RJ10416 (A0710-003) October 3, 2007
Computer Science

IBM Research Report

Towards an Agile Service System for a Global Call Center

Isaac Cheng, Vikas Krishna, Neil Boyette, Joel Bethea
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

Towards an Agile Service System for a Global Call Center

Isaac Cheng, Vikas Krishna, Neil Boyette, and Joel Bethea

IBM Almaden Research Center,
650 Harry Road, San Jose, CA 95120, U.S.A.
isaacc@us.ibm.com
vikas@us.ibm.com
nboyette@us.ibm.com
bethea@us.ibm.com

Abstract: In this paper, we share our experience with applying service-oriented techniques to improve the agility of a real-world global service system. We improved the process independence of the service system by decoupling application development from process management using a framework that is based on models and rules. We analyzed various aspects of the system and the impact of technologies being deployed. Our experience is documented as a case study as we anticipate that similar techniques can be applied to improve the agility of other significant service systems in the future.

1 Introduction

In the new Business Process Management paradigm, business agility can be achieved by decoupling application development from process management, very much like the power and flexibility realized from the decoupling of application development from data management in relational database technology. Towards a goal of improving process independence, the authors leveraged techniques of Service Oriented Architecture (SOA) and incorporated technologies such as web services and the Eclipse Modelling Framework (EMF), to develop the Custom Call Flow (CCF) framework [CBK06], [Ec06].

To test CCF in the real world, the authors worked closely with the Call Management (CM) team of a global enterprise to develop a tool and a system to make use of the framework. The resulting solution provided a means for management of call flow processes independent of the call flow application, and thus improved the overall agility of the system. In particular, the business processes and workflows can be easily changed by business analysts with a CCF-based authoring tool without incurring application code change. Independent from the processes, application development has been off-shored to another country, where software developers use a CCF-based software development kit. This strategic off-shoring resulted in better utilization of both technical and business resources. Specifically, code change in the application performed by the development team does not affect the business processes and workflows. One way of understanding the work in CCF is to look at is as a service system case study using the methodologies from Services Sciences, Management and Engineering (SSME) [Ma06], [Sp06], [Te06].

2 Call Center and Call Flows

A call center is a centralized location where companies manage inbound customer requests. To ensure consistent high quality of service to customers, companies often define formal processes, such as the entitlement process. Call flows can be designed to help customer service representatives comply with these processes. A call flow, a special type of workflow, describes the steps that systematically guide a customer to resolution of his or her request in multiple scenarios. Managing call flows is challenging for many reasons:

- Different call centers throughout the world have different needs, and require different applications to consume call flows. It is economical for different applications to share a common set of call flows.

- Products often need to be updated frequently. So do services and call flows. Services need to be available during call flow updates.

- Customers often provide their service representatives with unsolicited information. A call flow execution engine should take customers' input in any order and quickly compute the optimal path to a resolution.

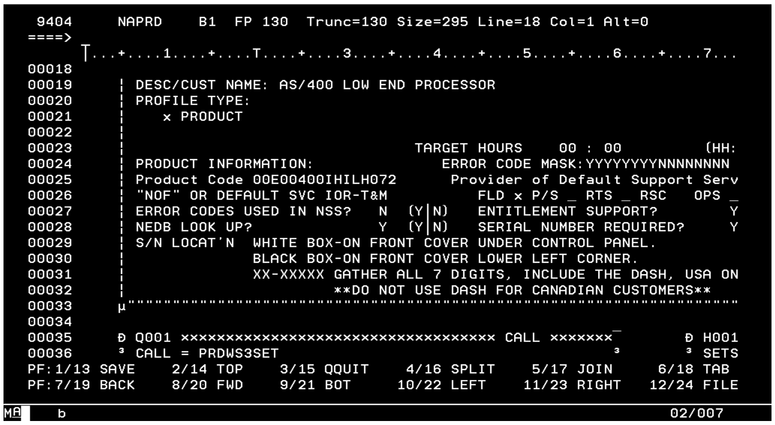
Call flows are designed to provide consistent management of post sales product support and as a services delivery system. Call flows guide the interaction from first customer contact until the time that the customer concurs that our solution solved their problem.

3 The Problem

Before we introduced CCF, the original call flow management system was a closed monolithic system where most major components were tightly coupled. The authoring tool, the runtime front-end, and the runtime back-end communicated with one another using a proprietary protocol, and the call flows themselves were written using a traditional programming language. As a result, the inherent lack of agility in the system has become a major problem.

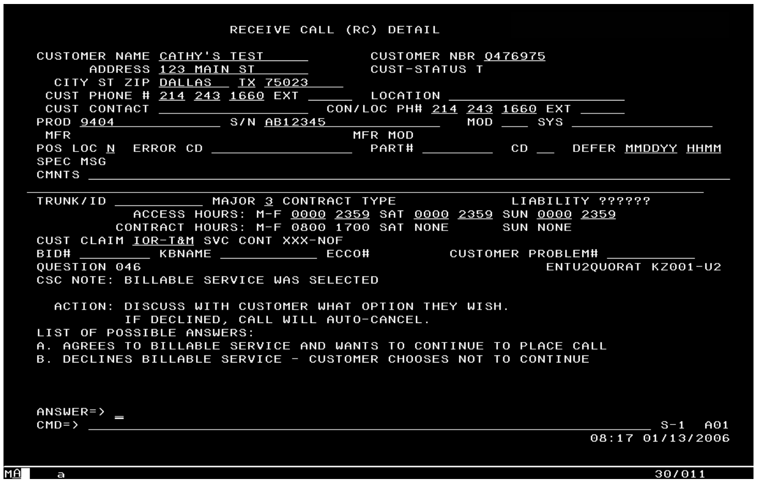
Call flows are often designed to support specific products, and as the products evolve, call flows must change accordingly. The call flow authoring tool used by the original call flow management system was tightly coupled with the rest of the system, and updating the call flows had been difficult (see Figure 1). It had also been very difficult to evolve the authoring tool to keep up with the current user-interface (UI) technologies. In particular, it was hard for a business architect to get a holistic view of a call flow to understand the logic of the call flow process in relation to the design of the UI. This was further complicated by the fact that the call flows were defined in code which was difficult for business analysts to understand. Finally, because of limitations of the proprietary protocol between the authoring tool and its call flow repository, supporting foreign languages and cultural information was impossible. All of these factors led to an authoring tool that was cumbersome and difficult to use, making the process of updating call flows extremely error prone.

The runtime components suffer from similar problems. The UI could hardly keep up with modern technologies (see Figure 2) and because of the tight coupling of the system, it was difficult to replace any algorithm, such as the one for business-rule inference, with a better one.



```
 9404     NAPRD    B1  FP 130  Trunc=130 Size=295 Line=18 Col=1 Alt=0
====>
       T...+....1....+....T....+....3....+....4....+....5....+....6....+....7...
00018
00019     ¦ DESC/CUST NAME: AS/400 LOW END PROCESSOR
00020     ¦ PROFILE TYPE:
00021     ¦    x PRODUCT
00022     ¦
00023     ¦                               TARGET HOURS    00 : 00          (HH:
00024     ¦ PRODUCT INFORMATION:             ERROR CODE MASK:YYYYYYYYNNNNNNNN
00025     ¦ Product Code 00E00400IHILH072      Provider of Default Support Serv
00026     ¦ "NOF" OR DEFAULT SVC IOR-T&M          FLD x P/S _ RTS _ RSC   OPS _
00027     ¦ ERROR CODES USED IN NSS?   N  (Y|N)   ENTITLEMENT SUPPORT?        Y
00028     ¦ NEDB LOOK UP?              Y  (Y|N)   SERIAL NUMBER REQUIRED?     Y
00029     ¦ S/N LOCAT'N  WHITE BOX-ON FRONT COVER UNDER CONTROL PANEL.
00030     ¦              BLACK BOX-ON FRONT COVER LOWER LEFT CORNER.
00031     ¦              XX-XXXXX GATHER ALL 7 DIGITS, INCLUDE THE DASH, USA ON
00032     ¦                       **DO NOT USE DASH FOR CANADIAN CUSTOMERS**
00033     µ""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
00034
00035     Ð Q001 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx CALL xxxxxxx¯       Ð H001
00036     ³ CALL = PRDWS3SET                                   ³       ³ SETS
PF:1/13 SAVE    2/14 TOP    3/15 QQUIT    4/16 SPLIT    5/17 JOIN     6/18 TAB
PF:7/19 BACK    8/20 FWD    9/21 BOT     10/22 LEFT    11/23 RIGHT   12/24 FILE
MA    b                                                              02/007
```

Figure 1: The Authoring Tool of the Existing Call Management System



```
                          RECEIVE CALL (RC) DETAIL

   CUSTOMER NAME CATHY'S TEST______        CUSTOMER NBR 0476975
         ADDRESS 123 MAIN ST_______        CUST-STATUS T
     CITY ST ZIP DALLAS__ TX 75023____
    CUST PHONE # 214 243 1660 EXT _____    LOCATION ____________________
    CUST CONTACT _________________    CON/LOC PH# 214 243 1660 EXT _____
   PROD 9404____________ S/N AB12345_____________   MOD ___  SYS ________________
    MFR                                   MFR MOD
   POS LOC N  ERROR CD ________________   PART# ________   CD __   DEFER MMDDYY HHMM
   SPEC MSG
   CMNTS __________________________________________________________________________
 ______________________________________________________________________________
   TRUNK/ID __________  MAJOR 3 CONTRACT TYPE                LIABILITY ??????
              ACCESS HOURS: M-F 0000 2359 SAT 0000 2359 SUN 0000 2359
            CONTRACT HOURS: M-F 0800 1700 SAT NONE      SUN NONE
   CUST CLAIM IOR-T&M SVC CONT XXX-NOF
   BID# ________ KBNAME ___________ ECCO#         CUSTOMER PROBLEM# __________
   QUESTION 046                                           ENTU2QUORAT KZ001-U2
   CSC NOTE: BILLABLE SERVICE WAS SELECTED

     ACTION: DISCUSS WITH CUSTOMER WHAT OPTION THEY WISH.
             IF DECLINED, CALL WILL AUTO-CANCEL.
   LIST OF POSSIBLE ANSWERS:
   A. AGREES TO BILLABLE SERVICE AND WANTS TO CONTINUE TO PLACE CALL
   B. DECLINES BILLABLE SERVICE - CUSTOMER CHOOSES NOT TO CONTINUE


   ANSWER=> _
   CMD=> ________________________________________________________________ S-1  A01
                                                              08:17 01/13/2006

MA    a                                                             30/011
```

Figure 2. The Runtime UI of the Existing Call Management System

The lack of agility of the existing call management system imposed unnecessary development, maintenance, and operational costs and degraded the quality of service. A new solution should leverage the existing substantial investment and work with the existing infrastructure on a low-risk migration path.

4 The Solution

Learning from the experience of the existing call management system, we designed CCF to provide business and technical agility. Our goal was to provide flexibility in both the framework architecture and the call flow update process. As an example, popular Web 2.0 technologies such as AJAX and wiki are cutting edge right now, but may become an obsolete legacy in just a few years. In a similar way, CCF has been designed with loose coupling between components, allowing individual components like the UI from the example, to be updated or replaced as technologies or business needs change without redesigning the whole framework. The same idea was used for the authoring tool and content repository. To facilitate these goals, we adopted a SOA approach in the CCF design.

Prior to implementation of the SOA components, we designed a model based on the requirements using EMF. The model was then used by both the authoring tool and the runtime engine and allowed each to leverage the benefits of a Model Driven Architecture (MDA). The authoring tool is an Eclipse based application for business architects to visually create, edit and manage call flows. This gives the business analysts a direct view into the process by displaying the elements and flow of the process in terms of a visual workflow rather than in code. The runtime engine is provided as a separate component. It provides a Java application development interface (API) which developers can use to write client applications on any platform using the UI toolkit of their choice. Both the authoring tool and the engine communicate with a call flow repository through Web services. In addition the runtime engine can interact with external services using SOA, enabling arbitrary business functionality to be incorporated into the call flow execution. The use of SOA in this manner allows business analysts to change business processes used in a call flow independently of the application UI and code. Neither the business analysts nor the UI application developers need to understand the underlying EMF model.

SOA allows us to leverage both the existing and new investments through data mediation. As a proof of concept, we have shown that CCF works with both legacy and newly introduced third-party back-ends. The legacy backend is CEDS, which is the same back-end used by the existing call management system. With CCF, sophisticated decision analysis has been implemented, enabling business analysts to design the call flow to choose which back-end to invoke at runtime. Whereas in the old system such decisions had to be hard coded into the system at design time, this method allows the decision to be deferred until runtime when more information pertinent to the decision is available. The runtime engine uses an efficient pattern-matching algorithm implemented by the Agent Building and Learning Environment [Ab06] to make intelligent decisions based on the rules of the business as defined by the analysts and the available information at runtime. Neither the business analysts nor the application developers need to understand the sophisticated pattern-matching algorithm. CCF appropriately hides both the model and the rule inference algorithm within the framework allowing both business analysts and UI application developers to work productively and independently. Agility is achieved by process independence.

5 Analysis

The CM comprises a service provider, service client, and a service target that is being transformed as a result of the service (see Figure 3). These elements and relationships are a useful way to describe a service system [Ga02]. The service provider is the call center of a global enterprise, and the target being transformed is the customer and case information.

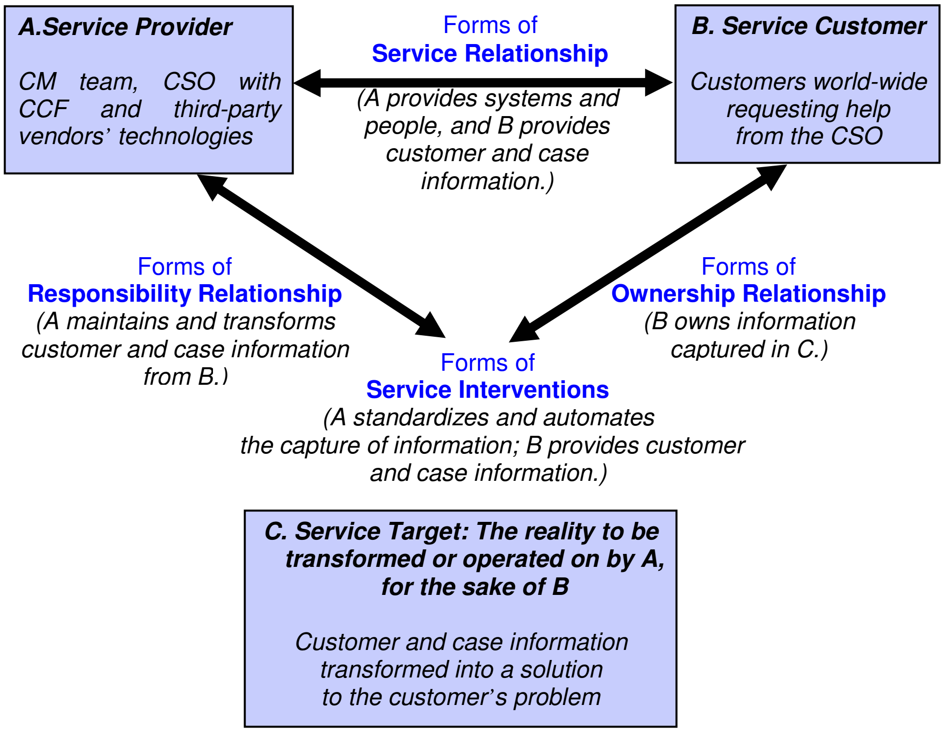

A.Service Provider
CM team, CSO with CCF and third-party vendors' technologies
Forms of
Service Relationship
(A provides systems and people, and B provides customer and case information.)
B. Service Customer
Customers world-wide requesting help from the CSO
Forms of
Responsibility Relationship
(A maintains and transforms customer and case information from B.)
Forms of
Ownership Relationship
(B owns information captured in C.)
Forms of
Service Interventions
(A standardizes and automates the capture of information; B provides customer and case information.)
C. Service Target: The reality to be transformed or operated on by A, for the sake of B
Customer and case information transformed into a solution to the customer's problem

Figure 3. The CM Service System

The call center consists of a Customer Service Organization (CSO) and a Call Management team. When a customer encounters a problem, she calls the CSO for help. A Customer Service Representative (CSR) answers the call and asks for both customer-specific and case-specific information. An example of customer information is a customer ID, which is linked to the terms of a service contract. This information is useful for the CSO for making decisions on service entitlement, up-selling, and crossselling service contracts based on its business rules. The case information is pertinent to the problem that the customer encountered. For instance, a customer may encounter the infamous “blue screen of death” syndrome. The system supporting the CSO may infer that it is probable that the problem was caused by an operating system made by a particular vendor. The CCF framework accepts modern inference engines as plug-in modules to support various kinds of rule-based transformations. The customer and case information is transformed into a solution to the original problem that the customer encountered.

The goal of this case study is to understand the tangible and immediate effects of deploying the new technologies under the innovative CCF framework. The analysis boundaries that the authors chose have provided actionable insights although we do not exclude other less obvious and potentially better approaches. The attributes of this service system through the three stages of its lifecycle are shown in Table 1 below.

Table 1. Stages of the CCF service system engagement

Solution Lifecycle	Description	People (Who?, How many?)	Technology (What?)	Shared Information (How?)	Value Proposition (Why?)
Initial Phase	Contact CM Gather requirements Propose Solution	Research Managers (1) CM process owners	Open minded approach	Formal & Informal Presentations Demo's of relevant applications SWOT Analysis	Identify needs and match, proven ROI
Development & Deployment Phase	Prototype solution Demo prototype Implement and deliver solution Stress test solution Deploy solution	CCF/ Research Development Team (4) Call Flow Administrators (tens) Call Flow Authors (hundreds) CSRs (thousands) located in various countries throughout the world Customers in all countries IBM does business in.	CCF Eclipse AJAX XML ABLE WebSphere™	Call flows Requirement documents Business process documents	High operational efficiency User-friendly and reliable call flow and rule updates on a daily basis
Training & Support Phase	Train and support	IGS – L-1, L-2 Research – L-3 (4)	Phone, pagers, email, and remote desktop	Training material Extensive documentation	Research became a trusted advisor and partner to CM

Service systems should satisfy all stakeholders. In the simplest interpretation of a service system, there are two stakeholders – a single service provider and a service consumer. It is one party doing something of value for another party. In a complex service system there are many different stakeholders, each with different expectations, arising from their relative needs and goals. Since many complex service systems are nested and recursive, with each instance of sub-service systems serving both internal and external stakeholders, it can become nearly unmanageable. Thus, a challenge in analyzing complex service systems is to evaluate them from the perspective of all service stakeholders. In this case study, we chose to simplify the assumptions by not overly emphasizing the service sub-system of IBM research providing a service to CM, even though in reality there are significant implications to this arrangement. The stakeholders interests in this case study are summarized in Table 2.

Table 2. Stakeholders

System Stakeholders	Challenges (before)	Benefits (after)
CM	• Difficult to off-shore development due to the inflexibility of the monolithic US-based existing system. • Decide to engage with research. • Pick third-party back-end vendors. • Make buy vs. build decisions on various components in the SOA. • Limited in business opportunities to pursue due to monolithic existing system.	• Able to off-shore development to China resulting in a 50% saving. • Additional revenue from selling similar WebSphere solutions. • Improved customer satisfaction • Increased customer loyalty, and potentially increased demand • Increased opportunities due to being able to offer same service through multiple channels (CSR, self help, chat, email, voice, autonomic etc.) and in multiple localizations.
CCF/Research	• Understand business and technical requirements • Provide the technical solution and expertise	• Tremendous credibility in its value proposition to prospective clients • Great platform for further CCF-based innovation and transformation of call centers

Call Flow Authors (a.k.a. Process Owners)	• Very difficult to update call flows • Very difficult to update business rules	• Improved productivity as they benefit from easier call flow updates • Improved productivity as they benefit from easier business rule updates • Reduced errors in any update due to syntax checking and easy-to-use testing facility of the new Eclipse-based authoring tool.
CSO Managers	• No way to make service keep up with the fast evolution of products • CSR turn-over problems. Very expensive to train new CSRs to use the existing tools.	• Able to make service keep up with the fast evolution of products because call flow authors have access to a state-of-the-art authoring tool. • Less expensive to train new CSRs because the new runtime UI is easy to use.
CSRs	• Poor productivity due to poor toolset • Lack of job satisfaction	• Far more productive • Lower turnover rate: likely to stay longer due to a far more user/worker-friendly toolset
Customers	• Poor service	• Problems are solved in shorter time • No need to deal with disgruntled CSRs

Service systems should adapt. Service systems should be adaptive in the short term, exhibiting resiliency and agility by adapting to fluctuations in demand or usage patterns. Service systems should also be adaptive over the long term, becoming more efficient and effective, utilizing feedback from within and outside the system to guide adaptations [Sp06]. The ability to update the call flows and rules that help the CSRs help the customers can be considered a kind of adaptability. In the original system, updating call flows and rules was difficult and potentially risky. After CCF was deployed, updating call flows and rules using a tool with built-in semantic and syntactic checking was a major advance in the ease with which the system could adapt to changing business and demand characteristics. These changes allowed the business process owners to more easily make error free changes to rules, and activation of rule updates did not affect service. In addition the new system allows for the addition of self-learning components which can automatically optimize the call flows, and possible automatically generate new call flows.

Service systems should account for people and technology related costs from a provider and client perspective. In this case study, we've considered the cost of the CCF solution not only in terms of the hardware and labor required to deploy and maintain the system, but also in terms of training service providers and clients. On the service provider side, CSRs had to learn a new web-based runtime client for getting the guidance for helping their customers, and call flow authors had to learn to how to change call flows and rules using the new Eclipse-based authoring tool. On the client side, there were no changes imposed by CCF. The customers who call the call center for help did not have to change anything or learn anything new.

In service systems, the value is co-produced by the service provider and service client during production of the service. The work to improve the service took place on the service provider side of the equation, with the client not being required to learn anything new or change the way they were asking for help. Thus, the nature of the co-production relationship in terms of inputs into the system was not changed from the clients' perspective. Because there is a cost to the service client in changing, and any changes that become cumbersome or increase work might result in the client defecting to another provider, the ability to improve service by changing the provider side of the service system is extremely valuable. In this case, the client sees only that the customer service system is more responsive and reliable; likely improving their satisfaction and potentially buying more services and products from the provider (effectively growing demand).

Service systems should become more efficient by standardizing client inputs. The unified services theory states that a service involves a provider and client working together to transform the clients inputs during performance of the service [Sa01]. Service efficiencies can be realized by standardizing client inputs. In this case study, the front stage of the call center is a way of standardizing how clients provide their support information. The introduction of CCF did not change anything in terms of what was required of the client. Rather, it added an extra layer of standardization (through the use of SOA components that allowed the UIs, algorithms, processes, and workflows to be easily interchangeable) which is transparent to the clients complete their part of the co-production relationship.

Service providers should use self-service and automation technologies to lower cost and improve service. Although the traditional telephone-based front stage of the call center is not a self-service system because it requires a CSR to answer the customer's call, the CCF framework makes automation possible because one can develop any customer-friendly runtime applications without deep knowledge of other components in the solution, such as legacy and third-party back-end and the authoring tool. The automation would lower the costs that are associated with managing CSRs. In addition there is the possibility now of automating parts of the call flow optimization and generation process which would allow for further improvements in service and at the same time lower costs further.

Service systems should scale to greater service capacity at declining costs to the service provider. Service systems that require equal increases in labor to achieve equivalent growth in service capacity do not yield increasing profits. CCF contributes to lowering costs while enabling capacity to increase by decoupling application development from process management and thus enhancing process independence. As the provider becomes responsible for servicing more products, only the call flows (and not the applications) need to be changed, and due to the visual nature of call flow definitions in the authoring tool it is simple (and thus cheap) to update the call flows since the task requires no programming skills.

CCF also contributes towards lowering costs by allowing reuse of the call flow information in other (lower cost) channels such as web self help, instant message/chat, and self-diagnostic features built in products. This is possible due to the flexible nature of call flows in this architecture. By keeping call flows loosely coupled from the client UI used to display them, a wide variety of client types may be used with the same set of call flows, enabling a high degree of reuse.

Ideally, service systems should combine technology, business, and social innovation to create new business models. A common example of a business model that could not exist without technology and social innovation is eBay. The buyer and seller reputation systems rely on internet technology to exist, and the social innovation that results is people buying goods from people they've never met in person. In this case study, CCF has not resulted in a new business model; rather it has made an existing model more efficient, i.e. resulted in a business process transformation.

A service system should appear to be customized to the customer to the degree that it is equivalent to cost. A service customer will reasonably expect to pay more for a customized service. At first glance, CCF apparently did not change the customer's perception of the degree their support service was customized. However, since this is a globally deployed call center, customers from different locales have different preferences, such as language. The old call management system is tied to an English-only encoding scheme. The CCF framework allows developers to use any character encoding scheme, and allows process owners to use any language to describe the processes.

What is even more interesting is that even within the same locale, such as the United States, different customers of the same product have different vocabularies that refer to equivalent concepts with respect to a call flow. For example, both Bank of A and B-Mart have bought DB2 products from the provider. What Bank of A calls a "branch" is equivalent to what B-Mart calls a "store" as far as the call flow (for a DB2 product) is concerned. The CCF framework allows the process owner to design a single base call flow for the product, and allows a runtime engine to intelligently decide the proper customization based on information acquired on the fly. The CCF framework lowers the costs of customization by facilitating asset reuse, such as sharing the base call flow, and runtime polymorphism.

A service system should provide evidence to the client that a service has or is being performed. While the service itself is often evidence enough, service satisfaction usually benefits from multiple forms of evidence. The evidence informs and reinforces to the client that a service has been or is being performed. Evidence can also convey a sense of value. When a customer calls the call center, a service ticket is open. When the customer's problem has been solved, the ticket is closed.

A service system should support transparency to the degree that is enhances value for the service client and preserves value for the service provider. Transparency is a way for service providers to share production information with the service client. It might also be considered a kind of evidence that the service is being performed. When a customer asks for help, they receive a ticket number to track the case. This allows her to see the progress of the service until she is satisfied. The transparency is also bidirectional. The original call management system already had adequate transparency, and there was no business need to change that. By preserving the same experience for the customer in the CCF solution, we have maintained the same level of transparency in the customer/provider relationship.

6 Conclusions

We have described a call management service system, the problems, and strategies used to address the problems. We then analyzed the system in terms of a service system framework.

As a result of this service engagement, the CCF research team that provided technology and expertise to CM has produced a system that exceeded expectations. With the CCF framework, business and technical agility is achieved by process independence. This has also lowered the costs and improved the productivity of the service system.

In summary, the research and CM teams that developed and deployed the CCF solution achieved the above results by:

1) Encapsulating certain complexity in the CCF framework allowing both business analysts and application developers to work productively and independently. Therefore:

    a. The call management system becomes agile enough to take advantage of advanced technologies and response to new business requirement efficiently in the long run.

    b. CM can strategically off-shore resources without introducing much of the unnecessary communication overheads.

2) Providing state-of-the-art tooling for updating call flows and business rules, including syntactic and semantic checking.

3) Providing superior asset reuse and customization of call flows.

Additional benefits, although less tangible but arguably just as important, include potentially improving client satisfaction, increasing demand, and lowering client defection rates. While additional interventions as described in the analysis might result in even better overall system performance, cost benefit analysis should precede any such steps.

This call center service system was dramatically improved by CCF. However, this is just one example of where CCF has provided value. Other domains that could potentially benefit from CCF include sales and distribution, finance, software self-healing and self-configuring capabilities, and healthcare response and information systems.

7 Bibliography

[Ab06] ABLE Rule Language, User's Guide and Reference, Version 2.3.0. ABLE Project Team, IBM T. J. Watson Research Center (2006).

[CBK06] Cheng, I., Boyette, N., Krishna, V. Towards a Low-Cost High-Quality Service Call Architecture. IEEE International Conference on Services Computing – SCC (2006) 261-264. http://doi.ieeecomputersociety.org/10.1109/SCC.2006.106

[Ec06] EMF: Eclipse Modeling Framework (2006). http://www.eclipse.org/emf/

[Ga02] Gadrey, J. & Gallouj, F. Productivity, Innovation and Knowledge in Services, New Economic and Socio-Economic Approaches. Cheltenham, UK: Edward Elgar (2002).

[Ma06] Maglio, P. P., Srinivasan, S., Kreulen, J. T., and Spohrer, J. 2006. Service Systems, Service Scientists, SSME, and Innovation. Communications of the ACM 49, 7 (Jul. 2006), 81-85. http://doi.acm.org/10.1145/1139922.1139955

[Sa01] Sampson, S.E. Understanding service businesses: Applying principles of Unified Services Theory (2nd ed.). New York: Wiley (2001).

[Sp06] Spohrer, J., Maglio, P., P., Bailey, J., Gruhl, D. Steps Toward a Science of Service Systems. Computer, Vol.40, Iss.1, (Jan. 2007). 71-77.

[Te06] Teboul, J. Service Is Front Stage: Positioning Services for Value Advantage. Insead Business Press. Palgrave Macmillan (2006).