Generalized Service Process Expressed by Context-Free Grammar

Fumihiro Maruyama

Abstract

This paper generalizes processes for various services and defines a generalized service process (GSP). The first half presents GSP in two processes: one on the customer side and the other on the provider side, expressed by context-free grammar with generic terms. The second half proposes a GSP-based system and its applications. A procedure is outlined for specializing GSP by reducing the production rules of GSP. Examples of specialization representing individual service processes are also illustrated. Finally, the effectiveness of the methodology is discussed in terms of how it helps us better understand individual services and share and reuse best practices and knowledge.

Keywords

Service modeling • Service process • Context-free grammar • Generalization • Specialization

1 Introduction

Due to their diversity, it is difficult to compare and analyze services across the boundaries of industries and/or business categories. One approach to this problem is to position individual services in a spectrum. The goods-service spectrum [1] is a spectrum of tangible and intangible parts. Individual services can be compared and analyzed by their position between one extreme, 100 % tangible, and the other, 100 % intangible. As servitization advances in terms of (1) from goods-centric to customer outcome-centric services or (2) from basic to advanced services, the intangible part accounts for a higher proportion. The mind-mechanism spectrum [2], on the other hand, is a spectrum of human service-oriented mind and mechanisms supporting the service through devices, instruction manuals, IT systems, and organizational structures. Mind-dominant hospitality is positioned near one extreme and mechanism-dominant self-service near the other. Qualitative comparison and analysis is possible through the approach of using a spectrum.

However, it is also difficult to share and reuse best practices and knowledge such as policies, metrics, insights, know-how, and ideas when industries or business categories are different. Another approach is to focus on a common feature of services. The objective of this research is to develop a methodology for comparing and analyzing various services and sharing and reusing best practices and knowledge by focusing on a common feature of services.

Since service can be defined as "any activity that one economic entity (called a *service provider*) does for another (called a *customer*) where value is cocreated by the two," the process of services can be regarded as an important common feature of services.

This paper generalizes processes for various services and defines a generalized service process (GSP) with the following four characteristics:

(a) It consists of two processes: one on the customer side and the other on the provider side.

F. Maruyama (✉)
Fujitsu Laboratories Ltd., 4-1-1 Kamikodanaka, Nakahara-ku, Kawasaki 211-8588, Japan
e-mail: maruyama.f@jp.fujitsu.com

© Springer Japan 2017

Y. Sawatani et al. (eds.), *Serviceology for Smart Service System*, DOI 10.1007/978-4-431-56074-6_12

(b) Both processes are expressed by context-free grammar (CFG). CFG allows us to express hierarchical and repetitive processes succinctly.
(c) There are points of contact between the two processes where interaction between the customer and the provider takes place. It is this interaction that leads to co-creation of value.
(d) The processes on the customer and provider sides for an individual service are expressed by a specialization of the GSP grammar.

After related work is reviewed and CFG is introduced, GSP is presented in two processes: one on the customer side and the other on the provider side, expressed by CFG with generic terms. Next, a GSP-based system and its applications are proposed. A procedure is outlined for specializing GSP by reducing the original set of production rules of GSP. Examples of specialization representing individual service processes are also illustrated. Finally, the effectiveness of the methodology is discussed in terms of how it helps us better understand individual services and share and reuse best practices and knowledge.

2 Related Work

The objective of the MIT Process Handbook Project [3] is to create a systematic and powerful method of organizing and sharing business knowledge. There is a publicly available online knowledge base developed by the project, which includes a set of representative templates and specific case examples [4]. As its name suggests, the project focuses on process activities. All process activities are considered to be specialized types of "act." The first level of specialization below "act" contains eight "generic verbs," that is, "create," "destroy," "modify," "preserve," "combine," "separate," "decide," and "manage." Process activities "sell by mail order" and "sell in retail store" are specializations of the generic sales process activity of "sell product." Specialization continues until process activities become specific enough, such as "sell using customized sales channel," where Dell Computer's case is attached. In addition to "generalization" and "specialization," there are two more attributes: "parts" and "uses." This is how the MIT Process Handbook organizes business cases and knowledge. Although process activities have their sub-activities as their "parts," there is no explicit representation of process flow. There is neither alternative order of activities nor repetitive activities. In other words, the MIT Process Handbook does not deal with processes as a time series of actions or operations.

Even though object-oriented analysis and design methodologies take full advantage of the object specialization hierarchy, the process specialization hierarchy is not supported in major process representations such as the state diagram. From this perspective, an approach to process specialization is proposed in the form of a set of transformations which, when applied to a process description, always results in specialization [5]. It concerns the specialization relationship between individual processes. This paper, on the other hand, assumes the most general service process, GSP, in such a way that individual service processes can be obtained as its specializations.

A study exists on the sequential structure of work processes using rule-based grammatical models [6]. It deals with routine work in an organization and takes technical assistance work provided by a software vendor as an example. Although its approach is bottom-up and it seems difficult to generalize it to services in general, the following four points that [6] points out also apply to this paper:

1. A grammar does not specify a fixed outcome; it defines a set of possibilities.
2. Grammatical models can capture the layered quality of action.
3. Grammatical models are well suited to representing dependencies between events that may be widely separated in an observed sequence.
4. Grammatical models have potential practical value because they provide a clear way to distinguish normatively correct instances of a routine from other instances.

There is an attempt to define a general process flow for services in order to take an accurate measurement of productivity and customer satisfaction of services [7]. The top-level process flow for service providers consists of "proposal," "preparation," "serving customers," "offer," and "after-sales service." There is another process flow for service recipients (customers) to use services, and there are points of contact between the two process flows.

This paper shares the objective of [3], expands the attempt of [7] on the basis of CFG grammar in the same way that [6] uses rule-based grammatical models, and proposes a GSP. While [5] applies a set of transformations to a process description such as a state diagram, restrictions are placed on the GSP grammar to obtain a specialization for an individual service process.

3 Context-Free Grammar (CFG)

CFG [8] is a formal grammar in which every production rule is of the form

$$V \rightarrow w$$

where V is a nonterminal symbol, or a variable, and w is a string of terminal symbols and/or nonterminal symbols (w can be empty denoted by ε). It is called "context-free" because its production rules can be applied regardless of the context of a nonterminal symbol. No matter which symbols surround it, the single nonterminal symbol on the left-hand side can always be replaced by the right-hand side.

CFG is used in linguistics to describe the structure of sentences and words in natural language. It is also used in computer science to define a programming language or a document type.

There is a special nonterminal symbol called the start variable, which is used to represent the whole sentence or program. Rule application to the start variable and repetitive rule application to the resulting strings eventually give us a string of terminal symbols, which is a valid sentence or program. The set of all the possible strings (all the valid sentences or programs) generated by a grammar is called the language of the grammar.

When CFG is used for service processes, a string of symbols represents a time series of processes from left to right in chronological order. A production rule breaks down a process into a time series of subprocesses. A production rule of the form

$$P \rightarrow w' P$$

where P is a nonterminal symbol and w' is a string of terminal symbols and/or nonterminal symbols, is called recursive. The above recursive rule means that the process denoted by P (process P) can carry out a time series of subprocesses denoted by w' first and then carry out process P. Recursive rules enable us to express repetitive processes.

This paper deals with specialization of CFG. A CFG is said to be a specialization of another CFG if the language generated by the former is a subset of the language generated by the latter. For example, if some of the production rules of a CFG are removed with the other elements (start variable, nonterminal, and terminal symbols) remaining intact, then the language generated by the resulting CFG is a subset of the language generated by the original CFG, which means the resulting CFG is a specialization of the original CFG.

The following design policies were set for expressing GSP by CFG:

- Use a generic vocabulary independent of industries and business categories. Detailed processes dependent on a specific industry or business category are out of scope.
- Introduce proper hierarchies.
- Express repetitive processes explicitly.
- Production rules can be redundant. Clear meaning of production rules is more important than avoiding redundancies.

4 Generalized Service Process (GSP)

In GSP expressed by CFG, symbols starting with a capital letter represent variables, and symbols in lowercase letters represent terminal symbols. A vertical bar represents logical disjunction and allows us to express multiple rules sharing a variable on the left-hand side in a single rule by connecting their right-hand sides like $X \rightarrow v \mid w\,X \mid \varepsilon$. "$\varepsilon$" stands for the empty string.

In the following GSP, S_c and S_p are the start variables and represent the whole processes on the customer side and the provider side, respectively. When a symbol, which represents a process, is used by both sides, it is distinguished by a subscript "$_c$" for the customer side and "$_p$" for the provider side.

4.1 Customer-Side GSP

The following is the customer-side GSP expressed by CFG:

$S_c \rightarrow$ recognize Explore_provider Explore_item Consume Settle$_c$ Attitude
  | recognize Explore_item Explore_provider Consume Settle$_c$ Attitude
Explore_provider $\rightarrow$ select_provider
  | select_provider visit_provider | abandon_provider
  | browse_provider Explore_provider
  | search_provider Explore_provider
  | get_estimate Explore_provider | ε
Explore_item $\rightarrow$ select_item | abandon_item
  | browse_item Explore_item | search_item Explore_item
  | get_estimate Explore_item | sample$_c$ Explore_item | ε
Consume $\rightarrow$ Procure Appreciate
  | Procure Appreciate Explore_item Consume
  | Procure Appreciate Explore_provider Consume
  | Procure Appreciate Explore_provider Explore_item Consume
  | Procure Appreciate Explore_item Explore_provider Consume | ε
Procure $\rightarrow$ reserve | pay | place_order | contract$_c$
  | request | informed | reserve Procure | pay Procure
  | place_order Procure | contract$_c$ Procure
  | request Procure | informed Procure
Appreciate $\rightarrow$ receive_item Evaluate
  | prepare$_c$ receive_item Evaluate
  | receive_item Evaluate Appreciate
  | prepare$_c$ receive_item Evaluate Appreciate | Evaluate
Evaluate $\rightarrow$ satisfied | fair | dissatisfied | unsuitable
  | reject | check Evaluate | inquire Evaluate
  | request Evaluate | dispute$_c$ Evaluate | informed Evaluate

$\texttt{Settle}_\texttt{c} \rightarrow \texttt{pay} \mid \texttt{dispose}_\texttt{c} \mid \texttt{return}_\texttt{c} \mid \texttt{obtain\_refund} \mid \texttt{feedback} \mid \texttt{Evaluate} \mid \texttt{pay Settle}_\texttt{c} \mid \texttt{dispose}_\texttt{c}\ \texttt{Settle}_\texttt{c} \mid \texttt{return}_\texttt{c}\ \texttt{Settle}_\texttt{c} \mid \texttt{obtain\_refund Settle}_\texttt{c} \mid \texttt{feedback Settle}_\texttt{c} \mid \varepsilon$

$\texttt{Attitude} \rightarrow \texttt{loyal} \mid \texttt{disloyal} \mid \texttt{neutral} \mid \texttt{support Attitude} \mid \texttt{dissuade Attitude} \mid \varepsilon$

The meanings of the processes appearing on the right-hand side of the two rules for S_c are as follows:

- The customer recognizes his/her own needs in `recognize`.
- He/she explores service providers and selects one in `Explore_provider`.
- He/she explores service items and selects one or a set of service items in `Explore_item`.
- He/she consumes the service item(s) provided by the provider in `Consume`.
- He/she settles with the provider in $\texttt{Settle}_\texttt{c}$.
- `Attitude` represents what attitude he/she takes after consuming the service item(s).

The customer basically controls the customer-side process, but the process is influenced by the provider in some cases.

4.2 Provider-Side GSP

The following is the provider-side GSP expressed by CFG:

$S_\mathrm{p} \rightarrow \texttt{Arouse Engage Greet Propose Provide Settle}_\texttt{p}\ \texttt{Behavioral\_review} \mid \texttt{Arouse Propose Engage Greet Provide Settle}_\texttt{p}\ \texttt{Behavioral\_review}$

$\texttt{Arouse} \rightarrow \texttt{merchandise} \mid \texttt{promote} \mid \texttt{merchandise Arouse} \mid \texttt{promote Arouse} \mid \varepsilon$

$\texttt{Engage} \rightarrow \texttt{publicize} \mid \texttt{advertise} \mid \texttt{contact} \mid \texttt{give\_estimate} \mid \texttt{guide} \mid \texttt{publicize Engage} \mid \texttt{advertise Engage} \mid \texttt{contact Engage} \mid \texttt{give\_estimate Engage} \mid \texttt{guide Engage} \mid \varepsilon$

$\texttt{Greet} \rightarrow \texttt{welcome} \mid \varepsilon$

$\texttt{Propose} \rightarrow \texttt{advertise} \mid \texttt{recommend} \mid \texttt{give\_estimate} \mid \texttt{sample}_\texttt{p} \mid \texttt{advertise Propose} \mid \texttt{recommend Propose} \mid \texttt{give\_estimate Propose} \mid \texttt{sample}_\texttt{p}\ \texttt{Propose} \mid \varepsilon$

$\texttt{Provide} \rightarrow \texttt{Close-deal Deliver} \mid \texttt{Close-deal Deliver Propose Provide} \mid \texttt{Close-deal Deliver Engage Greet Provide} \mid \texttt{Close-deal Deliver Engage Greet Propose Provide} \mid \texttt{Close-deal Deliver Propose Engage Greet Provide} \mid \varepsilon$

$\texttt{Close-deal} \rightarrow \texttt{accept\_reservation} \mid \texttt{receive\_payment} \mid \texttt{accept\_order} \mid \texttt{contract}_\texttt{p} \mid \texttt{respond} \mid \texttt{decline} \mid \texttt{accept\_reservation Close-deal} \mid \texttt{receive\_payment Close-deal} \mid \texttt{accept\_order Close-deal} \mid \texttt{contract}_\texttt{p}\ \texttt{Close-deal} \mid \texttt{respond Close-deal} \mid \varepsilon$

$\texttt{Deliver} \rightarrow \texttt{Arrange fulfill Assist} \mid \texttt{Arrange fulfill Assist Deliver} \mid \varepsilon$

$\texttt{Arrange} \rightarrow \texttt{prepare}_\texttt{p}\ \texttt{Assist} \mid \texttt{prepare}_\texttt{p}\ \texttt{Assist Arrange} \mid \varepsilon$

$\texttt{Assist} \rightarrow \texttt{redo} \mid \texttt{respond} \mid \texttt{instruct} \mid \texttt{offer} \mid \texttt{observe} \mid \texttt{dispute}_\texttt{p} \mid \texttt{report} \mid \texttt{respond Assist} \mid \texttt{instruct Assist} \mid \texttt{offer Assist} \mid \texttt{observe Assist} \mid \texttt{dispute}_\texttt{p}\ \texttt{Assist} \mid \texttt{report Assist} \mid \varepsilon$

$\texttt{Settle}_\texttt{p} \rightarrow \texttt{receive\_payment} \mid \texttt{dispose}_\texttt{p} \mid \texttt{return}_\texttt{p} \mid \texttt{refund} \mid \texttt{survey} \mid \texttt{observe} \mid \texttt{farewell} \mid \texttt{bill Settle}_\texttt{p} \mid \texttt{receive\_payment Settle}_\texttt{p} \mid \texttt{dispose}_\texttt{p}\ \texttt{Settle}_\texttt{p} \mid \texttt{return}_\texttt{p}\ \texttt{Settle}_\texttt{p} \mid \texttt{refund Settle}_\texttt{p} \mid \texttt{survey Settle}_\texttt{p} \mid \texttt{observe Settle}_\texttt{p} \mid \texttt{farewell Settle}_\texttt{p} \mid \varepsilon$

$\texttt{Behavioral\_review} \rightarrow \texttt{research} \mid \texttt{analyze} \mid \texttt{research Behavioral\_review} \mid \texttt{analyze Behavioral\_review} \mid \varepsilon$

The meanings of the processes appearing on the right-hand side of the two rules for S_p are as follows:

- The provider arouses the customer's needs in `Arouse`.
- He/she tries to establish relationships with the customer in `Engage`.
- He/she greets the customer in `Greet`.
- He/she proposes service items to the customer in `Propose`.
- He/she provides the service item(s) that the customer selects in `Provide`.
- He/she settles with the customer in $\texttt{Settle}_\texttt{p}$.
- He/she reviews the customer's attitude in `Behavioral_review`.

4.3 Points of Contact

The following is the list of points of contact between the customer-side and provider-side processes where interaction between the customer and the provider takes place.

"$\Rightarrow$" indicates that the process on the left-hand side works on the process on the right. In other words, when either the customer or the provider starts the process on the left, the process on the right is requested or induced. Strictly speaking, while the process on the right is required to start

in some cases, it is expected to start but not mandatory in other cases. "⇔" indicates that the process on either side can start first and work on the other process. A process is distinguished by its superior process shown in the succeeding parentheses if it is used in multiple processes.

```
Arouse⇒recognize
browse_provider⇒Engage
get_estimate(Explore_provider)⇒give_estimate
(Engage)
Engage⇒select_provider
guide⇒visit_provider
visit_provider⇒welcome
browse_item⇒Propose
search_item⇒Propose
get_estimate(Explore_item)⇒give_estimate(Pro-
pose)
sample_c⇒sample_p
Propose⇒select_item
reserve⇒accept_reservation
pay⇒receive_payment
place_order⇒accept_order
contract_c⇔contract_p
request(Procure)⇒respond(Close-deal)
report(Assist)⇒informed(Procure)
fulfill⇒receive_item
inquire⇒Assist
request(Evaluate)⇒respond(Assist)
report(Assist)⇒informed(Evaluate)
Evaluate(Appreciate)⇒observe(Assist)
reject⇒redo
dispute_c⇒dispute_p
bill⇒pay(Settle_c)
pay(Settle_c)⇒receive_payment
return_c⇒return_p
refund⇒obtain_refund
feedback⇔survey
Evaluate(Settle_c)⇒observe(Settle_p)
Attitude⇒Behavioral_review
```

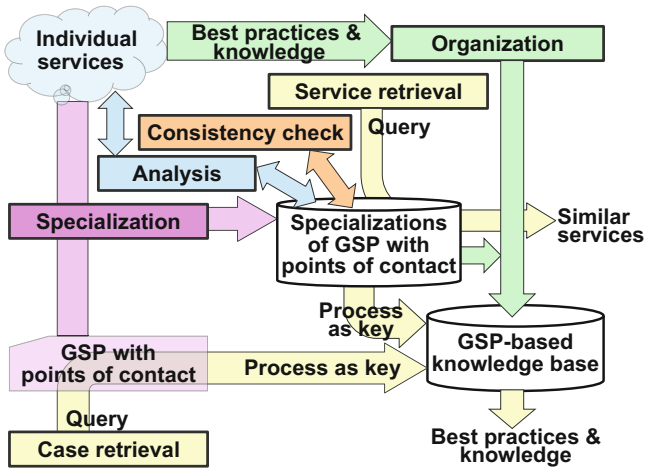

Fig. 1 Configuration of GSP-based system

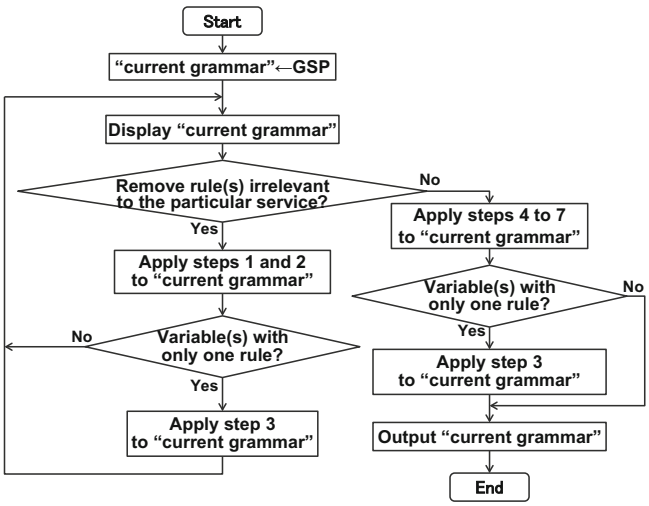

Fig. 2 Flowchart for specialization

5 GSP-Based System

Figure 1 shows the configuration of the proposed GSP-based system.

The system centers around two databases: one for specializations of GSP and the other for the GSP-based knowledge base. The rectangles in the figure represent procedures to perform on the databases.

The specialization procedure is the basis for all the other procedures. Given an individual service, it helps the user specialize GSP with respect to the service in question; there are two instances of specialization, one from the customer-side GSP and the other from the provider-side GSP. It stores the resulting specialization in the database along with relevant points of contact between the customer-side and provider-side processes. Figure 2 shows the flowchart for the specialization procedure. The "steps" mentioned in the figure are described in Sect. 5.1.

Although it is difficult to specify the grammar for an individual service from scratch, it would be easier to follow the steps in the figure to obtain the specialization of GSP for the service in question.

5.1 Specialization of GSP

The seven steps mentioned in the flowchart in Fig. 2 are as follows:

1. Remove production rules irrelevant to the service in question such as those including a process that is not carried out in the service in question.
2. Suppose the production rules for variable X include $X \rightarrow v$ and $X \rightarrow v\ X$. If it is asserted that process v is carried out only at the end of process X in the service in question, then remove $X \rightarrow v\ X$. If it is asserted that process X does not end with process v, on the other hand, then remove $X \rightarrow v$.
3. If there is only one rule for a variable, then replace its occurrences by the right-hand side of the single rule.
4. Suppose there are just two rules for variable X, $X \rightarrow$ v and $X \rightarrow$ w X. If it is asserted that processes v and w are exclusive, then remove $X \rightarrow w\ X$, apply Step 3 to the single rule of $X \rightarrow v$, and end up with the occurrences of X replaced by v. Otherwise, that is, processes v and w are compatible, then replace X by $w^*\ v$, where the asterisk indicates there is zero or more of the preceding element. In a special case of $X \rightarrow w\ X \mid \varepsilon$, replace X by w^*. In another special case of $X \rightarrow w \mid w\ X$, replace X by $w+$, where the plus sign indicates there is one or more of the preceding element. If there are just three rules for variable X, $X \rightarrow w$, $X \rightarrow w\ X$, and $X \rightarrow \varepsilon$, replace X by w^*.
5. If the right-hand side of every rule for variable X is a terminal symbol, $X \rightarrow v_i$ where v_i is a terminal symbol ($i = 1, 2, \ldots, n$), then replace X by $(v_1 \mid v_2 \mid \ldots \mid v_n)$.
6. If all the rules for variable X can be expressed as $X \rightarrow v_1 \mid v_2 \mid \ldots \mid \mathrm{v_n} \mid w\ X$ where $\mathrm{v_i}$ is a terminal symbol ($i = 1, 2, \ldots, n$), then replace X by $w^*\ (v_1 \mid v_2 \mid \ldots \mid v_n)$. If all the rules for variable X can be expressed as $X \rightarrow v_1 \mid v_2 \mid \ldots \mid v_n \mid w_1\ X \mid w_2\ X \mid \ldots \mid w_m\ X$ where w_i is a terminal symbol ($i = 1, 2, \ldots, m$), replace X by $(w_1 \mid w_2 \mid \ldots \mid \mathrm{w_m})^*\ (v_1 \mid v_2 \mid \ldots \mid v_n)$.
7. $(w_1 \mid w_2 \mid \ldots \mid \mathrm{w_m})^*$ can be further simplified by using knowledge of the service in question. If it is asserted that processes $w_1, w_2, \ldots w_m$ are carried out only once in this order, $(w_1 \mid w_2 \mid \ldots \mid w_m)^*$ can be reduced to $w_1\ w_2 \ldots w_m$ (string). $v_1^*\ (v_1 \mid v_2)$ can be reduced to $v_1 \mid (v_1\ v_2)$ if it is asserted that process v_1 is carried out only once. $v_1 \mid (v_1\ v_2)$ can be written as $v_1\ v_2?$, where the question mark indicates there is zero or one of the preceding element, v_2 in this case. The preceding element, v_2 in this case, is optional in other words.

5.2 Individual Service Processes

This section shows examples of specialization of the GSP grammar representing three individual service processes.

5.2.1 Google

The first example is Google's free service using its search engine [9]. Applying Step 1 of Sect. 5.1 to the customer-side GSP with respect to Google's search engine service leaves the following rules:

```
Sc → recognize Explore_provider Explore_item
  Consume Settlec Attitude
Explore_provider → select_provider visit_provider
Explore_item → select_item
Consume → Procure Appreciate
  | Procure Appreciate Explore_item Consume
Procure → place_order
Appreciate → receive_item Evaluate
Evaluate → satisfied | fair | dissatisfied | check
  Evaluate
Settlec → ε
Attitude → loyal | disloyal | neutral
```

Applying other steps of Sect. 5.1 to the above rules leaves the following rules for `Sc`, `Evaluate` and `Attitude`:

```
Sc → recognize select_provider visit_provider
  select_item (place_order receive_item Evaluate
  select_item)* place_order receive_item Evaluate
  Attitude
Evaluate → check* (satisfied | fair | dissatisfied)
Attitude → loyal | disloyal | neutral
```

The following is the specialization of the customer-side GSP grammar with all the above rules combined:

```
Sc → recognize select_provider visit_provider
select_item {place_order receive_item check*
(satisfied | fair | dissatisfied) select_item}*
place_order receive_item check* (satisfied |
fair | dissatisfied)(loyal | disloyal | neutral)
```
(1)

What the processes appearing in (1) mean in this context are as follows:

`recognize`: The customer wants to look up something.
`select_provider`: Selects the Google search engine.
`visit_provider`: Visits the Google site.
`select_item`: Selects a set of keywords.
`place_order`: Inputs the keywords.
`receive_item`: Receives search results.
`check`: Checks them by clicking links.
`satisfied`: Feels satisfied.
`fair`: Feels the results are fair.
`dissatisfied`: Feels dissatisfied.
`loyal`: Becomes a loyal customer.

disloyal: Becomes a disloyal customer.
neutral: Remains neutral.

Applying Step 1 of Sect. 5.1 to the provider-side GSP with respect to Google's search engine service leaves the following rules:

$S_p \rightarrow$ Arouse Engage Greet Propose Provide $Settle_p$ Behavioral_review
Arouse $\rightarrow \varepsilon$
Engage $\rightarrow \varepsilon$
Greet $\rightarrow$ welcome
Propose $\rightarrow$ recommend
Provide $\rightarrow$ Close-deal Deliver
| Close-deal Deliver Propose Provide
Close-deal $\rightarrow$ accept_order
Deliver $\rightarrow$ $prepare_p$ fulfill Assist
Assist $\rightarrow$ observe
$Settle_p \rightarrow \varepsilon$
Behavioral_review $\rightarrow$ analyze

Applying other steps of Sect. 5.1 to the above rules leaves the following specialization of the provider-side GSP grammar:

$S_p \rightarrow$ welcome recommend (accept_order $prepare_p$ fulfill observe recommend)* accept_order $prepare_p$ fulfill observe analyze (2)

What the processes appearing in (2) mean in this context are as follows:

welcome: The provider welcomes the customer.
recommend: Recommends keywords.
accept_order: Accepts keywords as input.
$prepare_p$: Executes the search engine.
fulfill: Displays search results.
observe: Accumulates customer's behavior.
analyze: Analyzes customer's search history.

The points of contact between the customer-side and provider-side processes for Google's search engine service are shown in Fig. 3.

5.2.2 QB House

QB House provides a no-frills rapid haircutting service at a reasonable price [10]. Applying Step 1 of Sect. 5.1 to the customer-side GSP with respect to QB House's haircutting service leaves the following rules:

$S_c \rightarrow$ recognize Explore_provider Explore_item Consume $Settle_c$ Attitude
Explore_provider $\rightarrow$ select_provider visit_provider
Explore_item $\rightarrow \varepsilon$
Consume $\rightarrow$ Procure Appreciate
Procure $\rightarrow$ request | pay Procure
Appreciate $\rightarrow$ receive_item Evaluate
| receive_item Evaluate Appreciate
Evaluate $\rightarrow$ satisfied | fair | dissatisfied | request Evaluate
$Settle_c \rightarrow \varepsilon$
Attitude $\rightarrow$ loyal | disloyal | neutral

Applying other steps of Sect. 5.1 to the above rules leaves the following specialization of the customer-side GSP grammar:

$S_c \rightarrow$ recognize select_provider visit_provider pay* request {receive_item request* (satisfied | fair | dissatisfied)}+ (loyal | disloyal | neutral)

If it is asserted that process pay is carried out only once, then pay* can be reduced to pay as follows:

$S_c \rightarrow$ recognize select_provider visit_provider pay request {receive_item request* (satisfied | fair | dissatisfied)}+ (loyal | disloyal | neutral) (3)

The customer-side process (3) illustrates the following scenario:

(i) The customer wants to have his/her hair cut.
(ii) The customer selects QB House.
(iii) The customer visits a QB House outlet.
(iv) The customer purchases a ticket from a vending machine.

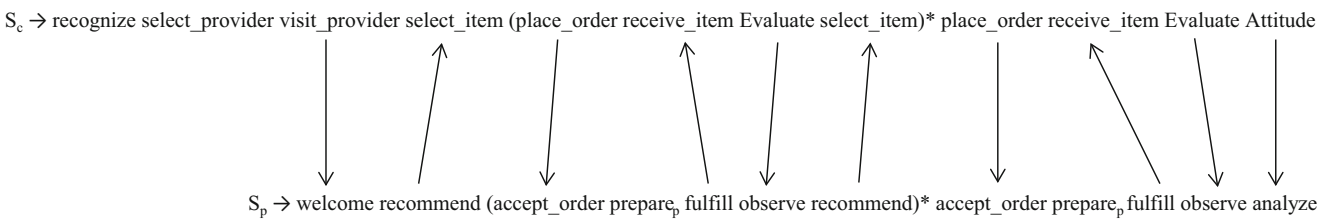

Fig. 3 Points of contact for Google's search engine service

(v) The customer makes a request for the hairstyle he/she wants.
(vi) The customer has his/her hair cut, makes an additional request, if any, and evaluates (satisfied/fair/dissatisfied). Step vi can be repeated.
(vii) The customer leaves the outlet and finds himself/herself a loyal/disloyal/neutral customer.

Applying Step 1 of Sect. 5.1 to the provider-side GSP with respect to QB House's haircutting service leaves the following rules:

```
S_p → Arouse Engage Greet Propose Provide Settle_p
  Behavioral_review
Arouse → ε
Engage → publicize | advertise | publicize Engage
  | advertise Engage
Greet → welcome
Propose → ε
Provide → Close-deal Deliver
Close-deal → receive_payment | respond
  | receive_payment Close-deal
Deliver → prepare_p fulfill Assist
  | prepare_p fulfill Assist Deliver
Assist → respond | respond Assist | ε
Settle_p → dispose_p
Behavioral_review → analyze
```

Applying other steps of Sect. 5.1 to the above rules leaves the following specialization of the provider-side GSP grammar:

```
S_p → (publicize | advertise)+ welcome
  receive_payment* (receive_payment | respond)
  (prepare_p fulfill respond*)+ dispose_p analyze
```

If it is asserted that process `receive_payment` and the first process of `respond` are carried out only once, then the rule can be reduced to the following:

```
S_p → (publicize | advertise)+ welcome
  receive_payment respond (prepare_p fulfill
  respond*)+ dispose_p analyze                    (4)
```

The provider-side process (4) illustrates the following scenario:

(i) The provider invites a customer by publicizing and/or advertising including indicating the expected waiting time with a signal.
(ii) The provider welcomes the customer at the outlet.
(iii) The provider sells a ticket with a vending machine.
(iv) The provider accommodates the request from the customer.
(v) The provider prepares, cuts the customer's hair, and meets an additional request, if any. Step v can be repeated.
(vi) The provider cleans up and finishes.
(vii) The provider analyzes customers' responses.

5.2.3 Typical Restaurant

The last example is about the service provided by a typical restaurant serving dinner. Applying Step 1 of Sect. 5.1 to the customer-side GSP with respect to dining at a typical restaurant leaves the following rules:

```
S_c → recognize Explore_provider Explore_item
Consume Settle_c Attitude
Explore_provider → select_provider visit_provider
  | browse_provider Explore_provider
  | search_provider Explore_provider
Explore_item → select_item | browse_item Explore_item
Consume → Procure Appreciate
  | Procure Appreciate Explore_item Consume
Procure → place_order | request | place_order Procure
Appreciate → receive_item Evaluate
  | receive_item Evaluate Appreciate
Evaluate → satisfied | fair | dissatisfied
Settle_c → pay
Attitude → loyal | disloyal | neutral | support
  Attitude | dissuade Attitude
```

The following is the specialization of the customer-side GSP grammar with all the remaining rules combined:

```
S_c → recognize (browse_provider | search_provider)*
  select_provider visit_provider browse_item*
  select_item [place_order* (place_order |
  request) {receive_item (satisfied | fair |
  dissatisfied)}+ browse_item* select_item]*
  place_order* (place_order | request)
  {receive_item (satisfied | fair | dissatisfied)}
  + pay (support | dissuade)* (loyal | disloyal |
  neutral)
```

If it is asserted that process `place_order` is carried out only once, then the rule can be reduced to the following:

```
S_c → recognize (browse_provider | search_
provider)* select_provider visit_provider
browse_item* select_item [place_order request?
{receive_item (satisfied | fair | dissatisfied)}+
browse_item* select_item]* place_order request?
{receive_item (satisfied | fair | dissatisfied)}
```

```
+ pay (support | dissuade)* (loyal | disloyal |
neutral)                                        (5)
```

The customer-side process (5) covers the following scenario:

(i) The customer wants to dine at a restaurant.
(ii) The customer may browse and/or search restaurants.
(iii) The customer selects a restaurant.
(iv) The customer visits the restaurant.
(v) The customer may browse a menu.
(vi) The customer selects dishes and/or a drink.
(vii) The customer orders them and makes an additional request, if any.
(viii) The customer receives them and evaluates (satisfied/fair/dissatisfied). Step viii can be repeated.
(ix) The customer may browse a menu again; select additional dishes and drink; order them; make an additional request, if any; receive them; and evaluate. This step is optional and can be repeated.
(x) The customer pays the bill and leaves the restaurant.
(xi) The customer may support the restaurant or speak ill of it.
(xii) The customer finds himself/herself a loyal/disloyal/neutral customer.

Applying Step 1 of Sect. 5.1 to the provider-side GSP with respect to the typical restaurant service leaves the following rules:

```
S_p → Arouse Engage Greet Propose Provide Settle_p
   Behavioral_review
Arouse → ε
Engage → publicize | advertise | publicize Engage
  | advertise Engage
Greet → welcome
Propose → recommend
Provide → Close-deal Deliver
  | Close-deal Deliver Propose Provide
Close-deal → accept_order | respond | accept_order
   Close-deal
Deliver → prepare_p fulfill Assist
  | prepare_p fulfill Assist Deliver
Assist → observe
Settle_p → dispose_p | bill Settle_p | receive_payment
   Settle_p | farewell Settle_p
Behavioral_review → analyze
```

The following is the specialization of the provider-side GSP grammar with all the remaining rules combined:

```
S_p → (publicize | advertise)+ welcome recommend
   {accept_order* (accept_order | respond)
   (prepare_p fulfill observe)+ recommend}*
   accept_order* (accept_order | respond)
   (prepare_p fulfill observe)+ (bill |
   receive_payment | farewell)* dispose_p analyze
```

If it is asserted that process `accept_order` is carried out only once, then `accept_order* (accept_order | respond)` can be reduced to `accept_order respond?`. If it is asserted that processes `bill`, `receive_payment`, and `farewell` are carried out only once in this order, then `(bill | receive_payment | farewell)*` can be reduced to `bill receive_payment farewell`.

```
S_p → (publicize | advertise)+ welcome recommend
   {accept_order respond? (prepare_p fulfill
   observe)+ recommend}* accept_order respond?
   (prepare_p fulfill observe)+ bill
   receive_payment farewelldispose_p analyze
                                               (6)
```

The provider-side process (6) covers the following scenario:

(i) The provider invites the customer by publicizing and/or advertising.
(ii) The provider welcomes the customer at the entrance and seats him/her at the table.
(iii) The provider recommends some dishes including today's special.
(iv) The provider accepts an order and responds to an additional request, if any.
(v) The provider prepares dishes, takes them to the table, and observes his/her reactions. Step v can be repeated.
(vi) The provider may accept an additional order; respond to an additional request, if any; prepare dishes; take them to the table; and observe. This step is optional and can be repeated.
(vii) The provider bills the customer, receives the payment, and says goodbye.
(viii) The provider cleans up the table.
(ix) The provider analyzes customers' responses.

6 GSP-Based Applications

6.1 GSP-Based Analysis

The three procedures of analysis, consistency check, and service retrieval in Fig. 1 can be used for analysis.

The analysis procedure is directly responsible for analysis. It takes data from an individual service and analyzes it based on the specialization of GSP for the service in question. Important indicators include time, such as the time required and the waiting time, and the number of repetitions of processes.

The following notation is useful for time. |←P Q→| denotes the time required from the start of process P to the end of process Q. On the other hand, P|←→|Q denotes the time required from the end of process P to the start of process Q, and |←P→|Q denotes the time required from the start of process P to the start of process Q. There may be other processes between P and Q. In the example of Google's search engine service, |←`place_order`→|`receive_item` denotes the waiting time for search results. In the example of dining at a typical restaurant, |←`place_order`→|`receive_item` denotes the waiting time for dishes. In the example of QB House's haircutting service, |←`pay`→|`receive_item` denotes the waiting time for the haircut. The waiting time is generalized to |←`Procure`→|`Appreciate` using the hierarchy of GSP. It covers the waiting time for the three examples and allows us to compare the waiting time among various services. This is an example of generalizing indicators based on the hierarchy of GSP for comparing a wider range of services.

The number of repetitions of processes is associated with the parts attached "*" or "+" in the specialization of GSP. In the example of Google, the number of repetitions of

```
{place_order receive_item check*
(satisfied | fair | dissatisfied)
select_item}*                    (7)
```

indicates the number of additional searches. In the example of dining at a typical restaurant, the number of repetitions of

```
[place_order request? {receive_item
(satisfied|fair|dissatisfied)}+browse_item*
select_item]*                    (8)
```

indicates the number of additional orders. The number of repetitions of

```
{receive_item (satisfied | fair |
dissatisfied)}+                  (9)
```

within (8) indicates the number of times dishes are brought to the table.

Since GSP includes processes such as "`abandon_provider`," "`abandon_item`," and "`reject`," abnormal processes can be expressed. It is possible to compare abnormal processes with normal ones for an individual service. It is also possible to compare abnormal processes with those of other services in terms of the "dropout" rate and so on. It is advisable to share and reuse best practices in other services with a lower "dropout" rate.

The consistency check procedure checks if there is any inconsistency in the specialization of GSP for an individual service. Inconsistency occurs when either process of a pair of points of contact in Sect. 4.3 is missing, which means that an action on one side is not adequately supported or responded to by the other side. Resolving inconsistency can improve the service in question. For example, if there is no process under the provider's "`Propose`," it would be worthwhile coming up with a process to work on the customer's "`select_item`" process by retrieving best practices in other services as described in Sect. 6.2. Inconsistency also occurs when the orders of processes on both sides are not corresponding.

The service retrieval procedure can calculate similarity between processes by matching specializations of the GSP grammar. Given an individual service, it returns services whose processes are similar to those of the service in question. Similar services may have priority over other services as to where to look for best practices and knowledge as described in Sect. 6.2.

6.2 GSP-Based Sharing and Reuse

The two procedures, organization and case retrieval, in Fig. 1 are responsible for sharing and reuse of best practices and knowledge such as policies, metrics, insights, know-how, and ideas.

The organization procedure takes best practices and knowledge in an individual service and stores them in the GSP-based knowledge base with the corresponding process attached.

The case retrieval procedure retrieves best practices and knowledge stored in the GSP-based knowledge base by using a process as a key. The user can be either in a special context with an individual service in mind or in a general context. In the latter case, he/she can use any process in GSP as a key. In the former case, he/she is expected to specify either a process appearing in the specialization of GSP for the service in question or a process that is currently missing but necessary for improving the service as illustrated in the example of "`Propose`" in Sect. 6.1.

There are two ways to extend a key in order to extend the scope of retrieval when the number of appropriate results is too few. One way is to use points of contact. "`select_item`" as a key can be extended by adding "`Propose`," which works on "`select_item`." Recommendation of keywords by Google and recommendation of products by Amazon through collaborative filtering may be found in the best practices with "`Propose`" attached. Another way to extend a key is to use the hierarchy of GSP. "`advertise`" as a key can be extended by adding its superior process "`Engage`," which includes processes

such as "publicize" and "guide." In order to narrow down a search, on the other hand, similar services returned by the service retrieval procedure described in Sect. 6.1 are given priority; best practices and knowledge associated with similar services are retrieved first.

What is expected to be shared and reused includes metrics in information retrieval and policies involving IT. Metrics in information retrieval such as precision, i.e., the fraction of retrieved instances that are relevant, and recall, i.e., the fraction of relevant instances that are retrieved, can be used outside of information services. IT becomes indispensable in some aspects of services and helps improve the quality of services when it is properly introduced. The GSP-based methodology allows us to share and reuse policies and metrics related to IT by retrieving best practices and knowledge from seemingly quite different services.

Among the points of contact between the customer-side and provider-side processes shown in Sect. 4.3 is Evaluate (Appreciate)⇒observe(Assist). The process of "observe" is important because it can find out how the customer evaluated the service item. In the example of Google, how the customer clicked the links in the search results can be observed. In an investigation of a cafeteria service, the amount of leftovers was observed. It is valuable to share and reuse best practices on how to observe the customer's behavior and how to assess the customer's evaluation. The GSP-based methodology allows us to share and reuse policies and metrics for the process of "observe."

A survey of the cafeteria service showed that the time required for the settlement process, which is clearing away the dishes, affects customer satisfaction. Since the customer has already consumed the service before the settlement process, taking a long time for settlement makes him/her feel worse. It is also the case with the settlement process at a hotel, that is, checkout. This is why some hotels introduce express checkout or advance payment, which is also good for reducing the staff or allowing them more time to meet customers' various needs around the busiest time of checkout. In general, best practices for $\mathtt{Settle}_c$ or $\mathtt{Settle}_p$ are very helpful. The proposed methodology helps share and reuse best practices for those processes among services in different industries or business categories.

6.3 GSP-Based Service Design

As a new application, there is a possibility of using GSP along with points of contact for designing a new service. An expected process on the customer side is first defined by specializing the customer-side GSP in Sect. 4.1. The overall design is to generate a process on the provider side that is consistent with the above specialization for the customer side. The consistency check procedure described in Sect. 6.1 can be used as a subroutine. More detailed design may use the case retrieval procedure described in Sect. 6.2 for referring to best practices in other services.

Assuming a customer-side process is important to the service provider in the first place. It is a key to providing a good service. It is also important for the service provider to update the customer-side process when he/she collects data about the customers' behavior. He/she might want to define multiple customer-side processes according to customer segmentation.

6.4 Discussion

As mentioned in the Introduction, the intangible part in the goods-service spectrum accounts for a higher proportion as servitization advances. The higher proportion the intangible part accounts for, the more important role GSP will play, because the intangible part is basically realized by processes.

GSP is thought to be able to capture the sequential structure of service processes. But capturing only the sequential structure due to the use of grammar is its limitation. Some work is done in parallel in services, especially in the back office; however, the main focus of the methodology is the sequential structure of service processes like [6]. While work done in parallel has been studied extensively in manufacturing, service is characterized by its interaction between the customer and the provider, and this interaction can be basically serialized.

Although the more detailed processes become, the harder it is to share them among services, it is possible to go deeper than the level of GSPs defined in Sects. 4.1 and 4.2. If it is useful to define additional more detailed processes, it is possible to do so without sharing them with other services.

7 Conclusion and Future Work

GSP is presented in two processes, one on the customer side and the other on the provider side, expressed by CFG with generic terms. Then a GSP-based system and its applications are proposed. A procedure is outlined for specializing GSP by reducing the original set of production rules of GSP. Examples of specialization representing individual service processes are also illustrated. Finally, the effectiveness of the methodology is discussed in terms of how it helps us better understand individual services and share and reuse best practices and knowledge.

GSP can be regarded as a kind of ontology [11], an explicit specification of a conceptualization which aims to support the sharing and reuse of formally represented knowledge. In the context of this paper, what is formally represented is not knowledge to be shared and reused but the structure of processes. GSP can be regarded as an ontology with definitions of process structure such as chronological order and repetition of processes in addition to hierarchical structure.

As part of our future work, the plan is to validate the practicality of the methodology and to implement its useful applications.

References

1. Berry, L. L., Parasuraman, A., 1991, Marketing Services: Competing through Quality, The Free Press, New York, NY.
2. Maruyama, F., Kohda, Y., Katsuyama, T., 2007, Issues of Service Innovation and Its Model, IEEE Joint Conference on E-Commerce Technology and Enterprise Computing, E-Commerce and E-Services, Tokyo, Japan, 23–26 July, 489–490.
3. Malone, T. W. et al., 2003, Tools for Inventing Organizations: Toward a Handbook of Organizational Processes, Organizing Business Knowledge: The MIT Process Handbook, 13–38.
4. MIT Process Handbook's website http://process.mit.edu/
5. Wyner, G. M., Lee, J., 2002, Process Specialization: Defining Specialization for State Diagrams, Computational & Mathematical Organization Theory, Volume 8, Issue 2, 133–155.
6. Pentland, B. T., Rueter, H. H., 1994, Organizational Routines as Grammars of Action, Administrative Science Quarterly, Vol. 39, No. 3, 484–510.
7. Yoshimoto, K., Saito, A., Mihara, K., Tango, S., Okubo, H., Suzuki, H., Yamaguchi, M., 2013, A Technique for Service Visualization, Journal of the Institute of Electronics, Information and Communication Engineers, Vol. 96, No. 3, pp. 610-615. (in Japanese)
8. Hopcroft, J. E., Ullman, J. D., 1979, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Reading, MA.
9. Google's search engine https://www.google.com/
10. QB House's website http://www.qbhouse.com/
11. Gruber, T. R., 1993, A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition, Vol.5, No.2, 199–220