designers today seem to be yearning for the physical and plastic characteristics of the past, instead of searching for the abstract ordering principle which the towns of the past happened to have, and which our modern conceptions of the city have not yet found. These designers fail to put new life into the city, because they merely imitate the appearance of the old, its concrete substance: they fail to unearth its inner nature.
>
> What is the inner nature, the ordering principle, which distinguishes the artificial city from the natural city? You will have guessed from the first paragraph what I believe this ordering principle to be. I believe that a natural city has the organisation of a semi-lattice; but that when we organise a city artificially, we organise it as a tree (Alexander 1966).

A natural city would have local planning within its neighbourhoods, where a city organized artificially is planned by a central authority. This article illustrates Alexander's emerging theory on pattern language to be applied not only to buildings, but also to larger scale built environments such as cities. When Alexander is recognized as seeking "life" within the building projects he has engaged, he certainly appreciated "life" in cities with the citation of Jane Jacobs' 1961 *The Death and Life of Great American Cities* in a similar pursuit.

These writings from 1964 and 1966 predate Alexander's appointment to the University of California at Berkeley. The next section sees further refinements on his ideas in collaboration with colleagues who would work with him for some decades to come.

3.3 Circa 1967, the institution on environmental pattern language described the pattern format

The incorporation of the Center for Environment Structure in Berkeley in 1967 formalized a vision as a hub for an environmental pattern language, both putting the pattern language into practice for buildings and cities, and conducting foundational research.

> The Center for Environmental Structure is an independent corporation set up to create an environmental pattern language. The Center will undertake architectural and planning projects within the framework of this language. It was incorporated in late March, 1967, and received tax exempt status as a non-profit corporation from the State of California and the Federal Government. It is based in Berkeley, California. [….]
>
> The Center received starting funds from the Kaufmann Foundation and the Bureau of Standards.
>
> <u>ACTIVITIES</u>
>
> The Center has three main activities. First, the Center will publish, and distribute, the coordinated pattern language, as it evolves. Second, the Center will undertake contracts to develop specific patterns and systems of patterns, within the pattern language, and to design buildings and parts of cities according to the language. Third, the Center will undertake basic research concerning the pattern language (Alexander, Ishikawa, and Silverstein 1967, iii–iv).

In the earliest work on pattern language, the intent comes through in descriptions about the idea. The *Pattern Manual* includes both a discussion of format and concrete examples to make the point. The problem is "what", the pattern is the "where" and range of context is "why". There's an unfortunate overloading of the label "pattern" to describe the triad of "problem, pattern, range of contexts" and the pattern section of the format.

> We begin with the following hypothesis: Every time a designer creates a pattern (or for that matter, entertains <u>any</u> idea about the physical environment), he essentially goes through a three-step process. He considers a PROBLEM, invents a PATTERN to solve the problem, and makes a mental note of the range of CONTEXTS where the pattern will solve the problem. For example, a designer considering the problem of traffic congestion and pedestrian access around central shopping districts might come up with the pattern "Linear pedestrian malls bounded on both sides by rows of shops; parking lots strung along, behind the shops." He would then make a mental note of the kinds of places where this pattern is useful: "Commercial districts serving 300,000 people, where existing streets can be closed

> and paved, with car access evenly distributed behind the stores." This three-step process may be characterized most simply as WHAT (mall between shops, parking behind), WHERE (commercial area serving 300,000), and WHY (ease traffic congestion, create pedestrian access). Of course, the sequences of these three steps is not always the same. Sometimes a pattern is invented before the problem is well understood; sometimes the context comes first, and inspires the creation of a pattern. There is not need to formalize the sequence; it can always be left to quirks of the moment and individual style.
>
> The format proposed here reflects this three-step process. It contains three sections: PATTERN, CONTEXT and PROBLEM.
>
> The format says that whenever a certain CONTEXT exists, a certain PROBLEM will arise; the stated PATTERN will solve the PROBLEM and therefore should be provided in the CONTEXT. While it is not claimed that the PATTERN specified is the only solution to the PROBLEM, it is implied that unless the PATTERN <u>or an equivalent</u> is provided, the PROBLEM will go unsolved.
>
> Every single physical pattern – from the smallest detail in a building, to the distribution of central business districts in an urban region – exhibits this logic; every pattern can be conceived according to this triad: CONTEXT … PATTERN … PROBLEM.
>
>> [We shall also use the word PATTERN to refer to the entire triad; as well as to the central solution of the triad. It will be clear whenever we use the word whether we are talking about the entire triad, or simply the PATTERN section.]
>
> For clarity the three sections should be preceded by a SUMMARY which abstracts the essential idea contained in the body of the pattern (Alexander, Ishikawa, and Silverstein 1967, 2–4).

The foundational hypothesis attempts to capture the thinking process of an experienced designer on a new encounter: issues within a phenomenon are observed as a challenge (problem); an intervention (pattern) that result in a better future state than current state is invented; and the containing conditions (range of contexts) in which the intervention will work are noted. For subsequent reuse, the order of in the format could then be described as intervention (pattern), containing conditions (range of contexts) and issues within a phenomenon (problem).

The pattern manual then includes two examples in the format of summary, context, pattern and problem. Following that is a discussion of each section of the format in detail. In the description of a pattern, the emphasis is on the relationship between parts, with invariant and variant parts specified.

> PATTERN
>
> Each pattern statement contains a number of <u>parts</u> and describes the spatial relations amongst these parts.
>
> Each part is a defined piece of space, identified by any number of characteristics ….
>
> The relation states the way these parts are to be arranged in space. Relations may include the size and shape of individual parts, as well as relationship between parts ….
>
> Every pattern contains at least one part, and at least one relation. [….]

> Every pattern defines a basic relationship between parts. In applying the pattern any variation is possible as long as the basic relationship holds. This means that the arrangement of parts in a specific building can vary a good deal, and still conform to a given pattern. In this sense, a pattern defines a whole family of possible variants. To define a pattern exactly, it must be clear just which features are essential, and just what variations are permissible. It will usually be helpful to show a single archetypal diagram which summarizes the invariant features, and make verbal statements describing the allowable variations. Drawings or photographs of a variety of different buildings, all of which conform to the pattern, also help to convey this idea.
>
> Patterns to not have to be stated in a numerically exact manner to make their invariances clear. Some ideas lend themselves to precise numerical statements, and some do not. (Alexander, Ishikawa, and Silverstein 1967, 11–13).

The spatial foundations for patterns speaks to the original domain for application of built environments. Invariant features as essential, and some variations are permitted to be labeled under the identity of that same pattern name, as opposed to another identity.

The pattern is placed within a context. The pattern cannot be independent of the context, its validity depends on the relationship between the pattern and context.

> CONTEXT
>
> The context is the spatial setting within which the pattern is valid.
>
> A context is very much like a pattern statement. It consists of one or more spatial parts and certain relationships among these parts. Each part may have a number of spatial characteristics associated with it. [….]
>
> Each statement describes a spatial setting within which a certain pattern is appropriate. [….]
>
> Clearly, the context may vary from short, common-sense statements to complex, analytically derived statements, many pages long. The essential point is this: The context must be a perfectly clear statement of exactly <u>where</u> the pattern is valid.
>
> Note that a pattern valid in one context may be quite wrong in a context only slightly different. […]
>
> Remember that every part in the context may have any number of aspatial characteristics associated with it. Furthermore, these aspatial modifiers may have any number of values associated with them. [….]
>
> Any time a context contains an aspatial modifier with an associated range of values, corresponding pattern variants should be given in the pattern statement (Alexander, Ishikawa, and Silverstein 1967, 14–16).

In considering context, systems thinking helps. If the pattern can be specified within a boundary, the context is the relevant features outside the boundary. Where features are or are not relevant could lead to sensemaking discussions.

The "problem" may better expressed as "problem statement". It has a sense of a current state where there is some dissatisfaction that might lead to an intervention towards some preferred alternative future state. However, it's also possible that the "problem" isn't worth fixing, or may even not be a real concern.

> PROBLEM
>
> The problems statement contains all the reasoning which lies behind the assertion "the stated pattern is valid in the stated context". If functions as a kind of string, tying together the context and the pattern. Although this seems to relegate the problem to a subsidiary position, in fact the problem is, from a human standpoint, the most important of the three components and may be many pages long.
>
> Let us examine the organization of the problem statement our house sign example. [....]
>
> In short, the problem exists because certain functional demands are not being met by the pattern currently governing the arrangement of house signs. The problem statement continues by isolating these functional demands: [....]
>
> Finally, the problem statement shows how the new pattern for house signs is derived from, and thus meets, all of the functional demands: [....]
>
> In a nutshell, this problem statement says, the existing house sign pattern creates a problem; this problem may be seen as a conflict between certain human demands under present conditions. The new pattern is derived from these demands and solves the problem. Every problem statement, no matter how it is internally organized, should exhibit this logic.
>
> There is always the chance that the problem stated is wrong. In fact, when we argue against a design idea it is almost always because we think the problem on which it is based is dubious, or we think important parts of the problem have been left out. In either case, it is clear that the rightness of the pattern hinges largely on whether the problem stated is correct.
>
> Like the house sign example, problem statements will generally contain at their core a number of functional demands. In the student room pattern, an example of such a demand is the statement "On occasion students want to leave and enter their rooms without feeling obligated to those living immediately around them". These demands have at one time or another been called requirements, needs, performance standards, facts, tendencies, objectives, constraints, activities, technical data, and so forth. Whatever they are called, these elements form the crux of the problem, and like hypotheses in science, they may be right or wrong. They may concern human behavior, economics, the state of technology, the political climate, whatever; no limits can be placed on the kinds of elements necessary describe a problem properly. It is only essential that the hypothetical nature of each element be made perfectly clear: No matter how intuitive or how "scientific", every element in every problem is a hypothesis potentially wrong.

A "problem statement" may be reframed as a "deficiency" or "dissatisfaction" if the emphasis is to be placed on the current state, rather than a future state design solution". While the problem statement seems as though it should be expressed as an objective state, the primacy of "human demands" leads to dealing with subjective judgements. What might be a problem for one person is not a problem for another. Establishing a "problem" therefore requires some craftsmanship in evolving to a understandable and reasonable "statement". In 1967, the term "forces" does not yet appear to have been introduced.

Let's try an alternative expression of the pattern format, to check our understanding. A pattern language could be expressed as a set of relations (e.g. patterns as parts of buildings and cities) in relation with a bounded variety of containing conditions (e.g. range of contexts, with parts in containing wholes) described as a dissatisfaction or deficiency in a current state (e.g. problem statement). In rigourous systems thinking, this can lead to questions about boundary critique and wicked problems (as problematiques, or systems of problems called messes).

3.4 Circa 1968, the feature of generativity was added to pattern language, evoking a systems appreciation

In an articulation of Alexander's expressions, the label "pattern" is not sufficient to describe the richness of vision. The label "pattern language" is broader, but still not complete. An extended label of "generative pattern language" can be aided by digging into systems thinking foundations. In the "Systems Generating Systems" article, an architectural theory is presented as four points:

> 1. There are two ideas hidden in the word system: the idea of a *system as a whole* and the idea of a *generating* system.
>
> 2. A *system as a whole* is not an object but a way of looking at an object. It focuses on some holistic property which can only be understood as a product of interaction among parts.
>
> 3. A *generating* system is not a view of a single thing. It is a kit of parts, with rules about the way these parts may be combined.
>
> 4. Almost every 'system as a whole' is generated by a 'generating system'. If we wish to make things which function as 'wholes' we shall have to invent generating systems to create them.
>
> In a properly functioning building, the building and the people in it together form a whole: a social, human whole. The building systems which have so far been created do not in this sense generate wholes at all (Alexander 1968).

This perspective puts the occupants in the building as part of the generating system. Architects who focus on only the built physical structure miss the whole to which Alexander speaks. The product of interactions between parts emerges new features and/or properties for the whole.

3.5 Circa 1968, pattern language in ranges of contexts was demonstrated with a variety of multi-service centers

An authentic generative pattern language is derived from practice, with reflective theorizing. At the advent of Center for Environmental Structure, *A Pattern Language Which Generates Multi-Service Centers* was a demonstration prototype created on an abductive grounding, since multi-service centers were a new idea in 1968.

> In this report, we present a prototype for multi-service center buildings.
>
> A multi-service center is a community facility, which provides a variety of special services to citizens. It is intended especially to help solve some of the problems of low income

> communities. Experimental multi-service centers have been started in many cities throughout the United States. however, there is not yet any agreement about the form which multi-service centers should take – either in their human organization, or in their special organization.
>
> Our report deals chiefly with the spatial organization; but since human and spatial organization cannot property be separated, many of the specifications given in this report, go deeply into question of human organization as well (Alexander, Ishikawa, and Silverstein 1968, 1).

The combination of human and spatial organization reflects a whole when or if the buildings were to be actually constructed. The human parts would include not only the citizens coming in for services, but also the public servants working there.

The challenge with a prototype is that there is a high degree of variability. The eight buildings generated by the pattern language would be at Hunts Point, San Francisco, Brooklyn, Bowery, Phoenix, Newark, and two in Harlem. The sites, constraints and clientele would be different in each case.

> We have not designed a prototype in quite the conventional sense, and must begin with a word of explanation about the nature and purpose of prototype buildings.
>
> A prototype is a generic scheme. It has not special site, no real client, no climate, no particular size. It is a kind of imaginary building, which is meant to convey certain essential ideas to designers of similar buildings. It is usually presented by means of loosely drawn schematic drawings, so that designers who are designing a building of this type, can mould it to fit whatever specific local conditions they are confronted with. It is meant to convey some essential, generic ideas, which can be applied many times over to special cases. It defines a family of buildings; and its meant to define this family of buildings in such a way that anyone who understands the prototype will be able to design specific members of this family.
>
> The ultimate purpose of a prototype design, then, is to provide guidelines which will generate a large number of specific buildings.
>
> Some will have many services, others will have fewer services. Some will be on main streets, others on side streets. Some will be in very dense neighbourhoods, others in neighbourhoods of lower density. Some will be multi-story, others will be single story. Some will be in warm climates, others in cold climates. No one prototype can do justice to this range of variation. A prototype would standardize the buildings, where standardization is inappropriate; it would tend to overlook the uniqueness of each special case.
>
> Our approach to prototype is intended to overcome this difficulty. <u>We have tried to reconcile the uniqueness of each community with the fact that certain organizational principles are valid from one community to another</u>.
>
> What we have devised, then, is a system of generating principles, which can be richly transformed according to local circumstances but which never fail to convey their essentials. This is rather like a grammar. English grammar is a set of generating principles which general all the possible sentences of English. It would be preposterous to suppose

> that one could convey the full richness of the English language by means of a few well chosen "prototypical" sentences.
>
> What we have devised, then, is a system of generating principles, which can be richly transformed according to local circumstances but which never fail to convey their essentials. This is rather like a grammar. English grammar is a set of generating principles which general all the possible sentences of English. It would be preposterous to suppose that one could convey the full richness of the English language by means of a few well chosen "prototypical" sentences (Alexander, Ishikawa, and Silverstein 1968, 1–2).

This introduction eases the reader into a "pattern language" with the alternative phrase of "generating principles. In the second chapter, an concise definition is provided.

> If we examine the patterns as they are presented in full, in the Appendix, we shall see that each pattern has two parts: the PATTERN statement itself, and a PROBLEM statement. The PATTERN statement is itself broken down into two further parts, an IF part, and a THEN part. In full the statement of each pattern reads like this:
>
> IF:X THEN:Z / PROBLEM:Y
>
> X defines a set of conditions. Y defines some problem which is always liable to occur under the conditions Z. Z defines some abstract spatial relation which needs to be present under the conditions X, in order to solve the problem Y.
>
> In short, IF the conditions X occur, THEN we should do Z, in order to solve the PROBLEM Y (Alexander, Ishikawa, and Silverstein 1968, 17).

This definition is a change from the published a year earlier in 1967. Remaining consistent is "problem Y" which was previously expressed as a problem. However, now "condition X" is used instead of "context" and "abstract spatial relation Z" is used instead of "solution". In addition, "Z … needs to be present under conditions X", so there must be a relation always associated. This change in language could be a softening away for from the language of a "solution", since the variability in the prototypes would mean variability in the solutions.

Stepping back to the first chapter, summaries of 64 patterns were included. To give a feel for them, here's the first six.

> Each pattern prescribes some feature of a multi-service center building. It describes a relationship which is required to solve a problem which will occur in that building. The summary does not describe this problem; it describes only the pattern. […]
>
> 1. Small Target Areas: The multi-service center services a target area with population of 34,000 ± 20%.
>
> 2. Location (1968): Service centers are located within two blocks of a major intersection.
>
> 3. Size Based on Population: The total size of an MSC which services a target area of population N, is .9N square feet.
>
> 4. Community Territory: The service center is divided into two zones, services and community territory; community territory includes space for community projects and a public area.

5. Small Services without Red Tape: No one service has a staff size greater than 12; each service is physically cohesive and autonomous; the services are loosely organized with respect to each other.

6. Expansion: The number of services can grow and the size of any one service can grow; but the relationship of all services to community territory does not change.

7. Entrance Locations: The building's main entrances are immediately visible to a person approaching, by foot or by car, from any direction.

8. Parking (1968): Either parking is provided for everyone [this will require .5N square feet for a target population of N], or there is emergency parking only; staff-only parking is never provided (Alexander, Ishikawa, and Silverstein 1968, 5–6).

In the original publication, small iconic drawings were included beside each pattern in the first chapter.

In the third chapter, the icons were drawn as a network, tracing relations between one pattern and another. For greater clarity, the network can be reproduced with text replacing the icon. In Illustration 15, the top of a pattern language network cascade is depicted. A few nodes and lines are coloured to make tracing interconnections easier, but otherwise don't have a meaning. The format was described:

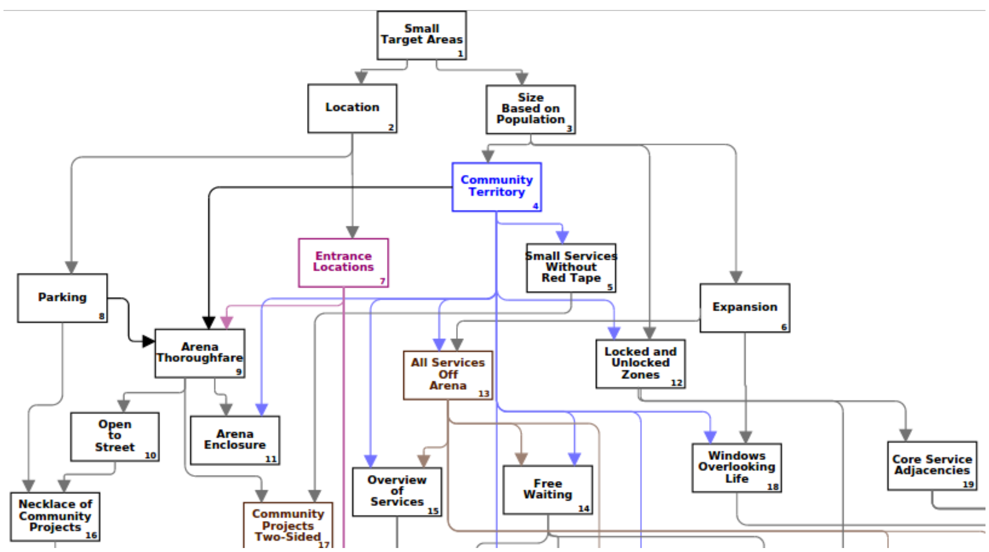


Illustration 15: A pattern language cascade (excerpt)

In each example we describe a hypothetical community, which needs a multi-service center. We show a design for a multi-service center building, appropriate for that community, which has been generated by the language. And we show, step by step, how the language helped generate this design.

For each example, the steps are presented in sequence (A, B, C, D, ….). Each step introduces new patterns into the design. At every step we mention the new patterns which have come into play and their interaction with local condition, in words; we show the form of the building, as it has been formed up to that step, diagrammatically; and we show a miniature drawing of the language cascade so that we can see which part of the cascade is responsible for this step, and where the part sits in the cascade as a whole.

[One point must be heavily underlined. Although the evolution of these designs is presented in a step-wise sequential manner, this is merely for convenience of presentation. it does not imply that the design process generated by the language, is, any way but the most general sense, itself sequential] (Alexander, Ishikawa, and Silverstein 1968, 19).

The first building generated by the pattern language was Hunts Point. The summary of the building was: "40,000 people -- Strong community corporation -- Large block worker program -- 9 to 12 services -- Site open to three sides -- Near major intersection and transit station". Here is the first part of a pattern language cascade specific to Hunts Point:

A

This multi-service center is to service 40,000 people. According to Pattern 1 Small Target Areas, this population is too large, but for political reasons, the decision stands and is irrevocable.

First a triangle site was selected, right on a major intersection (Pattern 2: Location (1968)). However, other requirements made it clear that this site was too small (Pattern 3 Size Based on Population (1968)), and a larger, rectangular site was chosen, one-half block from the original site (thus still conforming to Pattern 2 Location (1968)).

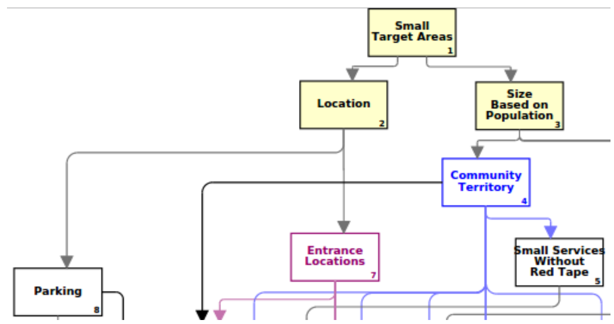


Illustration 16: Cascade A for Hunts Point

On this site there was room only for emergency parking, and so Pattern 8 (Parking) does not play a major role. Nor does 5 Small Services without Red Tape, which had not been formulated prior to the Hunts Point Design.

B

Pattern 16 (Necklace) calls for provisions for community projects around the "live" edge of the building; hence we confine services to the "dead" edge of this building, against other buildings.

C

Climate considerations made it clear that the arena could not be open (11: Arena Enclosure), and so it was developed as an interior street. Orientation of this "street" is given by local conditions in accordance with Pattern 7 (Entrance Locations).

D

The size of the arena and its relationship to waiting and services is established by Patterns 13 (All Services Off Arena), 14 (Free Waiting) and 15 (Overview of Services); and the arena is shaped accordingly.

[.....]

G

Finally, "pockets" in the arena are shaped and filled according to Patterns 29 (Activity Pockets), 35 (Information-Conversation), 43 (Waiting Diversions), and 42 (Sleeping Ok) (Alexander, Ishikawa, and Silverstein 1968, 22).

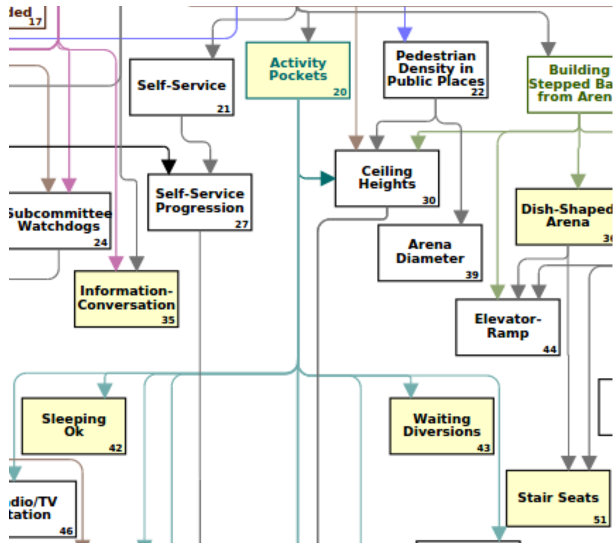


Illustration 17: Cascade G for Hunts Point

The second building generated by the pattern language was San Francisco. This was summarized as "Combination service and recreation center -- Mild climate -- Outdoor arena -- Strong community organization -- Corner site -- Off site parking provided". The pattern language cascade for San Francisco starts off differently.

A

To make the recreation part of the building highly accessible, the whole ground floor is devoted to recreation activity -- this area will be open late, according to Pattern 12 (Locked and Unlocked Zones); also it is highly visible from the street (10 Open to Street), and provides a thoroughfare (Pattern 9 Arena Thoroughfare). In this climate, the arena, which can be open to the sky (11 Arena Enclosure) takes on an unusual character -- it becomes a park. The whole ground floor becomes community territory (4 Community Territory).

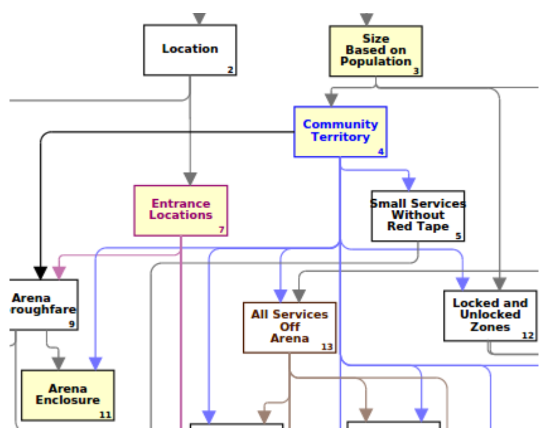


Illustration 18: Pattern language cascade A for San Francisco

B

The recreation ware, which will become the hang-out for many members of the community, gives the building a natural base for community organization. It is therefore essential to

> put information, and community organizers and community projects at ground level. Patterns 17 (Community Projects Two-Sided), 28 The Intake Process), 35 (Information-Conversation) and 16 (Necklace of Community Projects) put them into the positions shown.
>
> [….]
>
> **E**
>
> To get windows overlooking life (18 Windows Overlooking Life), there are holes from the second and third story, looking down into the recreation floor (Alexander, Ishikawa, and Silverstein 1968, 26).

Brooklyn was summarized as "12,000 persons -- Expansion key issue -- Steep site -- Parking must be provided -- Laundromat and news stand on site to be saved". Bowery was summarized as "20,000 persons -- Service primarily for elderly -- Site surrounded by old tenements on three sides -- Center to serve hot meals daily". Each of Phoenix, Newark, Harlem 1 and Harlem 2 was described uniquely.

This 1968 publication demonstrated how a pattern language for a multi-service center buildings would generate something different for each of the hypothetical sites with different conditions and contexts. Future work would be based on learning in practice, rather than hypothesizing.

Almost a decade would pass before the progress would be reported in three books. The first volume was 1979 *The Timeless Way of Building* that "describes a theory of planning and building which is, essentially a modern post-industrial version of the age-old pre-industrial and traditional processes which shaped the world's most beautiful towns and buildings for thousands of years" (Alexander 1979). The second volume was the 1977 *A Pattern Language* "explicit set of instructions for designing and building, which defines patterns at every scale, from the structure of a region to the nailing of a window; set out in such a way that laymen can use it to design a satisfying and ecologically appropriate environment for themselves and their activities" (Alexander, Ishikawa, and Silverstein 1977). *The Oregon Experiment* was the third volume in the series, with "the master plan for the University of Oregon, and … a practical way of implementing these ideas in a community" (Alexander et al. 1975).

3.6 By 2001-2004, examples of living structure and processes of creating them were illustrated and theorized

The 1977 *A Pattern Language* established a grammar for architectural programming, but was weak on generativity. The 1979 *Timeless Way of Building* described construction processes, but was weak in clarifying the "quality without a name". In the 1987 *A New Theory of Urban Design*, some of the work yet to be done was described.

> During the period of 1976-1978 one the authors (CA), had become aware a deeper level of structure lying "behind" the patterns. At this level of structure, it was possible to define a small number of geometric properties which seemed to be responsible for wholeness in space. Even more remarkable, it was possible to define a single process, loosely then called "the centering process," which was capable of producing this wholeness (with its fifteen or so geometric properties) at any scale at all, irrespective of the particular functional order required by the particularities of a given scale. [….]
>
> … we began to imagine a process of urban growth, or urban design, that would create wholeness in the city, almost spontaneously, from the actions of the members of the

> community … provided that every decision, at every instant, was guided by the centering process (Alexander et al. 1987, 4–5).

For built urban environments, seven detailed rules for growth were prescribed:

1. Piecemeal growth: Growth should occur incrementally.
2. The growth of larger wholes: Each increment of growth should help form larger centers.
3. Visions: Proposed growth must be experienced and expressed as a vision.
4. Positive urban space: Buildings must create coherent adjacent public space.
5. Layout of large buildings: The layout of a building should be coherent with with building's overall position.
6. Construction: The structure of every building must generate smaller wholes within itself.
7. Formation of centers: Every whole must be a center in itself and must also provide a system of centers around it.

These would rules would be fleshed out and expanded in the publication of the 2001-2004 four volume *The Nature of Order*. In 2007, reflections on "Empirical Findings from The Nature of Order" provided a self-critical assessment of the logic set – 1 to 15 in Book 1, The Phenomenon of Life; 14-24 in Book 2, The Process of Creating Life; 25 to 36 in Book 3, A Vision of a Living World; and 37-59 in Book 4, The Luminous Ground – with evidence as demonstrated or (strongly) indicated (Alexander 2007).

3.7 Circa 2012, the potential for unfolding wholeness through local adaptation was presented as an alternative to the dominant systems of efficiency and control

Published in 2012, *The Battle for Live and Beauty of the Earth* describes the development of Eishin campus circa 1985. With Christopher Alexander now elderly, this could be the last book he publishes. As a reflective work, the writing integrates the theory developed over 50 years with a history of the activities and choices encountered during the project. The most illuminating content comes in Chapter 11, "Flags: The Reality of the Land". Alexander writes "this thirty-eight page chapter 11 will create an unforgettable and archetypal form in the site, creating the organization of the campus itself" (Alexander 2012, 163).

> **Section 1: Site Layout In System-A: The Joy of Laying out the Site Plan on the Ground**
>
> The essence of site layout in System-A, and the way in which it fundamentally differs from making a site plan in a planning office, lies in the fact that one physically draws the site plan out of from configurations that may be seen because they are discernible *in the land*. Thus the site plan is not an abstractly conceived, or designed, or invented figure, but a figure pulled out from the features of the land itself. [....]
>
> In system-A, it is always the wholeness of the place that matters. To intensify the wholeness of any place -- whether it consists of existing buildings in a town or or virgin land that is largely unbuilt -- proposed construction and buildings decided, and that means "felt" and though through on the site itself. It is not possible to do it any other way, since the relationship which exists between the buildings and the world around them are complex and subtle.
>
> On a drawing or a plan, one simply does not see enough (Alexander 2012, 163–164).

The challenge with constructing a building from abstract concepts is that choices and alternatives are difficult to appreciate. Text and drawings are a poor surrogate for a process that tries to make a new built environment both physically and emotionally real.

A first draft of a pattern language was created over a few weeks. Firstly, school leaders were interviewed. Refinements were developed with a larger group of teachers and teachers, leading to 110 essential patterns. This was reviewed with the building committee, and then presented to the whole school. With the pattern language completed, an estimated project cost was projected. Trimming of the land and indoor space was done with an average percentage reduction, and then the faculty was asked to reallocate the spaces, increasing space in some places only at the expense of decreasing somewhere else. The next step on the site plan was to create a site plan. This first involved walking on the land, and then beginning the layout.

> **Beginning the Layout of the Site Plan**
>
> [...] Knowing the overall configuration of the land, we had already been thinking about the way the pattern language might generate a layout on that particular piece of land, given the direction of access, orientation, wind, views, slopes and so on. As the content of the pattern language became clear, we were trying to understand the site, and trying to imagine the global structure of a possible campus layout that would arise naturally from the structure of the land (Alexander 2012, 167).

The trimmed land and indoor space were generated from the pattern language within budget constraints. However, finalization of the pattern language did not yet constitute an architectural program. Patterns could be laid out on the land in a variety of ways, some with a greater unfolding wholeness than others. The many detailed features of the land are better appreciated in reality, rather than lost through the abstraction of model-making.

The pattern language has a system of centers, and the land has a system of centers. The challenge is to find a program that integrates the quality of both, together.

> **Section 2. Finding the Two Fundamental Systems of Centers**
>
> To make the creation process clear, it is first necessary to decide, *in general*, what it is that has to be done when the site plan is make with a pattern language. In any building project, before the site plan can be created, we must identify two difference systems of centers.
>
> (1) There is a system of centers which is defined by the pattern language. Pattern-language centers define the major entities which are going to become the building blocks of the new project. In our case, the case of the Eishin project, the language defined the main building blocks or centers from which the new school and university were going to be made. The included, for instance, the entrance gate, the entrance street, the Tanoji Center, the homebase street, the main square, the back streets, Judo Hall, and many others. [p. 168]
>
> (2) Secondly, we had the system of centers which existed in the land. This system was created by the land forms, the slopes and ridges, by the roads, by direction of access, by natural low spots, natural high spots, and by existing trees and existing buildings.
>
> **It must be emphasized that these two systems of centers already existed at the time one started walking out the site plan**.

The first system consists of **patterns** created notions or entities that exist in people's minds). These patterns exist in a loose and undeveloped form in people's minds, even if they have not explicitly built a pattern language. When the pattern language *is* explicitly defined, it is more clear and makes a more powerful system which will get better results, especially because it comes from the feelings of people themselves. See patterns on pages 131-152.

The second system exists in the form of **places** on the site, discernible places that can be seen and felt on the site, if you have sufficient sympathy with the land. You can make this system explicit, by making a map of the centers, and paying attention to their structure. Each of these two systems is real. Together they provide the raw material from which the community is going to be made (Alexander 2012, 168–169).

The land has a system of places that make a pacing layer slower than the campus system of buildings. If the land was to be reshaped, that work would have to be done before buildings were to be constructed on it. The system of places constrains the system of buildings,and yet could support a greater wholeness if the features of both integrated well.

The most important centers given by the pattern language were listed as (i) the entrance street; (ii) the Tanoji Center; (iii) Five College Buildings; (iv) The Homebase Street; (v) Individual Classroom Buildings; (vi) The Great Hall and Main Square; (vii) The Library and Research Center.

<< See Diagram 1, Alexander 2012, 170. >>

Diagram 1: Seven most important centers in the pattern language, which together give a broad conceptual picture of a possible layout that the centers can have. Not to scale.

The most dominant and strongest centers which existed as "natural places" in the land were listed as (i) Natural Entrance Position; (ii) The Ridge; (iii) The Swamp; (iv) A Natural Place for Large Buildings; (v) Minor Entrance Position; and (vi and vii) East and West Ends of the Ridge.

<< See Diagram 2, Alexander 2012, 171. >>

Diagram 2: Seven most important NATURAL centers in the land, which together can lead to a basic possible layout that centers can have, in their LOCATIONS in the land

The challenge in architectural programming then became finding an arrangement between the system of centers in the pattern language that preserved their relations to one another, and also coincided with the arrangement of key existing centers in the land.

Combining the Two Systems of Centers

What has to be done in creating a site plan for a community of an institution, is to bring these two systems of centers together. We have to hunt for a single configuration which springs from both centers, and integrates the qualities of both. We must find a way in which the system of centers defined by the pattern language can be placed, so that it enhances, preserves, and extends, the system of centers which is already in the land. It is a kind of healing process, which uses the new centers given by the pattern language, to heal the configuration of the old centers -- those that exist in the land.

In some case this is very hard to do because the two systems of centers may not coincide on all points. That is why it takes serious intellectual and emotional effort. In many

> architectural projects, this is the single most difficult phase of the work. The Eishin Campus was no exception. Including the time taken during the work on the pattern language, it took from May 1982 to January 1983, about nine months of continuous effort, to get the site plan right. When it was finally done, the site plan was a *discovery*, a real achievement, which came from constant study and experience of the site itself (Alexander 2012, 173).

The land could be walked. Arranging the system of buildings was eventually worked out by the use of scale models that could later be validated at full scale.

> **Using the Small Model at Berkeley**
>
> In order to make it possible to think about the problem of the overall plan form, while away from Japan, we made a series of accurate topographic models of the site. We have a large one in our office in Japan, at the metric scale of 1:100. And we had two in our Berkeley offices in California -- one made at a scale of 1:200, the other was made at 1:500. The last was very small, and therefore very helpful, because it allowed us to judge the configuration *as a whole*. Larger models show details very nicely, but you lose the drift of the gestalt, as it sits on the land, and reflects the land.
>
> In order to use these models, we recorded on them the seven most important facts about the land, which we had identified during our many visits to the site. [….]
>
> These facts seemed irreconcilable with the key patterns because there seemed to be no natural way of arranging the college precinct *and* the homebase street (as we had them in the pattern language) in the fashion consistent with these seven "facts" about the land. Finally, though, after all our efforts in Japan and in Berkeley, and after all the work on the site by everyone, and so many months of frustration, the problem did get solved.
>
> [...] a new point emerged. The fact that the homebase street would be more powerful as an *approach* to the Tanoji Center, than as something *hanging off* it. This was hard to see, at first, because it implied reversing the main sequence of the pattern language. But when we tried it, it was clear that the sequence almost instantaneously "jelled" with the land configuration. After playing with it more, we confirmed that it was indeed much better. The sequence of the pattern-language elements which we had taken as fixed, was suddenly reversed.

Instead of this:	We now had:
1) Entrance Street	1) Entrance Street
2) Main Square	2) Main Square
3) Tanoji Center (College)	3) Home Base Street (High School)
4) Home Base Street (High School)	4) Tanoji Center (College)

> The reorganization seems almost minor. but it dramatically affected the situation (Alexander 2012, 176–178).

The scale models enabled the members of the architectural team who had developed the pattern language and walked the land to collectively assess alternative approaches.

After the test plan diagrams and the balsa wood model of the site, the architectural team moved onto the Eishin site to perceive the design at full scale.

> **Section 3. Flags v. System-B and Mr. Miura**
>
> […] We had already made it clear that nearly all our work on the site plane was done on the site itself. Whatever we did on models, we used the models as if they were the site itself – and relied on feelings that we could feel in the model, imagining that it was the site itself. This was made necessary by the huge distance between California and Japan.
>
> As one works on a site and the plan gradually emerges, it is necessary, of course, to leave marks – sticks, stones, marketers of various kinds – to fix the position of the different things which have been decided. On the Eishin site, this was more important than usual, because the site was covered in tea bushes. These bushes were three to four feet high (deep). In a few places there were mulberry bushes which were even higher. A marker therefore had to be about six feet high, even to be seen at all.
>
> So we used six-foot-long bamboos. But even they could not be seen at a distance among the tea bushes. To see what was happening – to grasp the evolving site plan – one had to be able to identify key points from distances of several hundred meters. We therefore tied different colored ribbons and cloths – white yellow, blue, red – to the ends of our long bamboos. These were our markers – our *flags* (Alexander 2012, 180).

Unfortunately, with real estate negotiations underway, the real estate broker, Mr. Miura, went one night and removed every one of the two hundred flags. This led to a boiling point, and several days of angry talks.

> Only gradually was the *absolute necessity* of using the flags, now established as a fact. The compromise solution in which some markers, without flags, could be left, and in which we would work as fast as possible, were all threshed through again (Alexander 2012, 184).

In compromise, the flags that were not too big were used and then taken down at night so that the work could proceed. This led to the first hardline drawings derived from flag positions amongst the tea bush rows, calibrated against an aerial photo.

In programming the Eishin campus, Alexander's reflections of the team work practices underscore importance of assessment made in the real context of the project, rather than abstract drawings or textual descriptions. Arranging for unfolding wholeness is an activity best done in reality rather than in abstraction.

3.8 While most architects follow a teleological design process, Alexander's aims for ateleological

A system can be architecturally programmed to learn to a greater or lesser extent. Learning is a response to changes in the environment. A system that is designed for a point in time (e.g. the date of occupancy, or a release date) may resist later efforts for improvement. A system design to learn over time will generally meet essential initial wants and needs, and then enables adaptation in resource to emergent wants and needs following a long period of engagement. The capacity to learn can be associated with the degree of teleology or ateleology.

Systems which must adapt in a meaningful and holistic way must be able to learn. Teleological systems have a limited capacity to learn. This limited capacity is brought about by two factors. First, as systems become increasingly teleological, their set of alternative actions become progressively less. In these systems the only acceptable actions are the actions that make the systems behaviour converge towards the selected goal. This limited set of legitimate actions limits the system's ability to experiment, as behaviour that does not directly contribute to the converging behaviour is inefficient and ineffective. The lack of ability to experiment causes the system to lose its capability to expand its scope of actions – which limits its capability to learn. Teleological behaviour therefore occurs to a greater or lesser degree at the expense of learning.

Furthermore, to learn, the system must be able to appropriate the tacit information that is part of its continual interaction with its environment, for it is through appropriation that new understanding is constituted.

- The system must become part of a hermeneutic circle. The system must therefore be able to continually interpret and understand itself in terms of the whole. This implies that the system (as part) must understand its actions or behaviour in terms of its meaningfulness in relation to the whole. To lose this coherence would imply the loss of identity and the wisdom of the whole. Such continual reinterpretation would be seen as inefficient and ineffective forms of teleological behaviour.
- The system must remain open to the possibilities of new understanding. Remaining open to new understanding is to be distracted from, or lose sight of, the goals or objectives that are essential for teleological behaviour.

From a systems theory perspective Bateson (1980), using the work of Ashby (1957), showed that a system cannot learn (and thus evolve) unless it is stochastic. Bateson defines stochastic systems as systems that incorporate at least two processes. First, the system must have a random process – a process that can generate diversity. Second, the systems must have a built-in comparator that selects certain events, states, or alternatives based on some type of criteria. The determination of the criteria is critical as inappropriate criteria could force the system into short-term teleological behaviour. In the process of evolution the comparator is "natural selection". Stochastic systems are, however, divergent. Divergent systems' behaviour cannot be predicted, and not being predictable means they cannot be "controlled".

It should be clear, from the above, that learning and control are negatively related. Returning to the process of development it can be concluded that:

- The teleological design process (as a convergent process) is very predictable and thus controllable. The process does not, however, have the ability to evolve as it is not able to learn.
- To create an evolving development process a "stochastic" ability is needed. An ateleological design process, therefore, is the only way to create a dynamic and learning process between the designers, the users and the information system (Introna 1996, 24–25).

While the focus of this research on teleological and ateleological design processes was created in the context of information systems development, the line of reasoning appears to be valid for other types of systems.

The perspective of teleological and atelelogical design processes can be shifted to a broader application in systems development.

> **Ateleological systems "development"**
>
> *The ateleological concept*
>
> We should agree that a teleological approach to systems development seems not to provide the answers we so desperately need, what is the alternative? It may seem as if we have destroyed all basis for meaningful behaviour. [p. 25]
>
> It may be useful to try to contrast ateleological behaviour with teleological behaviour, before attempting to outline the alternative. A sense of the difference between teleological and ateleological systems development can be gleaned from the "attributes" of the processes as expressed in [Table 3].

Table 3: Teleological and ateleological development

Attributes of the design process	Development philosophies	
	Teleological development	Ateleological development
Ultimate purpose	Goal/purpose	Wholeness/harmony
Intermediate goals	Effectiveness/efficiency	Equilibrium/homeostasis
Design focus	Ends/result	Means/process
Designers	Explicit designer	Member/part
Design scope	Part	Whole
Design process	Creative problem solving	Local adaptation, reflection and learning
Design problems	Complexity and conflict	Time
Design management	Centralized	Decentralized
Design control	Direct intervention in line with a master plan	Indirect via rules and regulators

> Do examples of ateleological systems development exist? Yes, there are various examples in different fields of social interaction and development that can be distinguished in this manner.
>
> It must again be noted, before discussing these examples, that the distinction as made above is a specific way of using a language and that there is no claim here that this is the only way of distinguishing these two spheres of thinking. This is, nonetheless, a start in the articulation of what are two fundamentally different ways of viewing the world in general and systems development in particular (Introna 1996, 25–26).

Commercial systems have typically been developed under the premise of teleological development, with clear goals and centralized management. Open source systems have typically been developed under the premise of working collaboratively towards a whole (or multiple wholes), with decentralized management. This does not mean that one style would not work in the alternate context, although adjustments in work practices would be required.

Alexander is cited as a primary exponent of ateleological design, while the more popular style employed by architects of built environments is teleological.

> An appropriate example is found in architecture and urban design. Houses and cities are also examples of socio-technical systems, and in that sense they, the theories of their design and development, could give us a new understanding of the development of information systems as socio-technical systems. The use of architecture as a reference theoretical framework has been argued by Kling (1984) and more recently by Lee (1991).
>
> *Architecture and ateleological design*
>
> It must, however, be said that the mainstream thinking and philosophy in architecture and urban design are rationalistic and teleological. The exception to this rule is the thinking as embodied in the work of Alexander (1979). Alexander describes a design process that seeks to design buildings and cities that are alive, beautiful, and whole – those qualities that make people to want to dwell in them. To try to do justice to his theory is beyond the scope of this paper. Some of the important concepts will, however, be outlined. His theory is rich and subtle, and can best be understood by reading his work, especially *The Timeless Way of Building*.
>
> Alexander's first, and very important principle, is that the design process must be self-generative: "It is a process which brings order out of nothing but ourselves; it cannot be attained, but it will happen of its own accord, if we will only let it". The design process is not controlled by a "designer" – in this case the architect. The process must be in the hands of the people. It enables them to "design" that, which is meaningful for them. The adaptation between the people and the buildings is profound. [....]
>
> According to Alexander, the second important principle is that the development process should be piecemeal. There should be no big jumps. Each increment must contribute to the whole. It must make it more whole and more alive. This piecemeal process is implemented by a pattern language. It is a "language" because it provides a set of dynamically evolving patterns that are used to express – in physical space (buildings, cities) – the human events of the people using the language.
>
> What is a pattern? Patterns are fundamental geometric structures or relationships that, if applied, will generate wholeness. Patterns are expressed in terms of a rule "which establishes a relationship between a context, a system of forces which arise in that context, and a configuration which allows these forces to resolve themselves in that context". Patterns are, however, not a fixed set of rigid relationships, but are "a field – not fixed, but a bundle of relationships, capable of being different every time that it occurs, yet deep enough to bestow life whenever it occurs" (Introna 1996, 26–27).

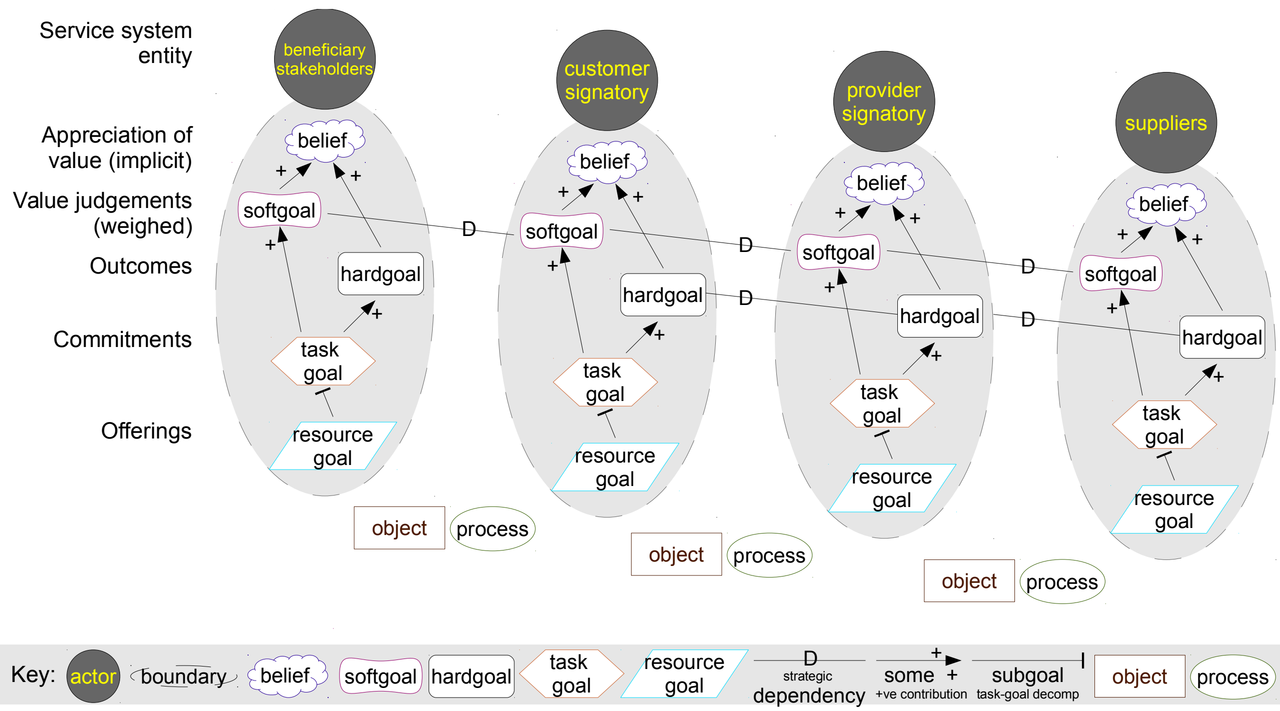
This recognition of the application of pattern language into architectural programming as ateleological brings the ideas around full circle. The articulation of practices that Alexander sought to clarify for others over a 50-year career may have been better expressed in the domain of information systems development. The opportunity to extend this thinking in service systems thinking is open.

4. POSSIBILITIES: THE EMERGING SERVICE SYSTEMS THINKING COMMUNITY HAS SOME QUESTIONS TO CONSIDER

Conversations for possibilities may be expressed as questions that may lead to alternative findings and convergent or divergent directions.

4.1 Can a diagrammatic notation style make representations easier for the service systems thinking community?

iStar and OPM are complementary languages worth considering.



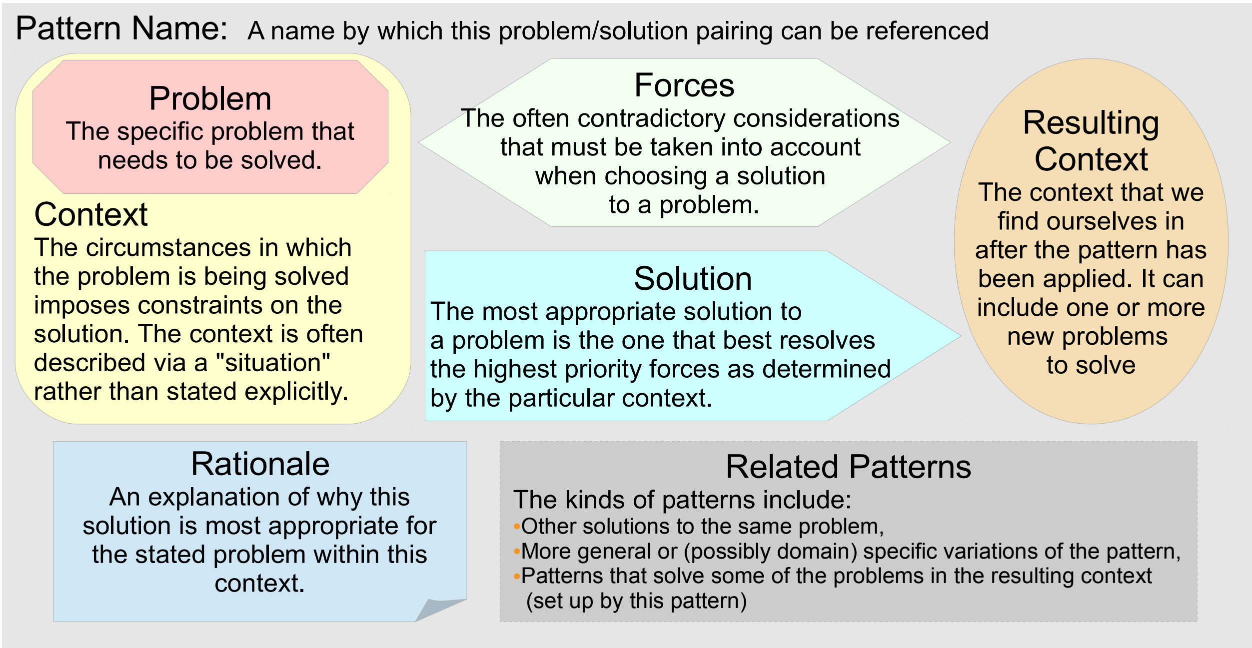
4.2 Should a generative pattern language for service systems thinking be retargeted towards architectural programming (i.e. problem seeking) rather than design (i.e problem solving)?

Sponsors and beneficiaries explicitly recognized as whom service systems should serve? Value implicitly as interactive value in softgoals?

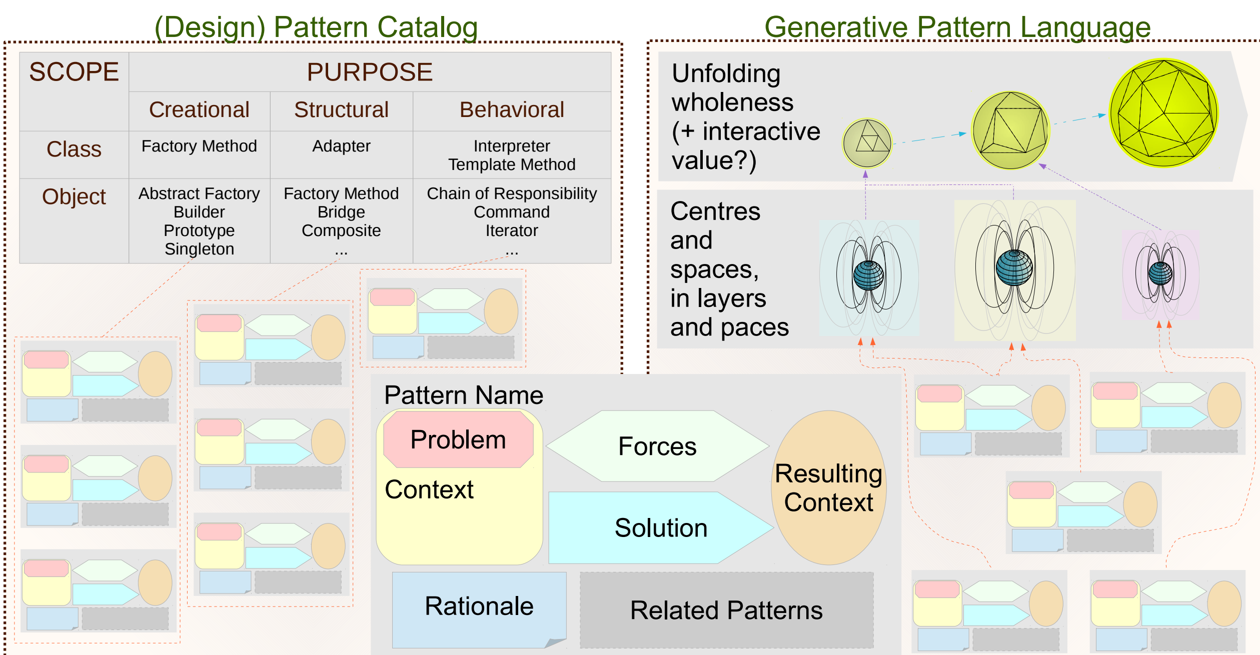
4.3 Can structural and processual viewpoints be simultaneously addressed in interactivity?

Through practice, as a pattern form for pattern writing has emerged (Meszaros and Doble 1997).

> Here is a short and necessarily incomplete definition of a pattern:
>
> A recurring structural configuration that solves a problem in a context, contributing to the wholeness of some whole, or system, that reflects some aesthetic or cultural value (Coplien and Harrison 2004).



End the separation of structural patterns from process (behavioral) patterns, unified with systems and subsystems?



4.4 Is there a contribution that Open Systems Theory make towards architectural programming?

What is systems thinking?

Systems thinking is a perspective on parts, wholes and their relations (Ing 2013).

Boundary judgement (Ulrich and Reynolds 2010).
Design Principle 1 (redundancy of parts); Design Principle 2 (redundancy of function built into parts); Design Principle 3 (redundancy of potentialities for broader future contexts) (Selsky, Ramírez, and Babüroğlu 2013).

5. FUTURE ACTION: CONTINUING THE CONVERSATIONS

At the beginning of the journey, Service Systems Thinking has been endorsed the some leading professional organizations. Preliminary outlines have been endorsed by:

- the International Society for the Systems Sciences (ISSS);
- the International Council on Systems Engineering (INCOSE); and
- the International Society for Service Innovation Professionals (ISSIP).

Meetings in which participation has be encouraged include:

- June 2014, Las Vegas – the International Symposium of the International Council on Systems Engineering (INCOSE);
- July 2014, Krakow, Poland, at the Human Side of Service Engineering Meeting; and
- July 2014, at the annual meeting of the International Society for the Systems Sciences (ISSS);

PloP 2013 is a place where advances in the development of pattern languages can be discussed.

REFERENCES

Ackoff, Russell L., and Fred E. Emery. 1972. *On Purposeful Systems*. Aldine-Atherton.

Alexander, Christopher. 1966. “A City Is Not a Tree.” *Design*, Council of Industrial Design, , no. 206.

———. 1968. “Systems Generating Systems.” *Architectural Digest* 38. http://books.google.ca/books?id=Ib4yEJErR5EC&pg=PA58.

———. 1979. *The Timeless Way of Building*. Oxford University Press. http://books.google.ca/books?id=H6CE9hlbO8sC.

———. 2007. “Empirical Findings from The Nature of Order.” *Environmental & Architectural Phenomenology* 18 (1): 11–19. http://www.arch.ksu.edu/seamon/Alexander_Nature%20of%20Order.htm.

———. 2012. *The Battle for the Life and Beauty of the Earth: A Struggle Between Two World-Systems*. Oxford University Press. http://books.google.com/books?id=xAMEywAACAAJ.

Alexander, Christopher, Sara Ishikawa, and Murray Silverstein. 1967. *Pattern Manual*. Berkeley, California: Center for Environmental Structure. http://books.google.com/books?id=_VYpGwAACAAJ.

———. 1968. *A Pattern Language Which Generates Multi-Service Centers*. Center for Environmental Structure. http://books.google.ca/books?id=FGdPAAAAMAAJ.

———. 1977. *A Pattern Language: Towns, Building, Construction*. New York: Oxford University Press US. http://books.google.ca/books?id=hwAHmktpk5IC.

Alexander, Christopher, Hajo Neis, Artemis Anninou, and Ingrid King. 1987. *A New Theory of Urban Design*. Oxford University Press. http://books.google.com/books?id=STCoRQzgDZcC.

Alexander, Christopher, Murray Silverstein, Schlomo Angel, Sara Ishikawa, and Denny Abrams. 1975. *The Oregon Experiment*. Oxford University Press. http://books.google.ca/books?id=u2NSI4vSu_IC.

Brand, Stewart. 1994. *How Buildings Learn: What Happens after They're Built*. New York: Viking. http://books.google.com/books?id=zkgRgdVN2GIC.
———. 1999. *The Clock of the Long Now: Time and Responsibility*. Basic Books.
Brand, Stewart, and James Runice. 1997. *Shearing Layers*. Vol. 6. 6 vols. How Buildings Learn. BBC. http://video.google.com/videoplay?docid=2283224496826631552.
Coplien, James O., and Neil B. Harrison. 2004. *Organizational Patterns of Agile Software Development*. Prentice-Hall, Inc. http://books.google.ca/books?id=6K5QAAAAMAAJ.
Cunningham, Ward. 2012. "Smallest Federated Wiki." *Federated Wiki*. http://ward.fed.wiki.org/view/welcome-visitors/view/smallest-federated-wiki.
Dori, Dov. 2006. "Modeling Knowledge with Graphics and Text Using Object-Process Methodology." In *Encyclopedia of Knowledge Management*, 683–93. Hershey, PA: Idea Group. http://esml.iem.technion.ac.il/wp-content/uploads/2011/08/Object-Process-Methodology.pdf.
———. 2011. "Object-Process Methodology for Structure-Behavior Co-Design." In *Handbook of Conceptual Modeling*, edited by David W. Embley and Bernhard Thalheim, 209–58. 10.1007/978-3-642-15865-0_7: Springer Berlin Heidelberg. http://esml.iem.technion.ac.il/wp-content/uploads/2011/08/opm.pdf.
Horkoff, Jennifer, and Eric Yu. 2006. "Basic I* Notation." *iStar Quick Guide*. http://istarwiki.org/tiki-index.php?page=iStarQuickGuide#Basic_i_Notation.
IBM. 2010. *The World's 4 Trillion Dollar Challenge: Using a System-of-Systems Approach to Build a Smarter Planet*. GBE03278-USEN-02. Institute for Business Value. http://www-935.ibm.com/services/us/gbs/bus/html/ibv-smarter-planet-system-of-systems.html.
IfM, and IBM. 2008. *Succeeding through Service Innovation: A Service Perspective for Education, Research, Business and Government*. Cambridge, UK: University of Cambridge Institute for Manufacturing. http://www.ifm.eng.cam.ac.uk/ssme/.
Ing, David. 2013. "Rethinking Systems Thinking: Learning and Coevolving with the World." *Systems Research and Behavioral Science* 30 (5): 527–47. doi:10.1002/sres.2229.
Introna, Lucas D. 1996. "Notes on Ateleological Information Systems Development." *Information Technology & People* 9 (4): 20–39. doi:10.1108/09593849610153412.
Lessard, Lysanne, and Eric Yu. 2013. "Service Systems Design: An Intentional Agent Perspective." *Human Factors and Ergonomics in Manufacturing & Service Industries* 23 (1): 68–75. doi:10.1002/hfm.20513.
Lusch, Robert F., and Stephen L. Vargo. 2006. "Service-Dominant Logic: Reactions, Reflections and Refinements." *Marketing Theory* 6 (3): 281–88. doi:10.1177/1470593106066781.
Lusch, Robert F., Stephen L. Vargo, and Gregor Wessels. 2008. "Toward a Conceptual Foundation for Service Science: Contributions from Service-Dominant Logic." *IBM Systems Journal* 47 (1): 5–14. doi:10.1147/sj.471.0005.
Meszaros, Gerald, and Jim Doble. 1997. "A Pattern Language for Pattern Writing." In . http://hillside.net/index.php/a-pattern-language-for-pattern-writing.
Ng, Irene C.L., and Laura A. Smith. 2012. "An Integrative Framework of Value." In *Toward a Better Understanding of the Role of Value in Markets and Marketing*, 9:207–43. Review of Marketing Research 9. Emerald Group Publishing Limited. http://www.emeraldinsight.com/doi/abs/10.1108/S1548-6435%282012%290000009011.
Normann, Richard. 1984. *Service Management: Strategy and Leadership in Service Businesses*. Wiley. http://books.google.ca/books/about/Service_management.html?id=gT4UAQAAMAAJ.
———. 2001. *Reframing Business*. New York: Wiley.
Normann, Richard, and Rafael Ramirez. 1993. "From Value Chain to Value Constellation: Designing Interactive Strategy." *Harvard Business Review* 71: 65–65. http://hbr.org/1993/07/designing-interactive-strategy.
———. 1994. *From Value Chain to Value Constellation: Designing Interactive Strategy*. Chichester, England: Wiley.
Parhankangas, Annaleena, David Ing, David L. Hawk, Gosia Dane, and Marianne Kosits. 2005. "Negotiated Order and Network Form Organizations." *Systems Research and Behavioral Science* 22 (5): 431–52. doi:10.1002/sres.717.
Pena, William M., and John W. Focke. 1969. *Problem Seeking: New Directions in Architectural Programming*. Houston, TX: Causill, Rowlett and Scott. http://eric.ed.gov/?id=ED037930.

Ramirez, Rafael, and Johan Wallin. 2000. *Prime Movers: Define Your Business or Have Someone Define It against You*. Chichester, England: Wiley.

Selsky, John W., Rafael Ramírez, and Oğuz N. Babüroğlu. 2013. “Collaborative Capability Design: Redundancy of Potentialities.” *Systemic Practice and Action Research* 26 (5): 377–95. doi:10.1007/s11213-012-9257-5.

Spohrer, Jim. 2005. “Why the World Needs More Systems Thinkers Focused on Service Systems.” Conference presentatation presented at the 49th Annual Meeting of the International Society for the Systems Sciences, Cancun, Mexico, July 4. http://isss.org/world/cancun-2005-retrospective#Monday.

Spohrer, Jim, and Stephen K. Kwan. 2009. “Service Science, Management, Engineering, and Design (SSMED): An Emerging Discipline - Outline & References.” *International Journal of Information Systems in the Service Sector* 1 (3): 1–31. doi:10.4018/jisss.2009070101.

Spohrer, Jim, and Paul P. Maglio. 2008. “The Emergence of Service Science: Toward Systematic Service Innovations to Accelerate Co-Creation of Value.” *Production and Operations Management* 17 (3): 238–46. doi:10.3401/poms.1080.0027.

———. 2010. “Toward a Science of Service Systems: Value and Symbols.” In *Service Science: Research and Innovations in the Service Economy*, edited by Paul P. Maglio, Cheryl A. Kieliszewski, and James C. Spohrer, 157–94. 10.1007/978-1-4419-1628-0_9. Boston, MA: Springer. http://dx.doi.org/10.1007/978-1-4419-1628-0_9.

Spohrer, Jim, Paul P. Maglio, John Bailey, and Daniel Gruhl. 2007. “Steps Toward a Science of Service Systems.” *Computer* 40 (1): 71–77. doi:10.1109/MC.2007.33.

Ulrich, Werner, and Martin Reynolds. 2010. “Critical Systems Heuristics.” In *Systems Approaches to Managing Change: A Practical Guide*, 243–92. Springer. http://dx.doi.org/10.1007/978-1-84882-809-4_6.

Vargo, Stephen L., and Robert F. Lusch. 2004a. “Evolving to a New Dominant Logic for Marketing.” *Journal of Marketing* 68 (1): 1–17. doi:Article.

———. 2004b. “The Four Service Marketing Myths Remnants of a Goods-Based, Manufacturing Model.” *Journal of Service Research* 6 (4): 324–35. doi:10.1177/1094670503262946.

———. 2008. “Service-Dominant Logic: Continuing the Evolution.” *Journal of the Academy of Marketing Science* 36 (1): 1–10. doi:10.1007/s11747-007-0069-6.

Winograd, Terry. 1986. “A Language/action Perspective on the Design of Cooperative Work.” In *Proceedings of the 1986 ACM Conference on Computer-Supported Cooperative Work*, 203–20. Austin, Texas: ACM. doi:10.1145/637069.637096.

Received September 2014; revised xxxx 2014; accepted xxxx 2014