Chapter 8
A Platform Identification Method for Service Family Design Using a Process Model and a Clustering Method

Seung Ki Moon[1], Timothy W. Simpson[2], Jun Shu[3], and Soundar R.T. Kumara[4]

Seung Ki Moon
Postdoctoral Research Associate
Department of Mechanical Engineering
The Texas A&M University,
College Station, TX 77843, USA
moonsky@gmail.com

Timothy W. Simpson
Professor
Departments of Mechanical & Nuclear Engineering and Industrial & Manufacturing Engineering
The Pennsylvania State University
University Park, PA 16802, USA
tws8@psu.edu

Jun Shu
Assistant Professor
Department of Supply Chain and Information System, the SMEAL College of Business,
The Pennsylvania State University,
University Park, PA 16802, USA
junshu@psu.edu

Soundar R.T. Kumara
Allen E., and Allen M., Pearce Professor
Department of Industrial & Manufacturing Engineering
The Pennsylvania State University
University Park, PA 16802, USA
skumara@psu.edu

[1] Seung Ki Moon is currently a postdoctoral research associate of Mechanical Engineering at Texas A&M University. He received his PhD in industrial engineering from the Pennsylvania State University, University Park in 2008 and his BS and MS degrees in industrial engineering from Hanyang University, South Korea, in 1992 and 1995, respectively. He was a Senior Research Engineer at the Hyundai Motor Company, South Korea. His research interests focus on family and platform design for products and services, universal design, strategic and multidiscipline design optimization, agent-based decision-making; engineering knowledge engineering; and intelligent information system and management.

Abstract The objective in this research is to introduce a method for identifying a service platform along with variant and unique modules to create a service family by integrating object-oriented concepts, ontologies, and data mining techniques. A service process model is introduced to describe a service based on a sequence using a graph model and object-oriented concepts. Fuzzy clustering is employed to partition service processes into subsets to identify common modules – the platform – and specific modules for the given service family. To demonstrate the proposed method, we apply it to select a platform for a family of banking services.

Abbreviations

CN Customer need
FCM Function-component matrix
FPM Function-process matrix

---

[2] Timothy W. Simpson is a professor of mechanical and industrial engineering and engineering design at the Pennsylvania State University. He received his PhD and MS degrees in mechanical engineering from Georgia Tech in 1998 and 1995 and his BS in mechanical engineering from Cornell University in 1994. He is the Director of the Learning Factory (www.lf.psu.edu) and the Product Realization Minor at Penn State. His research interests include product family and product platform design, mass customization, and data visualization to support complex systems design. He is an active member of ASME, AIAA, and ASEE. He is the Chair of the AIAA Multidisciplinary Design Optimization (MDO) Technical Committee and the Past Chair of the ASME Design Automation Executive Committee.

[3] Jun Shu is an assistant professor of supply chain and information systems at the Smeal College of Business Administration at Penn State University. He received his Bachelor degree in information and computer science from the University of California at Irvine and his PhD in industrial engineering and operations research from the University of California at Berkeley. In between degrees he worked at MCI's Consulting Division, Rockwell Semiconductors, and Cisco systems. He has served as a consultant for various companies including startups and established companies. He did his PhD research at the Department of Electrical Engineering and Computer Science at Berkeley in the area of communications networks. He is interested in the boundary-spanning research that combines knowledge from engineering, economics, and business. His current focuses include applying economic theory to the design of high-speed communications networks, RFID technologies for supply chain management, supply chain collaboration, and IT service supply chain management.

[4] Soundar R.T. Kumara is the Allen, E. and Allen, M. Pearce Chaired Professor of Industrial Engineering at The Pennsylvania State University. He holds a joint appointment with the department of Computer Science and Engineering and an affiliate appointment with the College of Information Sciences and Technology. He also serves as an Adjunct Professor of CR Rao Advanced Institute of Mathematics, Statistics and Computer Science (AIMSCS), University of Hyderabad, India. His research interests are in intelligent systems design, complex networks, and sensor networks. He has won several awards including the Penn State Engineering Society Premiere Research Award and the Penn State Faculty Scholar Medal – the highest research award at Penn State. He is also the recipient of PSU Graduate Faculty Teaching award. He is an elected Fellow of the Institute of Industrial Engineers and the International Academy of Production Engineering (CIRP).

FR Functional requirement
PC Partition coefficient
UML Unified modeling language

8.1 Introduction and Background

For mass customization, companies are increasing their efforts to reduce cost and lead-time when developing new products and services while satisfying individual customer needs. Mass customization depends on a company's ability to provide customized products or services based on economical and flexible development and production systems (da Silveira *et al.* 2001). By sharing and reusing assets such as components, processes, information, and knowledge across a family of products and services, companies can efficiently develop a set of differentiated economic offerings by improving flexibility and responsiveness of product and service development (Simpson, 2004). Product family design is a way to achieve cost-effective mass customization by allowing highly differentiated products to be developed from a common platform while targeting products to distinct market segments (Shooter *et al.* 2005).

A product family is a group of related products based on a product platform (Simpson *et al.* 2005). A product platform is the set of features, components or subsystems that remain constant from product to product, within a given product family. A successful product family depends on how well the trade-off between the economic benefits and performance losses incurred from having a shared platform are managed. For instance, high levels of commonality decrease interface and component costs while increasing customers' preference loss.

Services are an important source of revenue for many companies, since products can be paired with additional services to satisfy customers' needs, differentiate product offerings, and remain competitive in today's market. Service science research seeks to improve the productivity and quality of service by creating new innovations, facilitating business management, and applying practical applications (Hidaka 2006). Recently, theories and methodologies for mass-customized products are being applied to service development (Jiao *et al.* 2003), and the concept of product family design, in particular, provides good solutions to various customized service industries (Peters and Saidin 2000, Meyer and Detore 2001, Jiao *et al.* 2003). For example, in the IBM Malaysia service unit, modularization of the scope of work and processes has been applied to service level design for mass customization (Peters and Saidin 2000). Lincoln Re used platform concepts to develop new insurance services (Meyer and Detore, 2001). In this chapter, we extend concepts from platform-based product families to create a new approach for module-based service family design.

The objective in this research is to introduce a method to identify a service platform along with variant and unique modules in a service family by integrating object-oriented concepts, ontologies, data mining techniques, and fuzzy set theory.

Object-oriented concepts provide service analysis tools for describing a business process or a workflow process in a service (Arlow and Neustadt 2002, Hoffer *et al.* 2006). A function-process matrix is used to identify the relationships between the service functions and the service processes that are offered as part of a service. An ontology is applied to define properties that consist of attributes and behaviors for representing a service in a service hierarchical structure. A service process model is introduced to describe a service based on a sequence using a graph model and object-oriented concepts.

Data mining can be used to help identify customer needs, to find relationships between customer needs and functional requirements, and to cluster products based on functional similarity to facilitate modular design (Braha 2001). Fuzzy c-means clustering (FCM) (Bezdek 1981) is employed to partition service processes into subsets to identify a platform and modules in a given service family. The clustering results provide membership values that represent the corresponding membership level of each cluster, which can be considered as the degree of similarity among process features. Fuzzy set theory (Zadeh 1965) is used to determine platform levels that represent the membership values.

The remainder of this chapter is organized as follows. Section 8.2 describes the proposed method for identifying a service platform and service modules. Section 8.3 gives a case study using a family of banking services. Closing remarks and future work are presented in Section 8.4.

8.2 Method for Service Module and Platform Identification

To develop customized services, we propose the following definitions for service family design:

1. A service family is a set of services based on a service platform, facilitating mass customization by promoting customer value and providing a variety of services for different market segments cost-effectively.
2. A service platform is a common basis that consists of processes, activities, objects, and/or features that are shared and remain constant from service to service, within a given service family.
3. A service module is a set of service components for performing a service.
4. A service component is regarded as an activity to satisfy certain services, which are defined by a set of processes, operations, people, objects, and/or features.

These definitions provide a foundation for modeling customized families of services. Based on these definitions, we extend concepts from platform-based product family design to develop a module-based service family. A service platform consists of common service modules that are defined as service components representing functions and processes. Based on the service platform, we can create a variety of services and families of services for satisfying various market segments depending on service-related design factors such as location, facility

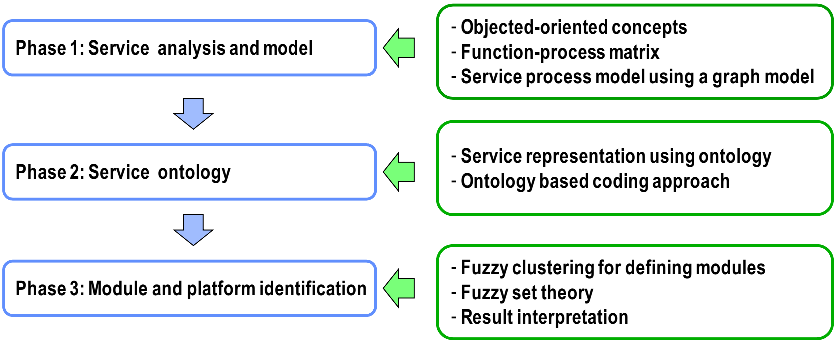

Figure 8.1 Proposed method for service module and platform identification

design and layout for effective customer and work flow, procedures and job definitions for service providers, measures to ensure quality, extent of customer involvement, equipment selection, and adequate service capacity (Fitzsimmons and Fitzsimmons 2004).

In this chapter, we introduce a method for identifying a platform along with variant and unique modules in service families using object-oriented concepts, ontologies, and data mining. Figure 8.1 shows the flow diagram of the proposed method that consists of three phases: (1) service analysis and model, (2) service ontology, and (3) module and platform identification. The next section discusses each phase of the method in detail.

8.2.1 Phase 1: Service Analysis and Model

8.2.1.1 Service Selection and Analysis

Figure 8.2 shows the process for developing a family of services based on a customer-driven approach. Information required to identify customer needs (CNs) can be collected by surveying prospective customers and by conducting a marketing study that begins by establishing target markets and customers. In the initial phase, CNs are analyzed to understand customer intention and determine a strategy for developing a service family. For example, the number of services can be decided by customer groups and classified according to CNs. CNs are also used to identify appropriate functional requirements (FRs), which are then mapped to the CNs. In service design, FRs represent processes and capabilities that can be determined by work flow, procedures and job definitions for service providers, and service quality. During conceptual design, services can be designed based on FRs, and their functional modules can also be determined. In particular, a family of services is first configured by defining a service platform. A service platform consists of several common modules that can be shared across a family of ser-

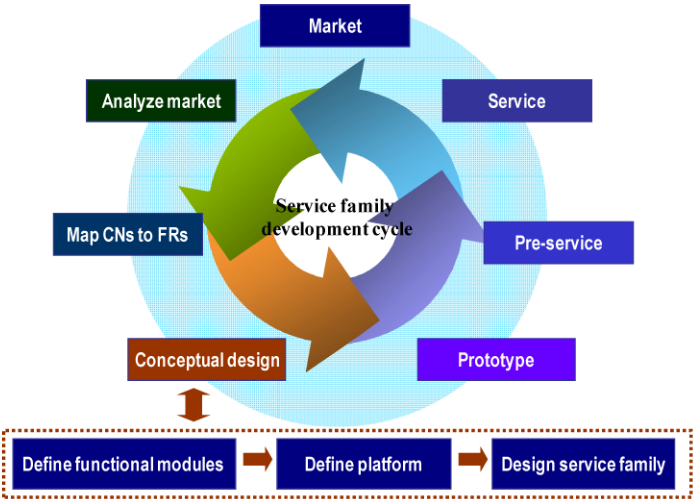

Figure 8.2 Service family design process

Table 8.1 A function-process matrix for service analysis

Process module / Functional module	Process 1	Process 2	.	.	.	Process *m*
Function 1	1					1
Function 2		1				1
.			.	.	.	1
.			.	.	.	
.			.	.	.	1
Function *n*		1				

vices. After conceptual design, through prototype and pre-service processes, final services are delivered.

Object-oriented concepts can be used to analyze service processes and identify service design factors. Object-oriented design and analysis methodologies are used to develop information systems by modeling a system as a set of objects in the area of software engineering and business (Schach 2004). Through service analysis using objected-oriented concepts, we can determine service-related design factors that are represented as processes, activities, objects, and/or features, as well as service functions and processes. These design factors are also used to define the properties of service components in service process model design.

Based on service functions and processes, a function-process matrix (FPM) is introduced to identify the relationships between functional modules and process modules in a service. The FPM is similar to the FCM (Strawbridge *et al.* 2002), which provides a mapping between a product's components and its sub-functions. Table 8.1 shows a conceptual representation of the FPM. The first vertical column shows service functions, the top horizontal row is service processes and the cells of the FPM represent the relationship between each function and process. The number "1" in a cell indicates that a relationship among a function and a process exists. For example, Function 1 in Table 8.1 entails Process 1 and Process *m* to achieve this function.

8.2.1.2 Service Process Model

A business process or workflow process is described by logically related activities to achieve a defined business goal or create value-added products or services to satisfy customer needs (Reijers 2003). In services, a process can be considered as a procedure, routine, and policy to create services, which are defined by a set of activities, ordering constraints, and data or materials used for the service activities. Unified modeling language (UML) can be used to analyze service processes and/or basic workflow. UML is a standardized specification language for system design and analysis using a set of concepts, constructs, terminology, and notation (Arlow and Neustadt 2002). For example, sequence diagrams are object-interaction diagrams that consider temporal sequencing and are useful for describing the behavior of use cases and the interaction between objects within a system (Hunt 2000). Activity diagrams provide a modeling method to represent the business and operational workflows using the detailed logic of a business rule. By analyzing the sequence diagrams or the activity diagrams for a service, we can obtain attributes and identify information flow among objects for service design. For instance, suppose that the objects of a deposit process in a banking service consist of a customer, an employee, an account, and a balance. An activity diagram for the deposit process can be represented as shown in Figure 8.3. Processes in the diagram are represented by activities and attributes for performing the service.

A process model can be defined by various languages with differences in their syntax and expressive rules (Cao *et al.* 2006). A graph model is employed to describe a service process model based on service sequences. Graphs are an abstraction developed specifically to represent relationships and consist of two distinct parts: (1) nodes and (2) edges. The nodes are things in the graph that have relationships, and the edges are pairs of nodes connected by a relationship (Berry and Linoff 1997). The encapsulation concept in object-oriented concepts reduces the complexity of representing a node in service component design. As shown in Figure 8.4, a node is defined as a service component with properties that can describe service processes, and an edge as a direction presenting information, data, and materials flow. A node can be defined by five properties: (1) activity, (2) object, (3) input flow, (4) output flow, and (5) state. Activity is a process to perform a particular service by an object and is used as the name of a node. The object represents an object performing activities using input flow in certain services. The flow includes information, data, and materials, which occur in service processes.

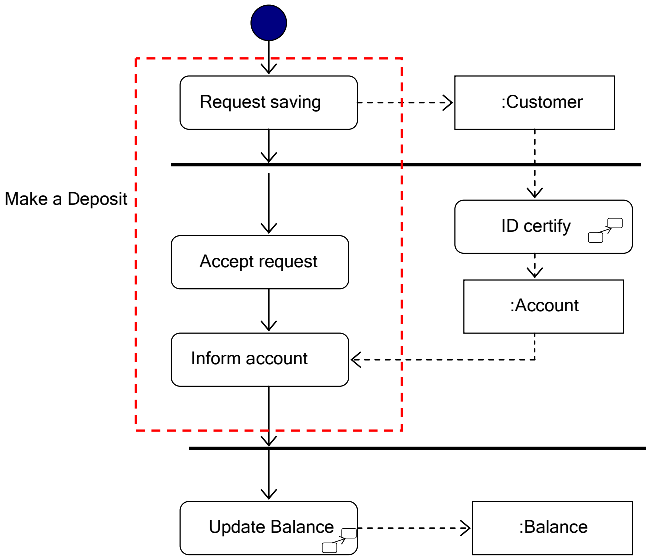

Figure 8.3 An example of an activity diagram for a banking service

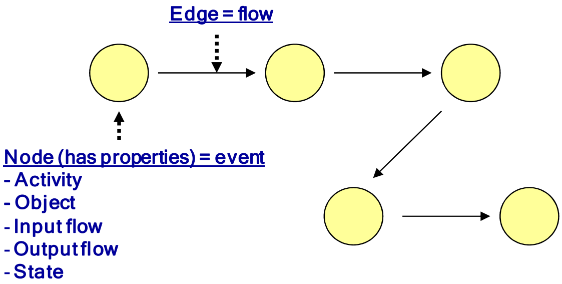

Figure 8.4 Service process model and properties for a node

States are defined as things (objects) that change the input flow and the output flow. For example, a node changes the state of its inputs (states), *i.e.,* information such as a customer's account balance or credit, materials such as money, and data in a banking service.

8.2.2 Phase 2: Service Ontology

To effectively define the relationships between functional hierarchies in a service, an appropriate representation scheme must be adopted for the services. An ontology consists of a set of concepts or terms and their relationships that describe some area of knowledge or build a representation of it (Swartout and Tate 1999).

Service ontology is developed to represent the relationships between functional modules and process modules as shown in Figure 8.5. In the service ontology, a process module has a hierarchical structure to provide process representation-based semantics of services.

The basic idea of modular design is to organize services as a set of distinct service modules that can be designed independently. Based on the concepts of the product module-based design (Kamrani and Salhieh 2000), we assume that a service can be decomposed into modules, which provide specific functions and processes. Service functions are achieved by the combination of service processes that are defined in the service ontology. Suppose that a service family consists of l services, $SF=(S_1, S_2, ..., S_l)$, and a service consists of f_i functional modules, $S_i = (y_{i,1}, y_{i,2}, ... y_{i,f}, ..., y_{i,f_i})$, where $y_{i,f}$ denotes service functional module f in service i. For service processes, suppose that a service consists of m_i service process modules, $S_i = (\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, ..., \mathbf{x}_{i,j}, ..., \mathbf{x}_{i,m_i})$, where $\mathbf{x}_{i,j}$ is process module j in service i and consists of a vector of length n_m, $\mathbf{x}_{ij} = (x_{i,j,1}, x_{i,j,2}, ..., x_{i,j,k}, ..., x_{i,j,n_m})$, and the individual scalar components $\mathbf{x}_{i,j,k} (k = 1, 2, ..., n_m)$ of a process module $\mathbf{x}_{i,j}$ are called process features. Each process feature consists of several attributes, $a_{i,j,k,t} (t = 1, 2, ..., t_n)$, representing the component, $x_{i,j,k} = (a_{i,j,k,1}, a_{i,j,k,2}, ..., a_{i,j,k,t}, ..., a_{i,j,k,t_n})$, where t_n is the number of properties defined in the service ontology. Figure 8.5 shows the corresponding hierarchy for representing a family of services. The identification of attributes is problem-dependent; an example can be found in the banking services case study. In this chapter, a coding approach is used to represent components' attributes for a given clustering method.

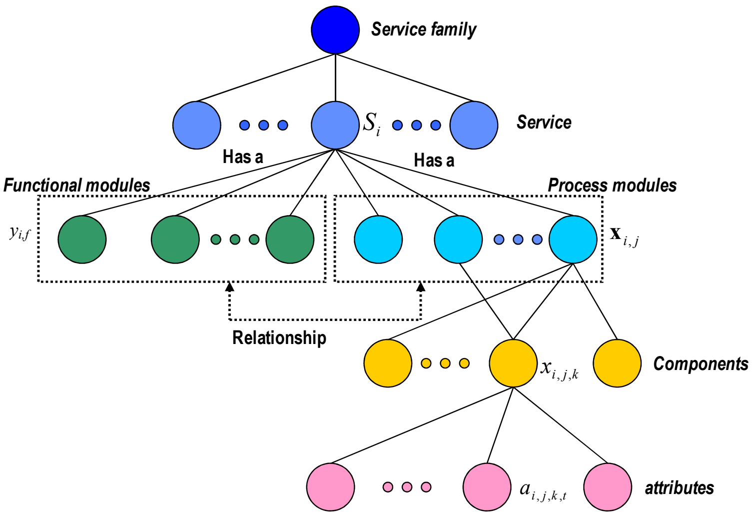

Figure 8.5 Hierarchy of the service ontology

8.2.3 Phase 3: Module and Platform Identification

8.2.3.1 Fuzzy Clustering for Defining Modules

Process decomposition for a service is often represented in a hierarchical structure as discussed in Section 8.2.2. A hierarchical clustering method can classify a set of objects by measuring the similarity between objects (Miyamoto 1990). Because heuristic methods for defining a module may provide overlapping or non-crisp boundaries among module clusters (Stone *et al.* 2000), the results of traditional clustering approaches are not appropriate to define clusters as modules in service design. Moreover, since design information for a service depends on the experience and knowledge of designers, design information, such as linguistic terms, may fail to describe a crisp representation completely. When clustering design information we need to assign the information to clusters with varying degrees of membership. Fuzzy membership can provide proper representation while also capturing the fuzziness of design knowledge (Braha 2001). Fuzzy clustering approaches can use fuzziness related to design features and provide more useful solutions (Xue and Dong 1997, Liao 2001). We employ FCM (Bezdek, 1981) to determine clusters for identifying modules for the service family. FCM is a clustering technique that is similar to k-means but uses fuzzy partitioning of data that is associated with different membership values between 0 and 1. Since FCM is an iterative algorithm, its aim is to find cluster centers that minimize a dissimilarity function.

Let $\mathbf{x}_k$ for $k = 1, 2, ...n$ be a process feature and a *d*-dimensional vector (*d* is the number of attributes), and $u_{i,k}$ the membership of $\mathbf{x}_k$ to the *i*th cluster $(i = 1, 2, ..., c)$. The $u_{i,k}$ representing a fuzzy case is between 0 and 1. For example, if $u_{i,k} = 0$, $u_{i,k}$ has non-membership to cluster *i*, and if $u_{i,k} = 1$, then it has full membership. Values between 0 and 1 indicate fractional membership. Generally, FCM is defined as the solution of the following minimization problem (Bezdek 1981):

$$J_{FCM}(U,V) = \{\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m \left\|X_k - v_i\right\|^2\} \tag{8.1}$$

subject to:

$$\sum_{i=1}^{c} u_{ik} = 1 \text{ for all } k \tag{8.2}$$

$$u_{ik} \in [0, 1] \tag{8.3}$$

where v_i is the cluster center of the *i*th cluster that consists of a *d*-dimensional vector, and *m* is a parameter $(m \geq 1)$ that indicates the fuzziness of the clusters. We use the FCM algorithm from Bezdek (1981) and Torra (2005) in this work. While this algorithm does not ensure convergence to a global optimum, it always converges to a local optimum that may lead to different local minima when using a different initial number of cluster centers.

In this FCM algorithm, since the cluster number *c* is determined before clustering, a validity index for an optimal *c* should be considered for defining the number of clusters. In this chapter, the partition coefficient (PC) is used to determine the best cluster number c (Bezdek 1974):

$$PC(c) = \frac{1}{n}\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^2 \tag{8.4}$$

where $1/c < \mathrm{PC}(c) < 1$. An optimal cluster number c^* maximizes PC(c), (the number of services + 1) < c < n–1.

The cluster number determines the number of modules. A maximum membership value in clusters is an indicator for assigning to a module that can be considered as a group of similar process features. Among clusters, clusters including the process features for all selected services become common modules for the platform.

8.2.3.2 Platform Level Determination

Since membership values from the results of clustering represent the degree of similarity among process features, we can consider the membership values as the corresponding membership level of each cluster. Based on fuzzy set theory (Zadeh 1965), membership values are measured using a rating scale of [0–1], and the ratings can be interpreted as fuzzy numbers based on different platform levels, such as low, medium, and high. Let X be a linguistic variable with the label "platform level" with U= [0, 1], and three fuzzy terms for the linguistic variable are defined as low (x_1), medium (x_2), and high (x_3) as shown in Figure 8.6. The membership function of each fuzzy set is assumed to be triangular, and the platform level can take three different linguistic terms. Platform level membership functions are proposed to represent and determine the platform level of a common module. Therefore, the membership values of functions in a common module are transferred into platform level values by the platform level membership functions. The platform level of the common module is determined by the maximum value among average membership level values for the module. For example, suppose two processes, A and B, are in a common module. If the membership values of the two processes are 0.4 and 0.6, then the platform level values of the value 0.4 are represented by 0 at high, 0.8 at middle, and 0.2 at low, while the platform level values for the 0.6 value are represented by 0.2 at high, 0.8 at middle, and 0 at low.

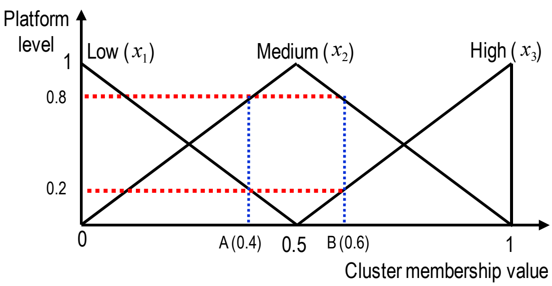

Figure 8.6 Fuzzy membership function representing platform level

Therefore, the platform level of the common module is determined as the middle level (*i.e.*, 0.1 at high, 0.8 at middle, and 0.1 at low).

8.2.3.3 Interpretation of Results

The final results determine the service platform along with the variant and unique modules for the service family, where the platform consists of common modules with a high platform level. If variant modules are selected as part of the platform, additional process features will be required to make them a common module. The service ontology is used to identify the meaning of modules using the relationship between service functions and processes. During conceptual design, these results can help decision-makers define the set of modules for the service family. The effective set of modules will lead to improved service family design. Additionally, since the proposed method uses the similarity of process features, we can evaluate the commonality of existing services by the membership values of clusters. A case study is presented next to demonstrate the proposed method.

8.3 Case Study

Consider a family of banking services consisting of four checking account services as shown in Table 8.2. The checking account services are designed for four differ-

Table 8.2 Four checking account services in a banking service family[5]

Option	Service A	Service B	Service C	Service D
Deposit	Yes	Yes	Yes	Yes
Withdraw	Yes	Yes	Yes	Yes
Transfer	Yes	Yes	Yes	Yes
Banking statement	Yes	Yes	Yes	Yes
Online account statement	Yes	Yes	Yes	Yes
Checking writing	Yes	Yes	Yes	Yes
ATM transactions	Yes	Yes	Yes	Yes
Online banking with bill pay	Yes	Yes	Yes	Yes
Telephone banking	Yes	Yes	Yes	Yes
Trade stocks online	Yes	No	Yes	Yes
Optional business economic checking	Yes	No	Yes	No
Maintenance fee	Yes	No	Yes	Yes
Additional checking and saving account	No	No	Yes	No
Loans and lines of credit	No	No	Yes	No
Service for cashier' check, and so on	No	No	Yes	No
Interest	No	No	Yes	No
Preferred rates on money market, CDs	No	No	Yes	No

[5] https://www.bankofamerica.com

ent market segments based on customers' preference, balance, credit, status, and so on. Using the proposed method, we determine a platform and a set of modules for this service family. This case study focuses on a process-based platform for the family of banking services at the conceptual stage of development.

8.3.1 Phase 1: Service Process Model

8.3.1.1 Service Selection and Analysis of the Service Family

Using service analysis, we determine the service functions and service processes in this set of four services. An FPM was developed to identify relationships between service functions and processes, as shown in Table 8.3.

Table 8.3 The function-process matrix for four checking account services

Process module / Functional module	Make a Deposit	Withdraw	Transfer Money	Trade Stocks	Check writing	Certify ID	Check Credit	Check Balance	Make a Loan	Open an Account	Record Transaction
Deposit	1					1					1
Withdraw		1				1		1			1
Transfer			1			1		1			1
Banking statement						1					1
Online account statement						1					1
Check writing		1			1	1		1			1
ATM transactions	1	1	1			1		1			1
Trade stocks online				1		1		1			1
Additional checking and saving account						1	1	1	1	1	1
Loans and lines of credit						1	1	1	1		1
Service for cashier' check, and so on		1			1	1	1	1			1
Online banking with bill pay			1			1		1			1
Interest						1	1	1	1		1
Preferred rates on Money Market, CDs						1	1	1	1		1
Telephone banking			1			1		1			1
Optional Business Economy Checking		1			1	1		1			1
Maintenance fee						1		1			1

8.3.1.2 Service Process Model

Based on the results of the service analysis, we can develop activity diagrams for service process modules to identify service processes or basic workflows as described in Phase 1. Through these activity diagrams we determine process features that are considered as the attributes of the service components in the checking account services. A service process model for a service function was developed from service process modules and service process components. For example, Figure 8.7 shows a service process model for a deposit service function that consists of three service process modules: (1) certify ID, (2) make a deposit, and (3) record transaction. The deposit process module is composed of three components: request, accept, and inform. Each service process component has five attributes as defined in Section 8.2.1.2.

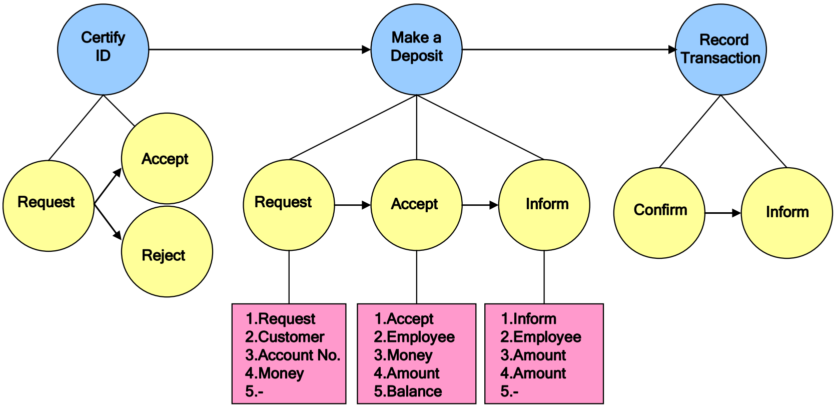

Figure 8.7 Service process model for a deposit service function

8.3.2 Phase 2: Service Ontology

The ontology for the four services was developed using Protégé[6], a graphical editing tool that has functions for developing domain ontologies, customizing the user interface, and integrating with other applications such as specific reasoning engines (Noy *et al.* 2001). Figure 8.8 shows the checking account service classes and all subclasses in Protégé. Process features in Table 8.4 are developed based on the service process analyses for the four checking account services. Each attribute takes a different code (number) related to its process feature in Table 8.4. For instance, if the attributes of a node consist of *accept* (activity), *employee* (object), *money* (input flow), *amount* (output flow), and *balance* (state), then the codes for

[6] http://protege.stanford.edu

the attributes are 1, 2, 4, 6, and 2, respectively. Process features' attributes are coded as shown in Table 8.4. Table 8.5 shows the 103 process features of the selected four services.

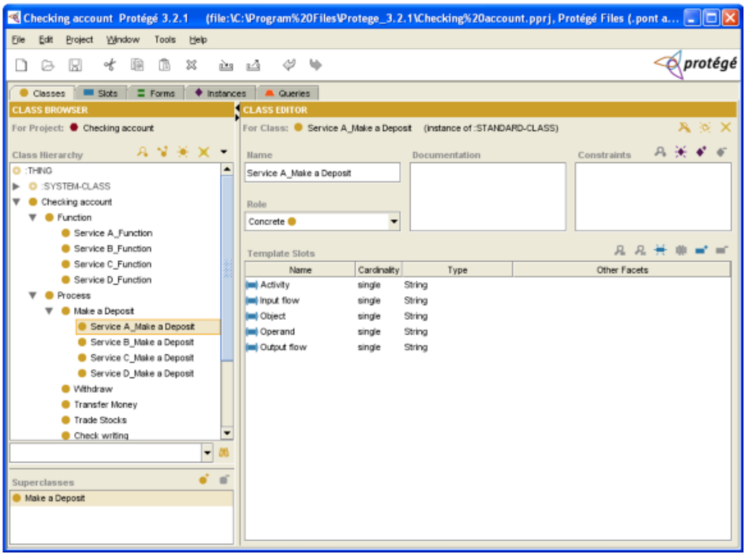

Figure 8.8 Checking account service classes and subclasses

Table 8.4 Attribute codes for process features in the four checking account services

Code	Activity	Object	Flow (contents)	State
1	Accept	Customer	Customer ID	Credit
2	Confirm	Employee	Account no.	Balance
3	Inform	Account	Credit	
4	Query	Trading (employee)	Money	
5	Request	Balance	Employee ID	
6	Reject		Amount	
7	Proposal		Balance	
8			Message	

Table 8.5 Service representation for four checking account services

			Attributes					Attribute codes				
Service	Module Process	Component	Activity	Object	Input flow	Output flow	State	Activity	Object	Input flow	Output flow	State
Service A	Make a Deposit	$X_{1,1,1}$	Request	Customer	Account No.	Money	-	5	1	2	4	0
		$X_{1,1,2}$	Accept	Employee	Money	Amount	Balance	1	2	4	6	2
		$X_{1,1,3}$	Inform	Employee	Amount	Amount	-	3	2	6	6	0
	Withdraw	$X_{1,2,1}$	Request	Customer	Account No.	Amount	-	5	1	2	6	0
		$X_{1,2,2}$	Accept	Employee	Amount	Money	Balance	1	2	6	4	2
		$X_{1,2,3}$	Inform	Employee	Amount	Amount	-	3	2	4	6	0
	Transfer Money	$X_{1,3,1}$	Request	Customer	Account No.	Amount	-	5	1	1	6	0
		$X_{1,3,2}$	Query	Employee	Amount	Amount	-	4	2	6	6	0
		$X_{1,3,3}$	Confirm	Account	Amount	Amount	Balance	2	3	6	6	2
		$X_{1,3,4}$	Inform	Employee	Amount	Message	-	3	2	6	8	0
	Trade Stocks	$X_{1,4,1}$	Request	Customer	Customer ID	Message	-	5	1	1	8	0
		$X_{1,4,2}$	Query	Employee	Message	Message	-	4	2	8	8	0
		$X_{1,4,3}$	Inform	Trading	Message	Message	-	3	4	8	8	0
	Check writing	$X_{1,5,1}$	Request	Customer	Account No.	Amount	-	5	1	2	6	0
		$X_{1,5,2}$	Conform	Employee	Amount	Amount	Balance	2	2	6	6	2
		$X_{1,5,3}$	Inform	Employee	Amount	Amount	-	3	2	6	6	0
	Certify ID	$X_{1,6,1}$	Query	Employee	Customer ID	Customer ID	-	4	2	1	1	0
		$X_{1,6,2}$	Accept	Account	Customer ID	Account No.	-	1	3	1	2	0
		$X_{1,6,3}$	Reject	Account	Customer ID	Message	-	6	3	1	8	0
	Check Credit	$X_{1,7,1}$	Query	Employee	Customer ID	Customer ID	-	4	2	1	1	0
		$X_{1,7,2}$	Inform	Account	Customer ID	Credit	-	3	3	1	3	0
	Check Balance	$X_{1,8,1}$	Query	Employee	Account No.	Account No.	-	4	2	2	2	0
		$X_{1,8,2}$	Inform	Account	Account No.	Balance	-	3	3	2	3	0
	Record Transaction	$X_{1,9,1}$	Conform	Account	Amount	Amount	Balance	2	3	6	6	2
		$X_{1,9,2}$	Inform	Account	Amount	Balance	-	3	3	6	7	0
Service B	Make a Deposit	$X_{2,1,1}$	Request	Customer	Account No.	Money	-	5	1	2	4	0
		$X_{2,1,2}$	Accept	Employee	Money	Amount	Balance	1	2	4	6	2
		$X_{2,1,3}$	Inform	Employee	Amount	Amount	-	3	2	6	6	0
	Withdraw	$X_{2,2,1}$	Request	Customer	Account No.	Amount	-	5	1	2	6	0
		$X_{2,2,2}$	Accept	Employee	Amount	Money	Balance	1	2	6	4	2
		$X_{2,2,3}$	Inform	Employee	Amount	Amount	-	3	2	4	6	0
	Transfer Money	$X_{2,3,1}$	Request	Customer	Account No.	Amount	-	5	1	1	6	0
		$X_{2,3,2}$	Query	Employee	Amount	Amount	-	4	2	6	6	0
		$X_{2,3,3}$	Confirm	Account	Amount	Amount	Balance	2	3	6	6	2
		$X_{2,3,4}$	Inform	Employee	Amount	Message	-	3	2	6	8	0
	Check writing	$X_{2,4,1}$	Request	Customer	Account No.	Amount	-	5	1	2	6	0
		$X_{2,4,2}$	Conform	Employee	Amount	Amount	Balance	2	2	6	6	2
		$X_{2,4,3}$	Inform	Employee	Amount	Amount	-	3	2	6	6	0
	Certify ID	$X_{2,5,1}$	Query	Employee	Customer ID	Customer ID	-	4	2	1	1	0
		$X_{2,5,2}$	Accept	Account	Customer ID	Account No.	-	1	3	1	2	0
		$X_{2,5,3}$	Reject	Account	Customer ID	Message	-	6	3	1	8	0
	Check Credit	$X_{2,6,1}$	Query	Employee	Customer ID	Customer ID	-	4	2	1	1	0
		$X_{2,6,2}$	Inform	Account	Customer ID	Credit	-	3	3	1	3	0
	Check Balance	$X_{2,7,1}$	Query	Employee	Account No.	Account No.	-	4	2	2	2	0
		$X_{2,7,2}$	Inform	Database	Account No.	Balance	-	3	3	2	3	0
	Record Transaction	$X_{2,8,1}$	Conform	Account	Amount	Amount	Balance	2	3	6	6	2
		$X_{2,8,2}$	Inform	Account	Amount	Balance	-	3	3	6	7	0
Service C	Make a Deposit	$X_{3,1,1}$	Request	Customer	Account No.	Money	-	5	1	2	4	0
		$X_{3,1,2}$	Accept	Employee	Money	Amount	Balance	1	2	4	6	2
		$X_{3,1,3}$	Inform	Employee	Amount	Amount	-	3	2	6	6	0
	Withdraw	$X_{3,2,1}$	Request	Customer	Account No.	Amount	-	5	1	2	6	0
		$X_{3,2,2}$	Accept	Employee	Amount	Money	Balance	1	2	6	4	2
		$X_{3,2,3}$	Inform	Employee	Amount	Amount	-	3	2	4	6	0
	Transfer Money	$X_{3,3,1}$	Request	Customer	Account No.	Amount	-	5	1	1	6	0
		$X_{3,3,2}$	Query	Employee	Amount	Amount	-	4	2	6	6	0
		$X_{3,3,3}$	Confirm	Account	Amount	Amount	Balance	2	3	6	6	2
		$X_{3,3,4}$	Inform	Employee	Amount	Message	-	3	2	6	8	0
	Trade Stocks	$X_{3,4,1}$	Request	Customer	Customer ID	Message	-	5	1	1	8	0
		$X_{3,4,2}$	Query	Employee	Message	Message	-	4	2	8	8	0
		$X_{3,4,3}$	Inform	Trading	Message	Message	-	3	4	8	8	0
	Check writing	$X_{3,5,1}$	Request	Customer	Account No.	Amount	-	5	1	2	6	0
		$X_{3,5,2}$	Conform	Employee	Amount	Amount	Balance	2	2	6	6	2
		$X_{3,5,3}$	Inform	Employee	Amount	Amount	-	3	2	6	6	0
	Certify ID	$X_{3,6,1}$	Query	Employee	Customer ID	Customer ID	-	4	2	1	1	0
		$X_{3,6,2}$	Accept	Account	Customer ID	Account No.	-	1	3	1	2	0
		$X_{3,6,3}$	Reject	Account	Customer ID	Message	-	6	3	1	8	0
	Check Credit	$X_{3,7,1}$	Query	Employee	Customer ID	Customer ID	-	4	2	1	1	0
		$X_{3,7,2}$	Inform	Account	Customer ID	Credit	-	3	3	1	3	0
	Check Balance	$X_{3,8,1}$	Query	Employee	Account No.	Account No.	-	4	2	2	2	0
		$X_{3,8,2}$	Inform	Account	Account No.	Balance	-	3	3	2	3	0
	Make a Loan	$X_{3,9,1}$	Proposal	Customer	Customer ID	Amount	-	7	1	1	6	0
		$X_{3,9,2}$	Accept	Employee	Amount	Message	Balance	1	2	6	8	2
		$X_{3,9,3}$	Reject	Employee	Amount	Message	-	6	2	6	8	0
	Open an Account	$X_{3,10,1}$	Request	Customer	Customer ID	Customer ID	-	5	1	1	1	0
		$X_{3,10,2}$	Accept	Employee	Customer ID	Account No.	-	1	2	1	2	0
		$X_{3,10,3}$	Inform	Employee	Account No.	Message	-	3	2	2	8	0
	Record Transaction	$X_{3,11,1}$	Conform	Account	Amount	Amount	Balance	2	3	6	6	2
		$X_{3,11,2}$	Inform	Account	Amount	Balance	-	3	3	6	7	0
Service D	Make a Deposit	$X_{4,1,1}$	Request	Customer	Account No.	Money	-	5	1	2	4	0
		$X_{4,1,2}$	Accept	Employee	Money	Amount	Balance	1	2	4	6	2
		$X_{4,1,3}$	Inform	Employee	Amount	Amount	-	3	2	6	6	0
	Withdraw	$X_{4,2,1}$	Request	Customer	Account No.	Amount	-	5	1	2	6	0
		$X_{4,2,2}$	Accept	Employee	Amount	Money	Balance	1	2	6	4	2
		$X_{4,2,3}$	Inform	Employee	Amount	Amount	-	3	2	4	6	0
	Transfer Money	$X_{4,3,1}$	Request	Customer	Account No.	Amount	-	5	1	1	6	0
		$X_{4,3,2}$	Query	Employee	Amount	Amount	-	4	2	6	6	0
		$X_{4,3,3}$	Confirm	Database	Amount	Amount	Balance	2	3	6	6	2
		$X_{4,3,4}$	Inform	Employee	Amount	Message	-	3	2	6	8	0
	Trade Stocks	$X_{4,4,1}$	Request	Customer	Customer ID	Message	-	5	1	1	8	0
		$X_{4,4,2}$	Query	Employee	Message	Message	-	4	2	8	8	0
		$X_{4,4,3}$	Inform	Trading	Message	Message	-	3	4	8	8	0
	Check writing	$X_{4,5,1}$	Request	Customer	Account No.	Amount	-	5	1	2	6	0
		$X_{4,5,2}$	Conform	Employee	Amount	Amount	Balance	2	2	6	6	2
		$X_{4,5,3}$	Inform	Employee	Amount	Amount	-	3	2	6	6	0
	Certify ID	$X_{4,6,1}$	Query	Employee	Customer ID	Customer ID	-	4	2	1	1	0
		$X_{4,6,2}$	Accept	Account	Customer ID	Account No.	-	1	3	1	2	0
		$X_{4,6,3}$	Reject	Account	Customer ID	Message	-	6	3	1	8	0
	Check Credit	$X_{4,7,1}$	Query	Employee	Customer ID	Customer ID	-	4	2	1	1	0
		$X_{4,7,2}$	Inform	Account	Customer ID	Credit	-	3	3	1	3	0
	Check Balance	$X_{4,8,1}$	Query	Employee	Account No.	Account No.	-	4	2	2	2	0
		$X_{4,8,2}$	Inform	Account	Account No.	Balance	-	3	3	2	3	0
	Record Transaction	$X_{4,9,1}$	Conform	Account	Amount	Amount	Balance	2	3	6	6	2
		$X_{4,9,2}$	Inform	Account	Amount	Balance	-	3	3	6	7	0

8.3.3 *Phase 3: Module and Platform Identification*

8.3.3.1 Fuzzy Clustering for Defining Modules

FCM was used to determine modules for the four checking account services. Since the number of clusters affects the number of initial modules, it is important to

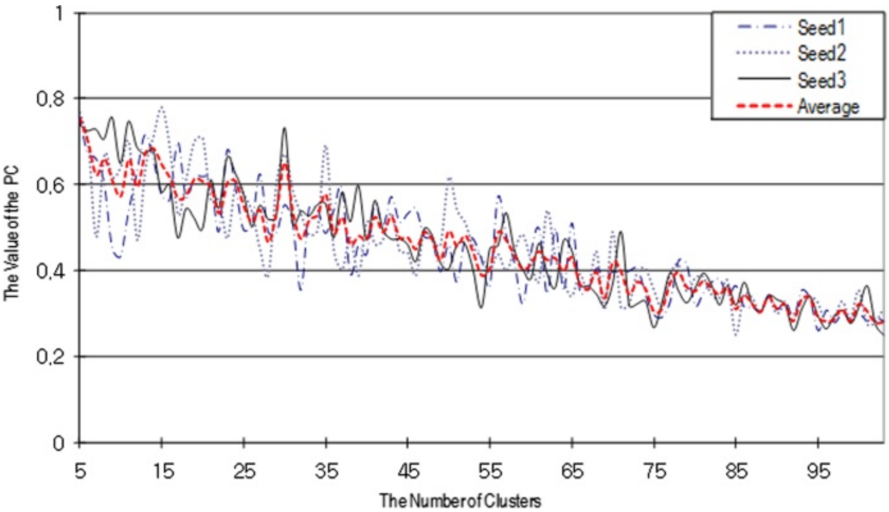

Figure 8.9 Values of the PC for three different initial seeds

Table 8.6 Clustering results for the four checking account services

Cluster	Service A	Service B	Service C	Service D
1	$X_{1,6,1}$ $X_{1,7,1}$ $X_{1,8,1}$	$X_{2,5,1}$ $X_{2,6,1}$ $X_{2,7,1}$	$X_{3,6,1}$ $X_{3,7,1}$ $X_{3,8,1}$ $X_{3,10,1}$	$X_{4,6,1}$ $X_{4,7,1}$ $X_{4,8,1}$
2	$X_{1,2,2}$ $X_{1,5,2}$ $X_{1,9,1}$	$X_{2,2,2}$ $X_{2,4,2}$ $X_{2,8,1}$	$X_{3,2,2}$ $X_{3,5,2}$ $X_{3,9,2}$ $X_{3,11,1}$	$X_{4,2,2}$ $X_{4,5,2}$ $X_{4,9,1}$
3	$X_{1,2,1}$ $X_{1,3,1}$ $X_{1,3,3}$ $X_{1,5,1}$	$X_{2,2,1}$ $X_{2,3,1}$ $X_{2,3,3}$ $X_{2,4,1}$	$X_{3,2,1}$ $X_{3,3,1}$ $X_{3,3,3}$ $X_{3,5,1}$ $X_{3,9,1}$	$X_{4,2,1}$ $X_{4,3,1}$ $X_{4,3,3}$ $X_{4,5,1}$
4	$X_{1,1,2}$	$X_{2,1,2}$	$X_{3,1,2}$	$X_{4,1,2}$
5	$X_{1,6,2}$ $X_{1,7,2}$ $X_{1,8,2}$	$X_{2,5,2}$ $X_{2,6,2}$ $X_{2,7,2}$	$X_{3,6,2}$ $X_{3,7,2}$ $X_{3,8,2}$ $X_{3,10,2}$	$X_{4,6,2}$ $X_{4,7,2}$ $X_{4,8,2}$
6	$X_{1,1,3}$ $X_{1,3,2}$ $X_{1,3,4}$ $X_{1,5,3}$ $X_{1,9,2}$	$X_{2,1,3}$ $X_{2,3,2}$ $X_{2,3,4}$ $X_{2,4,3}$ $X_{2,8,2}$	$X_{3,1,3}$ $X_{3,3,2}$ $X_{3,3,4}$ $X_{3,5,3}$ $X_{3,9,3}$ $X_{3,11,2}$	$X_{4,1,3}$ $X_{4,3,2}$ $X_{4,3,4}$ $X_{4,5,3}$ $X_{4,9,2}$
7	$X_{1,1,1}$	$X_{2,1,1}$	$X_{3,1,1}$	$X_{4,1,1}$
8	$X_{1,2,3}$	$X_{2,2,3}$	$X_{3,2,3}$ $X_{3,10,3}$	$X_{4,2,3}$
9	$X_{1,6,2}$ $X_{1,7,2}$ $X_{1,8,2}$	$X_{2,5,2}$ $X_{2,6,2}$ $X_{2,7,2}$	$X_{3,6,2}$ $X_{3,7,2}$ $X_{3,8,2}$ $X_{3,10,2}$	$X_{4,6,2}$ $X_{4,7,2}$ $X_{4,8,2}$
10	$X_{1,4,1}$ $X_{1,6,3}$	$X_{2,5,3}$	$X_{3,4,1}$ $X_{3,6,3}$	$X_{4,4,1}$ $X_{4,6,3}$
11	$X_{1,4,2}$ $X_{1,4,3}$		$X_{3,4,2}$ $X_{3,4,3}$	$X_{4,4,2}$ $X_{4,4,3}$

select the number of clusters for FCM effectively. An optimal cluster number c ($5 \le c \le 102$) was estimated by the validity index (PC) as defined in (8.4). Figure 8.9 illustrates the values of PC for three different initial seeds at fuzziness $= 1.7$ and for 10,000 iterations. In this example, $c^* = 11$ was selected as the number of clusters to determine a platform and modules for the four services, since 10 to 15 clusters provides higher average PC values than the other values. Table 8.6 shows the results of FCM using 11 clusters. Clusters that have process features for all four services can be considered as common modules.

8.3.3.2 Platform Level Determination and Result Interpretation

Using the platform level membership function described in Phase 3, the clusters' platform levels were determined as shown in Table 8.7. Since level values for Clusters 1, 3, 4, 7, and 8 indicate high platform level, these common modules can be combined into the platform for this family of four banking services.

Table 8.7 New platform and modules for the family of checking accounts

cluster	Platform level			Design
	low	middle	high	
1	0	0.2749	0.7251	Platform (Request, Query, Accept, Inform)
3	0.0039	0.1118	0.8843	
4	0	0.0114	0.9886	
7	0	0	1	
8	0.1088	0.0932	0.798	
2	0.0755	0.3711	0.5533	Module (variant and unique)
5/9	0.0903	0.9097	0	
6	0.0339	0.4048	0.5613	
10	0	0.3886	0.6114	
11	0	0.2257	0.7443	

The clusters for the suggested service platform embody a request module, a query module, an accept module, and an inform module in terms of the activities listed in Table 8.6. Therefore, the platform for the checking account services can be designed by integrating processes that are related to these activities involving a customer and an employee. Variant and unique modules can be used to increase the number of services according to customers' needs or functional requirements. The service ontology can help a designer to search for the appropriate process features related to particular service functions and processes for service design. During the conceptual stages of development, this information can provide designers with guidelines for effective service family design.

8.4 Closing Remarks and Future Work

In this chapter, we proposed a new method for identifying a service process-based platform along with variant and unique modules in a service family using object-oriented concepts, ontology, and data mining techniques. An FPM was introduced and used to identify relationships between service functions and service processes in a family of services. Object-oriented concepts were used to support service analysis and representation combining ontologies. Based on a graph model, a service process model was introduced to describe a service represented by the service ontology. Fuzzy c-means clustering was employed to cluster the process features of services based on the similarity among them and identify a service platform within the family. We demonstrated the proposed method to determine a service platform using a case study involving a family of four banking services.

The proposed method can help designers to use the newly-identified design knowledge to synthesize a platform that consists of common modules and determine a process-based platform and modules that can be adapted to service design during initial and conceptual design phase. In addition, the service design knowledge presented within an ontology can provide information and specific combinations of related modules and components based on specific constraints. It is possible that a designer can also search all of the related components in a module in service family design. Therefore, the method can help design a variety of services within a service family. Since the proposed method uses process features during clustering to determine service process modules, functional requirements for services in a family should be considered during service platform design. Future research efforts will focus on expanding the proposed method to reflect functional requirements, reusability, and configurability in platform and module design, and extending its application to various service areas and large-scale service design.

Acknowledgments This work was funded by the National Science Foundation through Grant No. IIS-0325402. Any opinions, findings, and conclusions or recommendations presented in this chapter are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Arlow J, Neustadt I (2002) UML and the Unified Process: Practical Object-Oriented Analysis and Design. Addison-Wesley, London

Berry MJA, Linoff G (1997) Data Mining Techniques: for Marketing, sales, and customer support. Wiley, New York

Bezdek JC (1974) Numerical Taxonomy with Fuzzy Sets. J of Mathematics Biology 1(1)57–71

Bezdek JC (1981) Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York

Braha D (2001) Data Mining for Design and Manufacturing: Methods and Applications. Kluwer, Dordrecht

Cao J, Wang J, Law K, Zhang S, Li M (2006) An Interactive Service Customization Model. Information and Software Technology 48(4):280–296
da Silveira G, Borenstein D, Fogliatto FS (2001) Mass Customization: literature review and research directions. International J of Production Economics 72(1):1–13
Fitzsimmons JA, Fitzsimmons MJ (2004) Service Management: Operations Strategy, and Information Technology. McGraw–Hill, New York
Hidaka K (2006) Trends in services sciences in Japan and abroad. Q Rev 19(2):35–47
Hoffer JA, George JF, Valacich JS (2006) Modern systems analysis and design. Pearson Prentice Hall, Upper Saddle River, NJ
Hunt J (2000) The Unified Process for Practitioners: Object Oriented Design, UML, and Java. Springer, London
Jiao J, Ma Q, Tseng MM (2003) Towards high value-added products and services: mass customization and beyond. Technovation 23(10):809–831
Kamrani AK, Salhieh SM (2000) Product Design for Modularity. Kluwer, Boston
Liao TW (2001) Classification and coding approaches to part family formation under a fuzzy environment. Fuzzy Sets and Systems 122(3):425–441
Meyer MH, Detore A (2001) Perspective: creating a platform-based approach for developing new services. The J of Product Innovation Management 18(3):188–204
Miyamoto S (1990) Fuzzy Sets in Information Retrieval and Cluster Analysis. Kluwer, Dordrecht
Noy NF, Sintek M, Decker S, Crubezy M, Fergerson RW, Musen MA (2001) Creating semantic web contents with Protege-2000. IEEE Intelligent Systems 16(2):61–71
Peters L, Saidin H (2000) IT and the mass customization of services: the challenge of implementation. International J of Information Management 20(2):103–119
Reijers HA (2003) Design and Control of Workflow Processes: Business Process Management for the Service Industry. Springer, Berlin
Schach SR (2004) An Introduction to Object-oriented Analysis and Design with UML and the Unified Process. McGraw–Hill/Irwin, Boston
Shooter SB, Simpson TW, Kumara SRT, Stone RB, Terpenny JP (2005) Toward an information management infrastructure for product family planning and platform customization. International J of Mass Customization 1(1):134–155
Simpson TW (2004) Product platform design and customization: status and promise. Artificial Intelligence for Engineering Des, Analysis, and Manufacturing 18(1):3–20
Simpson TW, Siddique Z, Jiao J (2005) Product Platform and Product Family Design: Methods and Applications. Springer, New York
Stone RB, Wood KL, Crawford RH (2000) A heuristic method for identifying modules for product architectures. Design Studies 21(1):5–31
Strawbridge B, McAdams DA, Stone RB (2002) A computational approach to conceptual design. ASME Design Engineering Technical Conference – Design Theory and Methodology Conference, paper No. DETC2002/DTM-34001
Swartout W, Tate A (1999) Ontologies. IEEE Transactions on Intelligent Systems 14(1):18–19
Torra V (2005) Fuzzy c-means for fuzzy hierarchical clustering. IEEE International Conference on Fuzzy Systems, Reno
Xue D, Dong Z (1997) Coding and clustering of design and manufacturing features for concurrent design. Computer in Industry 34(1):139–153
Zadeh LA (1965) Fuzzy sets. Information and Control 8:338–353