The Long and Winding Road to a Course on Service System Design

Robert J. Glushko
University of California, Berkeley

glushko@berkeley.edu
31 May 2011

Abstract

The "Service System" concept has rapidly emerged as a foundational concept in service science as the appropriate abstraction for describing how interconnected sets of services integrate the "value network" in some domain (Maglio et al, 2006; Spohrer, et al 2007). But the novelty and breadth that makes the service system an important idea also makes it challenging to teach. In this paper I describe nine years of experience developing and teaching university courses about service system design. During this period I taught or co-taught five different courses and have retrospectively identified six distinct intellectual and pedagogical approaches in the evolution of the current course. Some key success factors for a course on service system design are: (1) a team design project; (2) an emphasis on design artifacts rather than on a particular design methodology; (3) traceability of design decisions through design artifacts; (4) domain-dependent and domain-independent design patterns; (5) using previous team projects and their design artifacts as examples and metamodels.

Introduction

There is surely ongoing innovation in the teaching of calculus, modern languages, 19th century American history and in many other subjects found in the typical university course catalog. But in established subjects there is significant consensus and conventional wisdom about what concepts are fundamental, the best order to introduce them, the most felicitous examples and exercises, and a canon of well-regarded literature or textbooks.

Courses in service management and operations have been taught for a few decades in business schools and hotel/hospitality management programs, and they mostly emphasize person-to-person or "experience-intensive" services (Zomerdijk and Voss, 2009). New technologies and especially the web have made self-service and multichannel service systems an important part of these curricula, but in most universities where service management and operations are taught the dominant focus is still on domains in which person-to-person encounters remain the most important contributor to value creation (Davis and Bedrow, 2008). Furthermore, a handful of textbooks, many of which have gone through numerous editions (e.g. Davis and Heineke, 2005; Fitzsimmons and Fitzsimmons, 2011), have been widely adopted in these business and management programs, confirming that there is some consensus about important topics and pedagogical approaches.

In contrast, the academic unit in which I teach service system design is a School of Information (or "ISchool"), a novel kind of university program that might be described as either "the intersection of

business, computer science, and social science" or "a 21st century library school." Either characterization suggests a focus on the design of information-intensive systems and services (Apte and Mason, 1995) with a strong emphasis on technology and its organizational or social context. Neither characterization does justice to the aggressive way that most ISchools are developing a multi- or trans-disciplinary character as they seek to prepare students for a society and economy that is increasingly information-driven and services-led.

As a result, the ISchool perspective on the design and operation of service systems differs substantially from that in business or management schools and is also quite distinct from the more practical and enterprise-oriented perspective in MIS programs (Lyons, 2010). The primary difference is that ISchools mix students with business, social science, engineering, or computer science undergraduate degrees in the same multi-disciplinary courses. And even though there is an extensive body of research and practice in database design, business process analysis, data modeling, user interface design and other activities that are essential in the design of information-intensive service systems, very little of this work is explicitly framed in a way that makes it directly applicable (with the notable exception of Alter's "work systems" approach (Alter, 2008)).

It is challenging to develop and teach a graduate-level university course on information-intensive service system design without the foundation provided by an acknowledged canon of literature and consensus about how to teach it. But this meant I could freely innovate without the pressure I would feel if I had to explicitly reject conventional approaches and syllabus materials. My hope is that by explaining the long and winding road I followed to get to my current course I can help others develop courses on service system design that are appropriate for the students and perspective of their own academic institutions.

Six Phases of Course Design and Delivery

The course I am currently teaching is titled "Information Systems and Service Design." (UC Berkeley School of Information , 2011). I have taught it annually in the Fall semester since 2008. Its oldest ancestor is a course titled "Document Engineering" that I first taught in the Spring semester of 2002. This was three years before I had even heard of "service systems," but in retrospect that course was a good first approximation to a course on service system design. In the nine years between my first attempt to teach service system design to the present day, I have taught or co-taught five different courses and have retrospectively identified six distinct intellectual and pedagogical approaches that they embodied.

The path from 2002 to 2011 isn't straight and narrow. In most cases, I initially revised a course in an incremental way by rearranging topics or adding new ones. A year or two later I would realize that more radical changes were required to truly integrate the new content or to provide students with a more effective learning experience. This would result in the abandonment of the old course and the creation of one or more new ones. This evolution of my courses is shown in Figure 1.

The current course is satisfying to teach and is evaluated as successful by the students, but there are surely still detours and side trips to be taken before it reaches the end of its evolutionary path. I conclude this paper by presenting my planned revisions for the course before I teach it again in the Fall semester 2011. That will be the seventh phase of its evolution.

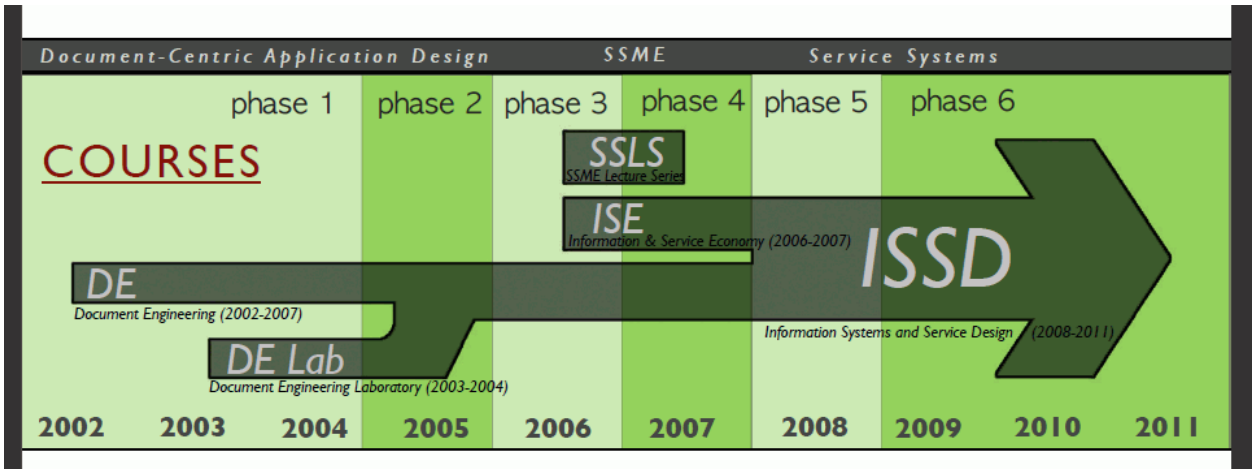

Figure 1. The Evolution of "The Information Systems and Service Design" Course

Phase 1: Systematizing Document Engineering (2002-2004)

I came to the University of California in 2002 after almost 25 years in industry to teach a course I called "Document Engineering" but which in retrospect might have more accurately been titled "Document and Process Modeling for Information-Intensive Service Design." This course was my attempt to reflect on and systematize my own professional experiences in electronic publishing, content management, and business-to-business service systems. In these domains the critical design artifacts are robust and reusable models of documents and the information components from which they are assembled, along with models of the business processes or services for which documents serve as the inputs or outputs. The analysis and design techniques of Document Engineering ensure that the document models are sufficient and efficient when they are implemented as the data format in document automation contexts or as the interface and process specifications in document-centric or service oriented applications.

Two aspects or principles of Document Engineering that were systematized in this first course remain essential in the service systems design course I teach today. The first is the central role of design patterns to capture and communicate best practices in design and implementation, and the second is the emphasis on design artifacts rather than on a particular design methodology to produce the artifacts. I will discuss each of these in turn.

Design Patterns

A design pattern is a model that is sufficiently general, adaptable, and worthy of imitation that it can be reused (Glushko and McGrath, 2005, p. 90). Design patterns simplify work, encourage best practices, and assist in analysis. They also encourage modularity and transparent substitution of equivalent components. In the process of developing the Document Engineering course I identified numerous design patterns for the information exchanges within and between enterprises. Supply chains, business-to-business marketplaces, auctions, information brokers or aggregators, and content syndication networks are examples of inter-enterprise business processes that can be described as patterns of information exchanges. Examples of internal business processes that can be described as patterns of information exchanges include procurement, inventory management, human resources management, and accounting. The existence of common words like catalog, order, invoice, payment and receipt that apply to almost every business domain confirms that there are very general patterns for describing how business gets done.

Over time, as the Document Engineering course evolved into a course explicitly focused on service system design, the repertoire of design patterns I taught evolved as well. For example, as IBM is effectively demonstrating with its "Smarter Planet" marketing campaign, many service systems follow the same design pattern in which sensor networks collect information that is aggregated, analyzed, and acted upon by a combination of human and computational agents (IBM, 2011). But in many domains the document-centric applications I taught in Document Engineering are a subset of the service systems in that domain, so many of the design patterns and the case studies I taught in the Document Engineering course are still relevant in the 2010 Information Systems and Service Design one. A classic paper titled "Staple Yourself to an Order" (Shapiro, Rangan and Sviokla, 1992) that analyzes the patterns by which orders flow through a business became part of my syllabus in 2003 and has remained a part of my service science "canon" ever since.

An Artifact-Centered Design Emphasis

Any design approach involves some steps or methods for scoping, analysis, idea generation, and implementation. Each of these methods creates work products or artifacts that capture their results. Some design approaches emphasize the methods under the assumption that precise or even prescriptive methods are needed to ensure that the artifacts are created correctly and completely. But the Document Engineering approach was biased in the other direction, emphasizing the artifacts that needed to be created without making any prescription about the methods by which that was accomplished.

Document Engineering mandates an artifact-centered design approach because it directly follows from the constraints of the business problems in which patterns of information exchanges are a key part of the solution. The participants in a supply chain or content syndication network can be on opposite sides of the planet, coordinating their activities by information exchanges that are loosely coupled to preserve agility and flexibility in business relationships. In these situations what matters is the information and process models that govern those information exchanges. The document interfaces and the patterns by which documents are produced and consumed are visible, but the methods by which the models were developed or their implementations in the business systems on each side of the information exchanged are neither visible nor relevant. Neither side has any need to impose a prescriptive modeling and design methodology on the other; they only need to exchange their modeling artifacts so they understand what to expect from each other.

This emphasis on design artifacts rather than design methods that is part of the Document Engineering approach remains appropriate in the design of service systems. In information-intensive service systems, an important abstraction that makes it possible to combine person-to-person and computational services is the idea that both kinds of services can be described in terms of the information exchanged by the service provider and service consumer. This abstract view of service interfaces enables loose coupling of services, the transparent substitution of one service provider for another (as in outsourcing), and the transformation of a person-to-person service to self-service or to a computationally provided one that requires no human involvement. What matters is what the interface is and what the service does, which implies a focus on design artifacts.

Assignments and Projects

In a design course "hands-on" assignments and projects are essential. When I came to Berkeley in 2002 I already had many years of experience teaching Document Engineering concepts and skills to my employees in the two companies I had started in the 1990s, and I adapted what I had done with them to the university context. For example, I had students analyze half dozen different calendars from different academic departments and campus organizations to develop an information model that could

accommodate all of them. I also had students analyze the processes by which they selected courses and registered for them and then develop a "To Be" model that made the process more robust and efficient. Unfortunately, my students complained that I expected too much work from them – and they were right – because at first when I thought "this assignment should take a week" I was estimating using a model of a full-time employee rather than that of a student for whom my course was only one-fourth of an academic workload. Some students, especially those with little work experience, complained that my assignments were too "messy" compared with those in other courses – and they were also right about that. I thought it was important for students to deal with legacy data formats, incompatible processes, and other real-world issues, but it was probably wrong to impose this much complexity from the very beginning.

Berkeley ISchool students typically have a few years of post-undergraduate work experience. Not surprisingly, students with more work experience tended to appreciate the real-world context of my assignments and do better work. But this suggests that it is likely to be very difficult to teach service system design to undergraduates because they mostly lack the kinds of work experiences that would help them understand the organizational and technological constraints that service systems inevitably face.

One pedagogical decision I got right was to have students begin information and modeling tasks on their own and then work in project teams to complete them. This approach motivates students to work hard on their individual work and hammers home the principle that all models are biased by the modeler's experience and point of view. The diversity of the academic backgrounds and work experiences among the Berkeley ISchool students made it easy for me to create multi-disciplinary design teams that would reliably experience the intellectual and political negotiations needed to create models that work for multiple stakeholders and different points of view.

In the 2003 and 2004 academic years I co-taught with Brian Hayes a second design called the "Document Engineering Laboratory" that emphasized XML, web architecture, and other technical topics important in the implementation of document-centric service systems by project teams of students. Students liked this second course, but because not all students took both courses and their topical coverage wasn't easy to synchronize, the synergies I expected between the courses didn't fully materialize. I concluded that a semester-long team project was desirable but that it would be have to be an integral part of a single course rather than in a second optional one.

Phase 2: Systematic Document Engineering (2005)

In late Spring 2004, after teaching the Document Engineering course three times, I felt that I had systematized a new design discipline for document-intensive service systems. I thought I would take a few months to turn my 30 lectures with about 30 notes slides in each into a book. My naiveté seems greater each time I recall this audacious plan.

A year of nearly nonstop work later, after enlisting co-author Tim McGrath to help, we had a draft of a book on Document Engineering (Glushko and McGrath, 2005) that was in stable enough form to use when I taught the course in the Spring of 2005. Because my instructions about analysis and design methods and the artifacts they produced were now much more complete, I introduced a team project requirement that year and modified most of the course assignments to be done in support of the project that students had chosen. This was an important change because it makes the assignments cumulative and made it possible to introduce the idea of design traceability – that design decisions should be explicitly based on prior observations and analysis.

A second important change I made in Spring 2005 was to broaden the domain coverage of the course by adding numerous case study readings from industries or business contexts that were markedly different from the electronic publishing and business-to-business areas that are highlighted in the Document Engineering book. These included healthcare informatics, online education, financial services, records management and regulatory compliance. I confess that my motivation for this expansion was to "beef up" the reading list for the course after I had in most cases replaced several readings for each lecture with a single draft chapter from the book. Even though I had spent three years synthesizing everything I knew into the book, students perceive a course as less challenging and interesting if most of the readings came from a single book written by the course instructor. As it turned out, expanding the scope of the Document Engineering course this way led me to appreciate the broader perspective of service system design.

Phase 3: Extending Document Engineering to Service System Scope (2006)

In the Summer of 2005 the Document Engineering book was published. But at that very moment the tight systematic analysis and design approach the book embodied was unraveling in my mind because I was getting intrigued by the emerging notions of service science and service systems. Jim Spohrer of IBM played a critical role in this respect when he told me that he liked my book because it the first thing he'd read that focused on the design of document –centric service systems.

However, the more I learned about service systems the more I realized that the Document Engineering perspective on value creation was incomplete. It focused primarily on the "back stage" of service systems and its theoretical approach for designing the "front stage" was too narrow, dealing mostly with "model-based user interfaces" that use the information and process models of the back stage in a literal and automated way. In addition, the domains of direct and indirect procurement, single-source publishing, and web-based services that had dominated my professional experiences were ones in which value-creation tends to be incremental and deterministic with relatively minimal amounts of value co-creation typical of person-to-person services systems.

So in the Spring of 2006 I made an effort to extend the Document Engineering approach to cover the front stage of information-intensive service systems by adding a few lectures on user interface design and experience design. But it felt a bit tacked on.

Phase 4: Designing a Discipline with Discipline (2006-2007)

At about the same time several Berkeley faculty from different departments, partly encouraged by IBM Faculty Awards, began meeting to discuss the possibility of starting a Service Science, Management and Engineering certificate or degree program. We analyzed existing courses taught from the perspective of our respective disciplines of information systems, business, industrial engineering, and computer science.

We considered selecting a subset of existing courses that at face value could fit into an SSME package but ultimately concluded that this effort alone would not give faculty or students any integrated understanding of what SSME might be. More importantly, if we aimed only to define a curriculum without the intellectual framework provided by a model of a SSME discipline, we would lack any principled way to evaluate it or know how to evolve it over time.

So instead of continuing to evolve my Document Engineering course in an ad hoc way toward some ill-defined service system design target, I took on the challenge of starting with a blank slate and asking "what questions would a service science be expected to answer?" This analysis led to a framework we called the "Discipline by Life Cycle Matrix" (Glushko, 2008), defined by crossing the

activities in a lifecycle with potential disciplines that might have something to say about each activity. So for a lifecycle of Strategy followed by Design (Innovation, Specification, Composition, Realization) and then by Operation (Deployment, Management, Evaluation, Optimization, Evolution), we can ask how business, information science, industrial engineering, economics, law and so on can tell us what to do at each stage and how to do it.

This framework seemed more robust, generative, and abstract than other approaches we considered and seemed to apply to services in every domain. It also provided a clear model of what a curriculum on service system design might want to teach.

"Information and Services Economy" Course

The framework made it obvious that it would be impossible to satisfy the new curricular requirements that we had just identified by adding topics to my existing Document Engineering course. I had already added broadened its domain coverage in Spring 2005, and added some "front stage" design topics in the Spring of 2006. The course by now resembled the proverbial python that has just swallowed an animal, leaving its shape distorted and looking like it would be impossible to fully digest what it was trying to eat. It was clear that at least one new course was needed.

AnnaLee Saxenian and I designed a new course we titled "The Information and Services Economy" as a survey of the emergence of the information and services economy through a review of literature from the numerous disciplines embodied in the "Discipline x Lifecycle Matrix." We started with broad, framing ideas and classic papers from economics and organizational theory about the nature of work and how it is organized. We taught how technology, business architectures, and the law coevolve to explain the progression from the manufacturing era to the 21st century. We took a top-down approach that briefly introduced service design and the service life cycle, but we intentionally deferred details of design methods, technology and artifacts to more specialized courses like Document Engineering.

We encouraged students interested in service science to take the Information and Services Economy course in the fall semester and Document Engineering in the spring because on paper the two courses did a good job introducing many of the theoretical and practical topics needed to design service systems. But once again, the courses tended to appeal to different sets of students, and in the two years in which both courses were taught (2006-2008) only a disappointingly small handful of students took both courses.

Service Science Lecture Series

Because I personally wanted to understand services better and to assess student interest in a potential SSME certificate or degree program at Berkeley, we organized a year-long Service Science lecture series in the Fall of 2006 and the Spring of 2007. Its purpose was to bring in experts in a variety of service system domains, most of whom were service industry executives or senior consultants, which a smattering of academics. In the Spring semester I tried to schedule speakers in the lecture series to complement the topics in my Document Engineering course. But while the lecture series was successful in its own right, it didn't work as an adjunct to my course, for much the same reasons that a separate project course had failed to yield my hoped-for synergies a few years earlier.

Phase 5: The Lifecycle "Methodology Mixup" (2008)

Since our best-laid plans for a two-course sequence on service science and service design had gone awry, I decided in the summer of 2008 to stop teaching both courses and to design a new course that would combine the most successful and necessary parts of each. This meant that I would preserve

some of the management, strategy and organizational design topics from the Information and Services Economy course and keep the information and process modeling parts of Document Engineering that applied broadly to service systems.

I was concerned that it would be difficult to incorporate management and strategy topics into a service systems perspective but was fortunate to find a recently published book that fit the bill, "Outside-In Software Development" (Kessler and Sweitzer, 2008). Despite its title, this book is much less about software development than about the organizational context of software design. It discusses business model, product management and ecosystem considerations I'd not seen in other software engineering and software project management texts. In particular, the text emphasizes the idea that the essence of good design is making tradeoffs to finding the compromises among multiple stakeholders with different goals and constraints.

Because of the existing focus of Document Engineering on document-centric and information-intensive service systems, I knew I needed to add new material covering experience-intensive service systems. Here I relied on much of the classic work on service design and operations commonly taught in business schools and hotel /hospitality management programs to introduce blueprinting (Bitner et al, 2008), "touchpoint" design, and similar "front stage" topics (Teboul, 2006). I also brought in many topics normally included in usability and user-centered design courses, such as ethnographic / contextual analysis (Holtzblatt et al, 2005) and persona development (Pruitt and Grudin, 2003).

"Information Systems and Service Design" Course

The new course I taught in Fall 2008 was titled "Information Systems and Service Design" (UC Berkeley School of Information, 2011). My syllabus followed the phases in a conventional iterative design lifecycle and at each phase I would introduce two kinds of methods, one more appropriate for information-intensive service systems and one more appropriate for experience-intensive ones. For example, on the continuum between "experience-intensive" and "information-intensive" service systems "documents" and other information sources go from being incidental or occasional to being ubiquitous and intrinsic to the goals and activities of the stakeholders and actors. On the "experience-intensive" end of this continuum the most important things to study are the human participants, so I taught ethnographic techniques of "contextual inquiry" that emphasize participant observation and interviews. In contrast, on the "information-intensive" end we need to pay most attention to the documents, so I taught the methods for document discovery and analysis that I'd systematized in Document Engineering. Similarly, when I covered process analysis I contrasted service blueprinting, typically used to model services with a dominant person-to-person context, with UML sequence diagrams (Booch et al, 1999) and information flow models (Eppinger et al, 1994), typically used to describe service systems with a strong computational or information processing character.

I had imagined that this approach of mixing the typical experience design methodology with the typical information systems design methodology would enable students to understand how designers with different experiences and perspectives could learn to work together and adapt to different design contexts. I had just written (with Lindsay Tabas) a paper about the need to recognize that experience or "front stage" designers and information systems or "back stage" designers looked at design problems in different and often incompatible ways (Glushko and Tabas, 2008). The two kinds of designers often have different educational backgrounds and organizational status, so they don't always understand and appreciate their respective design contributions.

I had expected my "methodology mix" to be intellectually provocative for students but for some of them it turned out to be more confusing than challenging. My plan was to broaden the

methodological breadth of my course, but I had traded coverage for coherence. One of the students said that my lectures seemed a bit schizophrenic because they gave conflicting design advice and didn't provide principles for resolving the conflicts. As I had seen in previous courses, students with less work experience favored a more prescriptive approach to design, but to me it was essential to teach the need to adapt the design methodology to the specific context and constraints posed by each situation.

Projects as Case Studies and Design Patterns

Who wouldn't be disappointed to learn that some students found his lectures a bit schizophrenic? I took some consolation in my interpretation that the broader coverage of the design lifecycle and design methods enabled students to take on more ambitious team projects that were more diverse and more "service system like" than the projects they had done in my now-retired Document Engineering course. The projects were:

- "Open Textbooks" - a web-based mashup service system for students to purchase textbooks at the best available price, speed, or convenience to the student
- "Courseland - a "services platform" that connects with university legacy applications that simplifies administrative tasks for students, including course registration, changes to course schedules, payment of university fees and tracking of course/degree requirements.
- "University Village Energy Management" - a service system that enables the student tenants of a housing complex to better manage their energy usage, integrate consumption and billing services, and create social incentives for conservation with visualizations in public spaces in the housing complex
- "Bookland" - a service system designed for a large bookstore chain that also has a web retail site. Its goals were to improve customer satisfaction, increase sales, improve the efficiency of multichannel store operations, and enable the company to gather more useful marketing information with location-based offers and book tracking.

I realized that student projects would make excellent case studies in later courses because they would better reinforce my way of conceptualizing and teaching service system design than any other reading I might assign. So I required all the projects to follow templates for presentations and final reports, which made them easily "consumable" additions to my course syllabus (see Chapter 4, "Making Products Consumable" in Kessler and Sweitzer, 2008). Assigning the 2008 project case studies in 2009 made it easier for students to define and scope a project and get started on it earlier. As a result most 2009 projects seemed even more ambitious and thorough than the 2008 case studies that they learned from, which in turn made the 2009 project final reports even better case studies for the 2010 course. This is a virtuous cycle that I intend to implement in every design course I teach from now on.

Phase 6: Design Patterns for Service System Design (2009-2010).

When I reflected on why the lecture part of my new course hadn't gone as well as I hoped, I realized that I had taken for granted that students would grasp the service system concept and see how the front stage and back stage worked together. I had discussed this idea in the first lecture but didn't revisit it as much as I should have. I had also forgotten two success factors that I had built on in previous courses. First, I had not exploited design patterns as much as I used to. Second, I had started emphasizing design methodology instead of the artifacts produced by the design methods.

So after teaching it just once, I substantially revised my Information Systems and Service Design course for the Fall of 2009. I centered the course on the service system concept and dealt head on with the perceived conflict between front stage and back stage design methods. I spent several lectures

discussing service quality and value creation models to ensure that students appreciated that these are best understood as properties of the service system (Heskett et al, 1994; Parasuraman et al, 1984). This new pedagogical approach worked well: students with a front stage design bias better appreciated that the "moment of truth" that their work emphasized depended on the back stage satisfying the preconditions for a successful service encounter. Similarly, students with a back stage design bias came to view the importance of the front stage in preserving or amplifying the value created by back stage.

Design Patterns that Bridge the Front and Back Stages

Learning to view the service system as an integrated value creation system became significantly easier in this revised course because I presented many of the central design choices as tradeoffs between front stage or experience-intensive and back stage or information-intensive approaches. This recast what students had previously seen as conflicts into design patterns. For example:

- The placement of the "line of visibility" between the front and back stage is the parameter of a design pattern that changes how much of the value creation is visible to the customer and potentially personalizable
- A generic service offering can be increased or decreased in service intensity by changing the number of touch points
- Capturing, managing, integrating and retrieving information allows service providers to substitute information for interaction; high intensity or many touch points are unnecessary if stored information or computation makes interaction unnecessary
- The level of abstraction of the interfaces that specify the inputs and outputs of a service (how the service is requested and how the service provider responds) determines the degree to which one provide can be transparently substituted for another.

The "Seven Contexts" Design Pattern

These design patterns differentiate service systems of the same type or industry domain, but they can be "rolled up" into a more abstract framework that also helps us understand service systems as they vary across many domains and over time. This framework, which I have called "Seven Contexts"(Glushko, 2010), takes an abstract view of service contexts that highlights what different kinds of services and service encounters have in common rather than emphasizing their differences. It treats different design contexts as building blocks that can be combined to create complex service systems. The Seven Contexts (Glushko, 2010, p. 221-222):

- The "person-to-person" (Context 1), "self-service" (Context 3), and "multi-channel" (Context 4) ones are canonical in service design.
- Context 2, "technology-enhanced person-to-person" service, is included to highlight the design issues that emerge in contexts that are transitional or intermediate between "pure" person-to-person service encounters and self-service ones.
- Context 5, "services on multiple devices or platforms," combines and specializes many of the design concerns for the "self-service" and "multi-channel" contexts, but it raises additional distinctive ones.
- Context 6, called "back-stage intensive" or "computational" here, is a subset of what are often called "machine to machine" or "computer to computer" services, but these labels are less precise than needed when additional contexts are introduced.
- Context 7, "location-based and context-aware services," combines and specializes design concerns from "self-service," "multi-channel," and "back stage" contexts (3, 5, and 6), but likewise raises new ones.

When I revised the Information Systems and Service Design course in Fall 2009, I used this Seven Contexts design pattern as the new organizational framework for the course syllabus. I introduced each of the seven contexts along with the characteristic design challenges each poses for service system designers, along with the design methods best suited for responding to those challenges. Throughout the semester we analyzed numerous service systems in terms of the seven contexts and they proved sufficient to describe all of them (note: the "Bookland" team project from the 2008 course is analyzed as a case study in the Seven Contexts paper (Glushko, 2010, p. 240-245)).

The Portfolio Model of Methodology

The focus on design patterns had the beneficial side effect of undermining the notion of strict adherence to a particular design methodology that had been implied in the 2008 version of the Information Systems and Service Design course. Seeing that different design contexts implied different design methods made students realize that they would be unlikely to follow the same design approach on every project. There would always be a need to adapt the methodology in some way, emphasizing some activities more or less than normatively specified by a "standard methodology" because of the particular service system domain, schedule, resource, or stakeholder considerations. So in practice, the "methodology" employed in any given project is likely to be a subset of design techniques selected and adapted for it from a larger portfolio. IBM Consulting calls this customized subset methodology the "Engagement Model" (Ominsky et al, 2002).

Teaching students design patterns encouraged them to think of themselves as service system designers rather than more narrowly as front stage or back stage designers. The "methodology mix" that didn't work well in Fall 2008 seemed to go down a lot better in the Fall 2009 and Fall 2010 versions of the course because I had been able to get students to focus on end-to-end considerations of value creation in which different methods make sense for different lifecycle phases and different combinations of the seven contexts.

Phase 7 – What's Next? (2011 and beyond)

I have always pushed students to take on "messy" design projects that involve real-world constraints, clients and stakeholders who might have conflicting requirements, corporate strategies or politics to align with and negotiate, and legacy data and applications that have to be dealt with. But more realistic design projects are hard to fit into a single semester, and real world constraints undermine the normative schedules assumed by textbook methodologies. How can we give students realistic but tractable design experiences?

In Fall 2011 when I next teach service system design my students will be working in a "guided design environment" that combines many of the functions of a course management system with support for following design methods and creating design artifacts. The course management functions will keep track of the syllabus readings, instructor's lecture notes, and other information that implements the academic components of the course. The design functions will allow the students to follow the path through the design lifecycle that is appropriate for their project and give them support and templates for creating design artifacts. The content management system that underlies the design environment will encourage traceability of design decisions by enabling easy linking of work in earlier phases to later decisions. Social media capabilities will allow students to contribute relevant and up-to-date content into the corpus by adding web bookmarks to relevant lectures, readings, and design artifacts.

This structured framework for design will enable the instructor to assess student work despite the variety in schedules and work products that more realistic projects will yield. It should also greatly increase the ability to reuse work from previous courses as case studies, design examples, and so on. This will be the seventh phase of the evolution of my service system design course. I doubt that it will be the last.

Figure 2 summarizes the key concepts I emphasized in each phase of my teaching, the courses in which those concepts were embodied, and my publications that captured my thinking at each point.

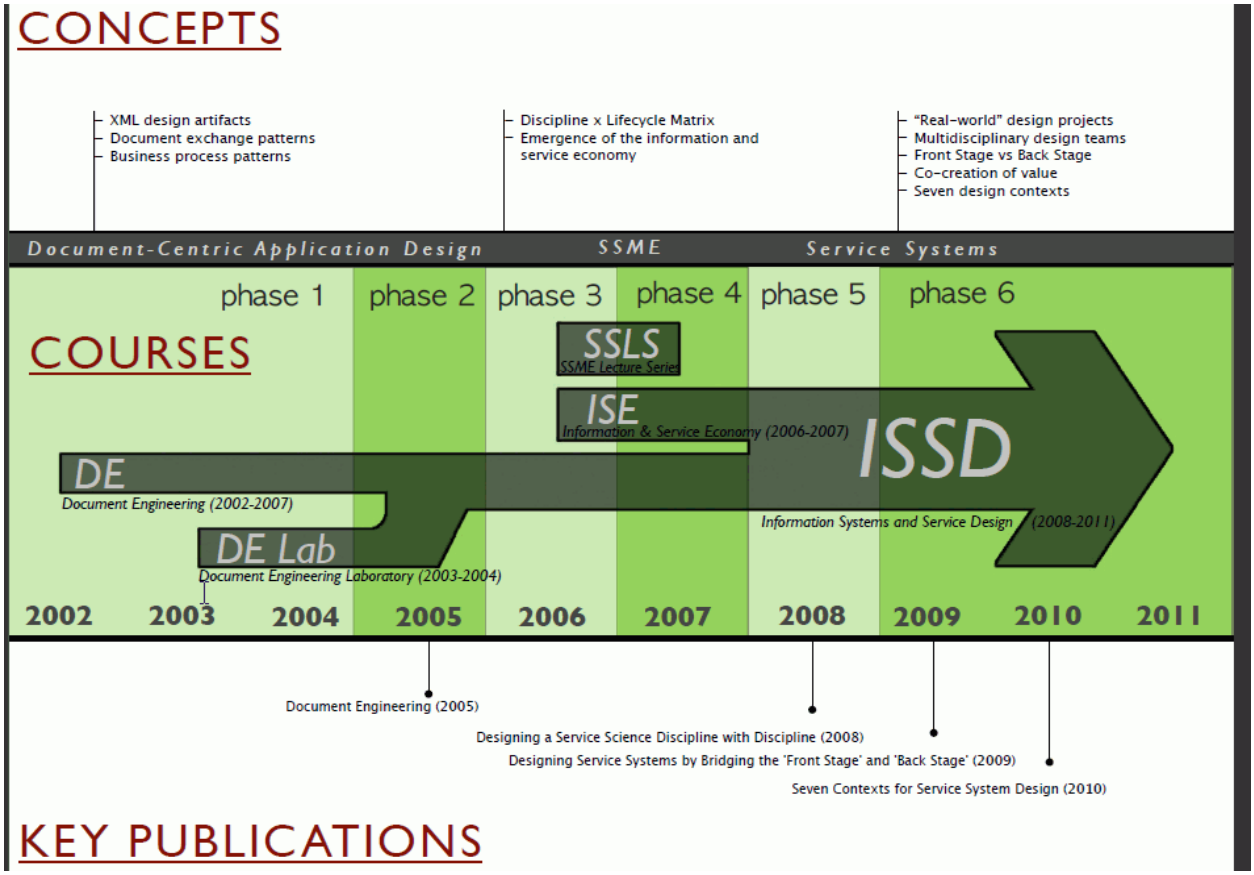

Figure 2. The Long and Winding Road to a Course on Service System Design

Acknowledgments

I have written this paper as a personal retrospective but the story it tells involved collaborations with other UC Berkeley Information school faculty, especially AnnaLee Saxenian and Brian Hayes. Many graduate student research and teaching assistants also made substantial contributions, but rather than list them and inadvertently omit someone let me just thank them all as a group. Jim Spohrer and Paul Maglio at IBM Almaden made me realize I had been teaching service system design since 2002 and have helped me evolve that first course into something that explicitly sets out to do that.

References

Alter, S. (2008).Service system fundamentals: Work system, value chain, and life cycle, *IBM Systems Journal*, 47(1): 71 - 85, 2008.

Apte, U., and Mason, R. (1995). Global Disaggregation of Information-Intensive Services, *Management Science*, 41(7): 1250-1262, July 1995.

Bitner, M., Ostrom, A., and Morgan, F. (2008). Service Blueprinting: A Practical Technique for Service Innovation. *California Management Review*, 50(3): 66-94.

Booch, G., Rumbaugh, J., and Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, Reading, MA, 1999

Davis, M. M., and Berdrow, I. (2008). Service Science: Catalyst for Change in Business School Curriculum, *IBM Systems Journal*, 47(1): 29-39, 2008

Davis, Mark M. and Janelle Heineke (2005) *Operations Management: Integrating Manufacturing and Services.* Fifth Edition. McGraw-Hill Irwin. Boston, MA.

Eppinger, S.D., D.E. Whitney, R.P. Smith, D.A. Gebala. 1994. A model-based method for organizing tasks In product development. *Research in Engineering Design*, 6(1): 1-13.

Fitzsimmons, J. A., and Fitzsimmons, M. J. (2011). *Service Management*. Seventh Edition, McGraw Hill.

Glushko, R. (2008). Designing a Service Science Discipline with Discipline, *IBM Systems Journal*, 47(1): 15-27, 2008.

Glushko, R. (2010). Seven Contexts for Service System Design, in Maglio, P. P., Kieliszewski, C, & Spohrer, J. *Handbook of Service Science*, 219-249, 2010.

Glushko, R., and McGrath, T. (2005). *Document Engineering: Analyzing and Designing Documents for Business Informatics and Web Services*. MIT Press.

Glushko, R. & Tabas, L. (2009) "Designing Service Systems by Bridging the 'Front Stage' and 'Back Stage'", *Information Systems and E-Business Management*, 7(4):407-427, 2009.

Heskett, J., Jones, T., Loverman, G., Sasser Jr., W., and Schlesinger, L. (1994). Putting the Service-Profit Chain to Work. *Harvard Business Review*, 72(2): 164–174.

Holtzblatt, K., Wendell, J., and Wood, S. (2005). *Rapid contextual design: a how-to guide to key techniques for user-centered design*. Morgan Kauffman.

IBM (2011). Smarter Planet. http://www.ibm.com/smarterplanet

Kessler, C., and Sweitzer, J. (2008). *Outside-in Software Development*, IBM Press.

Lyons, K. 2010. Service Science in iSchools. iConference, 2010, February 3-6, 2010, Urbana-Champaign, IL (https://www.ideals.illinois.edu/bitstream/handle/2142/14946/KLyons-final-submitted-iConference-service-science-ischools.pdf?sequence=2)

Maglio, P., Srinivasan, S., Kreulen, J., and Spohrer, J. (2006). Service Systems, Service Scientists, SSME, and Innovation. *Communications of the ACM*, 49(7): 81-85.

Ominsky, M., Stern, K., and Rudd, J. (2002) User-centered design at IBM consulting. *International Journal of Human-Computer Interaction* 14(3–4):349–368

Parasuraman, A., Zeithaml, V., and Berry, L. (1985). A Conceptual Model of Service Quality and Its Implications for Future Research, *Journal of Marketing*, Autumn 1985.

Pruitt, J., and Grudin, J. (2003). Personas: Practice and Theory, *Proceedings of the Converence on Designing for User Experiences*, 2003.

Shapiro, B.P., Rangan, V., Sviokla, J., (1992). Staple yourself to an order. *Harvard Business Review* July/August, 113–122.

Spohrer, J., Maglio, P., Bailey, J., and Gruhl, D. (2007). Steps Toward a Science of Service Systems, *IEEE Computer*, January 2007, 71-77

Teboul, J. (2006). *Service is Front Stage*. Insead Business Press, 2006.

UC Berkeley School of Information (2011). Info 228: Information Systems and Service Design http://www.ischool.berkeley.edu/courses/228

Zomerdijk, L., and Voss, C. (2010) Service Design for Experience-Centric Services, *Journal of Service Research*, 13(1): 67-82.