UML Application Guideline for Offshore Development

Ver. 3.0

August 2010

NPO UMTP

Offshore Software Development Working Group

Contents

1. Purpose

(1) How this guideline is positioned in the entire offshore development guideline

The offshore system development uses normal development cycles including requirement analysis, design, implementation, test and development methodology with each technical process, project management, budget planning and quality management technique the same as non-offshore local development. This guideline focuses on "offshore software development using UML" and demonstrates hints and tips based on practice and experience on offshore development and UML development.

The following explains how this guideline is positioned in the entire offshore systems development. Offshore systems development requires the following factors.

(1) Systems Development Methodology (Development processes, development procedures and development techniques)

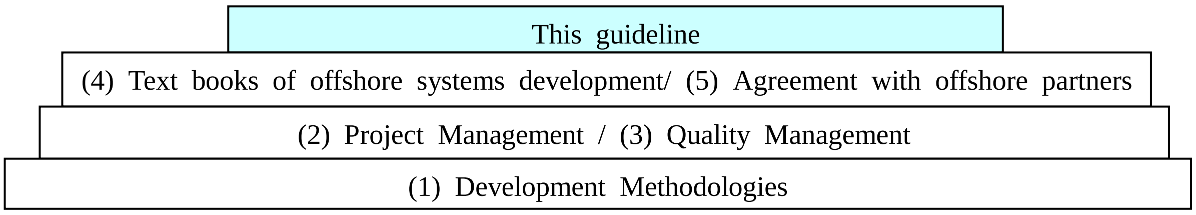
(2) Project Management (Management by such documents as project management documents)

(3) Quality Management (Management by such documents as quality management documents)

(4) Text books of offshore systems development business (Case study books of offshore systems development available in book stores, mail magazines of offshore systems development, unique expertise and know-how of the companies involved, and so on)

(5) Agreements with the offshore partners (Statement of works of each party, project schedules, deliverable and prices)

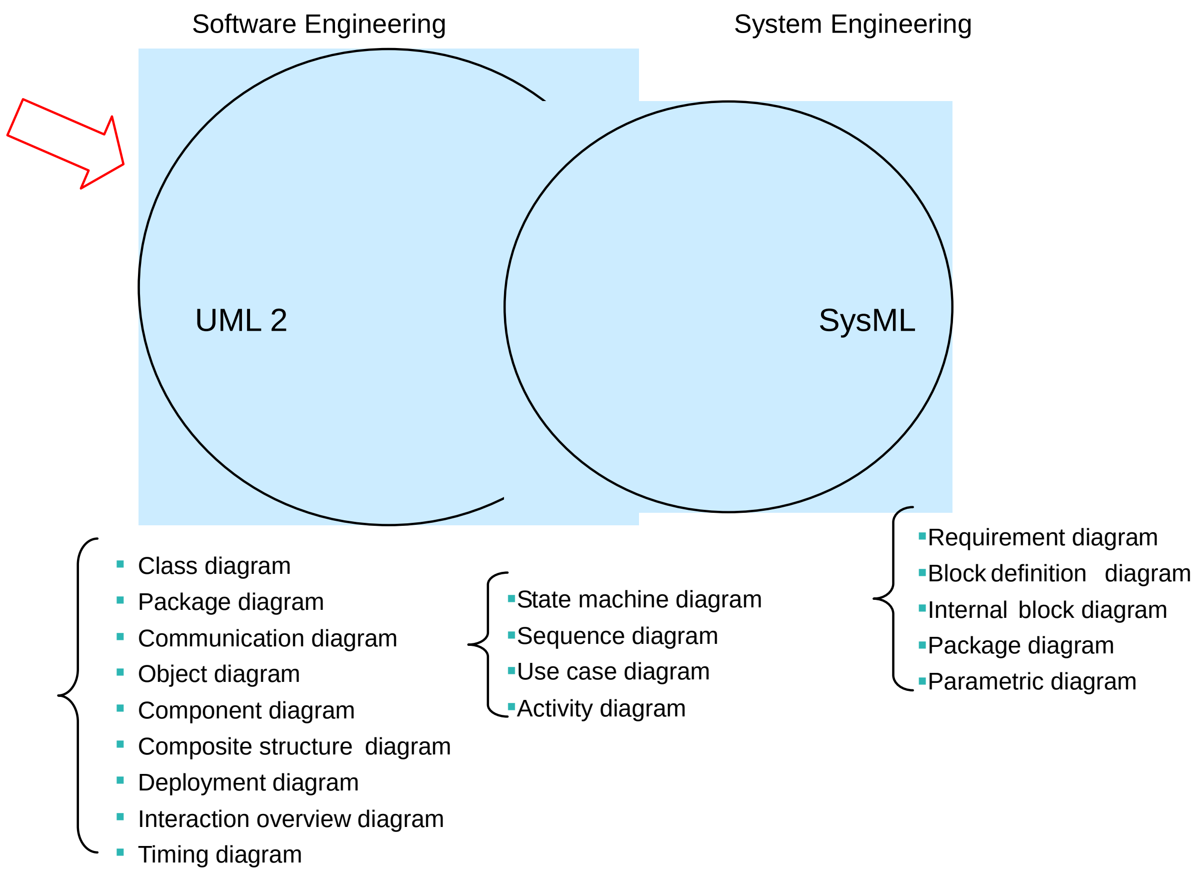
Systems development projects typically require (1) through (3) above. Offshore systems developments further require (4) and (5). Participating companies need to prepare (1) through (5) prior to the project start. This guideline is to describe some common development options, development procedures and relevant artifacts by primarily addressing to the modeling technology and is to supplement (1) through (5) as proven know-how. This document is written based on the assumption that participating companies has prepared all (1) through (5), and is intended to improve performance of offshore systems development by using it as supplemental information. Accordingly, this guideline does not address such company specific issues as scope of work, work procedures and so forth.


This guideline
(4) Text books of offshore systems development/ (5) Agreement with offshore partners
(2) Project Management / (3) Quality Management
(1) Development Methodologies

(2) What this guideline covers

This guideline covers the following areas.

“Systems Development”, in broad sense, can be categorized to software engineering using “UML2” and to system engineering using “SysML”. This guideline describes output of consideration focused on software engineering.



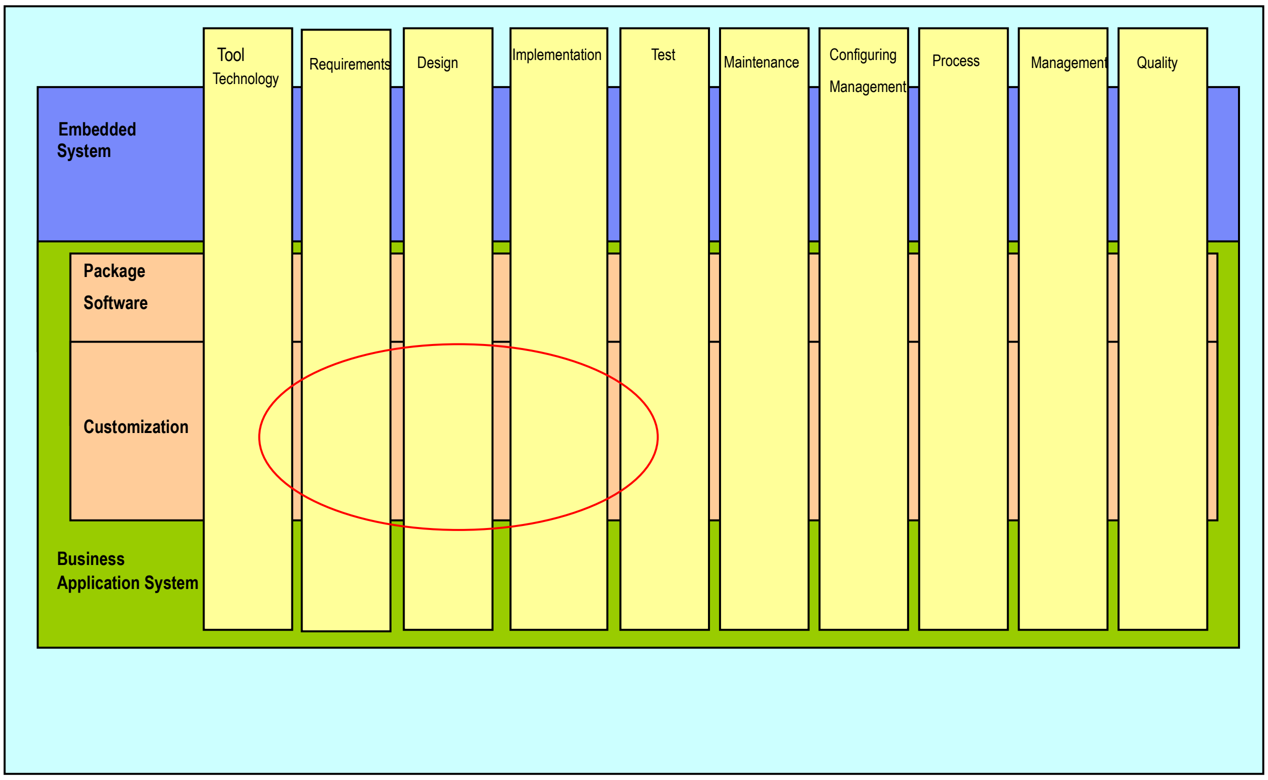
Here are some details of Software Engineering.

Software engineering includes;

- Development technique elements on each process development cycle such as requirement – design – test - maintenance
- Product supporting elements such as tools, development methods, and configuring procedures
- Project maintenance elements such as development processes and budget planning
- Quality maintenance elements such as specification guide and program manuals

Software for engineering includes 1) embedded system software and 2) business application software. The business application has 2a) limited customized design/implementation case within the product specification by using a software package and 2b) large custom development case with new design/implementation although it is difficult to have application package.

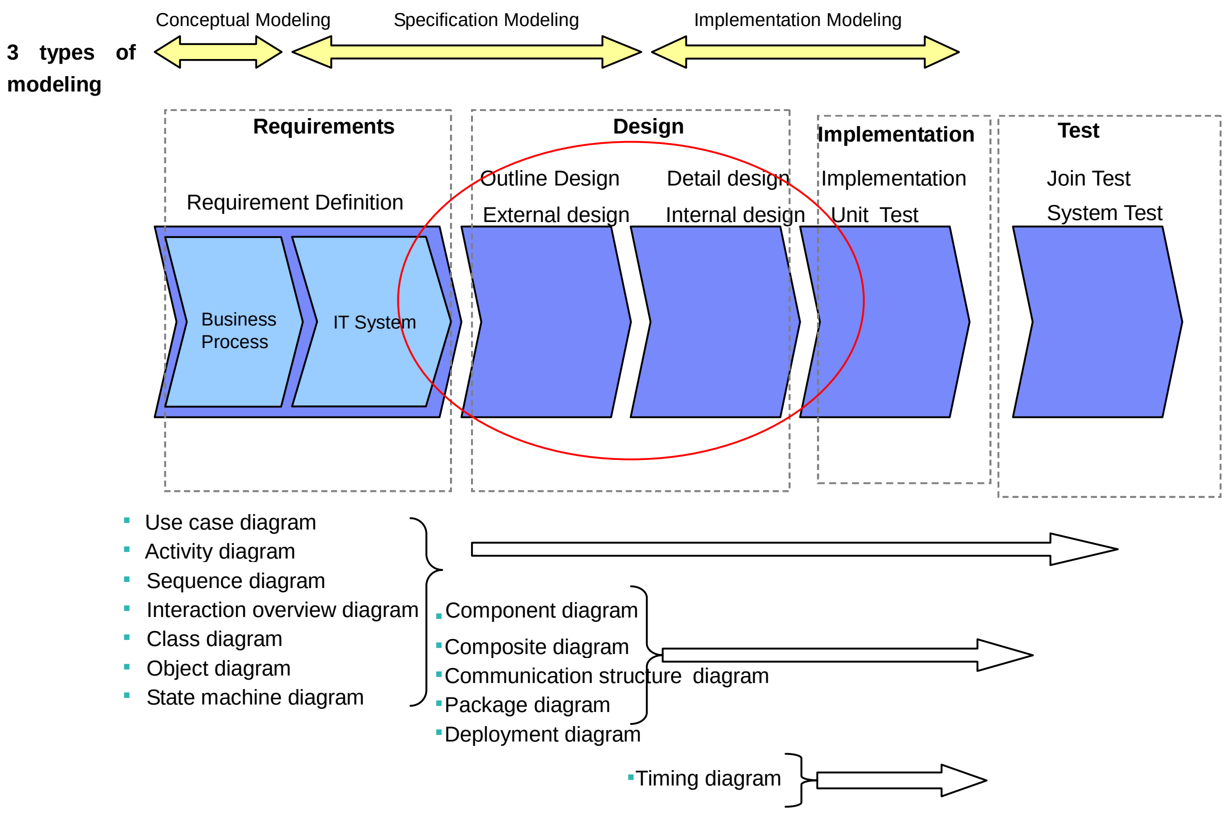
This guideline gives hints and tips about each of the process development cycle elements, especially custom business application software development.



Further, here are some details of each systems development cycle.

Development cycle goes through requirement definition from final requirements, design, implementation, and test. UML is used from IT system requirement definition stage for business process until the implementation stage starting programming. In other words, the requirement definition and design specification are described in UML. The specification in UML is delivered to offshore members who implement the software based on the UML.

This guideline describes the specification in UML for requirement definition and external/internal design. From the modeling point of view, specification modeling and implementation modeling are covered by this guideline also.



This guideline has used requirement definition, external design, internal design, and implementation. However, please refer to the following concepts being described in the next parts.

- Business analysis
- Requirement analysis
- System analysis
- Architecture design
- Detailed design
- Implementation and Unit Test
- Join Test

The offshore development is described as follows in the UML process mentioned above.

In this guideline, offshore means from internal design (detailed design) to implementation after completing external design (architecture design).

In the point of the developers, the issues are;

"What specification, and how, are drawn up in UML for offshore members?"

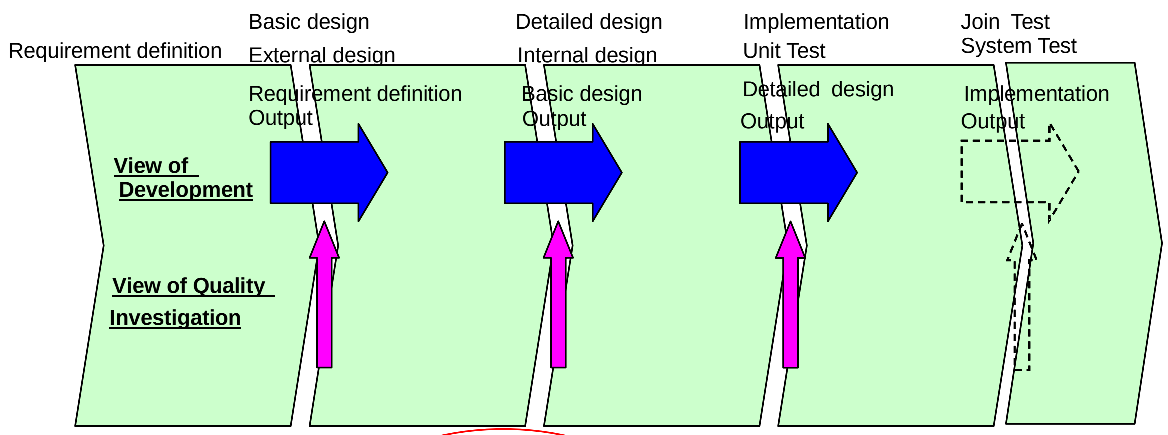
"What are the main features for offshore development?"

Also, the offshore development results are investigated in the following points.

"What factors are necessary to check after offshore development?"

"Which aspects should be satisfied with the offshore development?"

In particular, this guideline gives the answers to the questions above.



		Requirement definition →Basic Design	Basic Design → Detailed Design	Detailed Design →Implementation	Implementation → Join Test/System Test
View of Developers	Japan	How to use/make?	What is to be given?	–	
	Offshore	–	What is to be received?	–	
View of Quality Investigation	Japan	What elements to check?	Level of contents/level to be achieved	What elements to check?	
	Offshore	–	What is o be received	What elements to check?	

The subject of each process has been explained. This guideline will demonstrate mainly two subjects. Firstly ,it will assume the 'Offshore internal design ~ implementation' which is the typical pattern of offshore development and show the requirement definition and the design summary produced by the Japanese project team and how to present them to the offshore team members.

Secondly, it will describe the development output verification of both the Japanese project team and the offshore team.

The subject scope is described in other ways as follows.

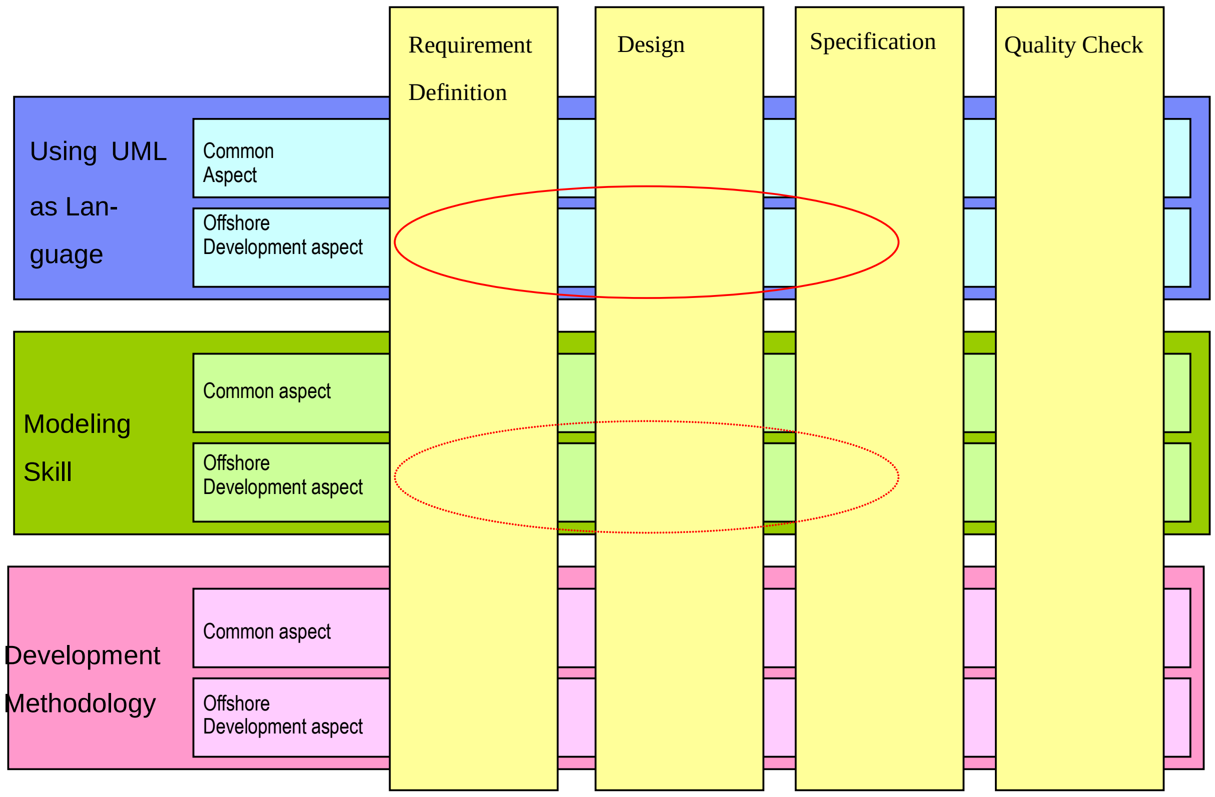
The development process is to analyze requirements, design appropriate systems, send the specification to off-shore, develop program based on the specification, and test the quality.

The relevant points from the above development process are;

- Offshore development has some similarities and some differences with local development. Therefore, the following peculiar cases should be considered.
- Techniques and know-how in case of using UML as a language
- Techniques and know-how in case of modeling
- Techniques and know-how of development methods as an intellectual property based on experience

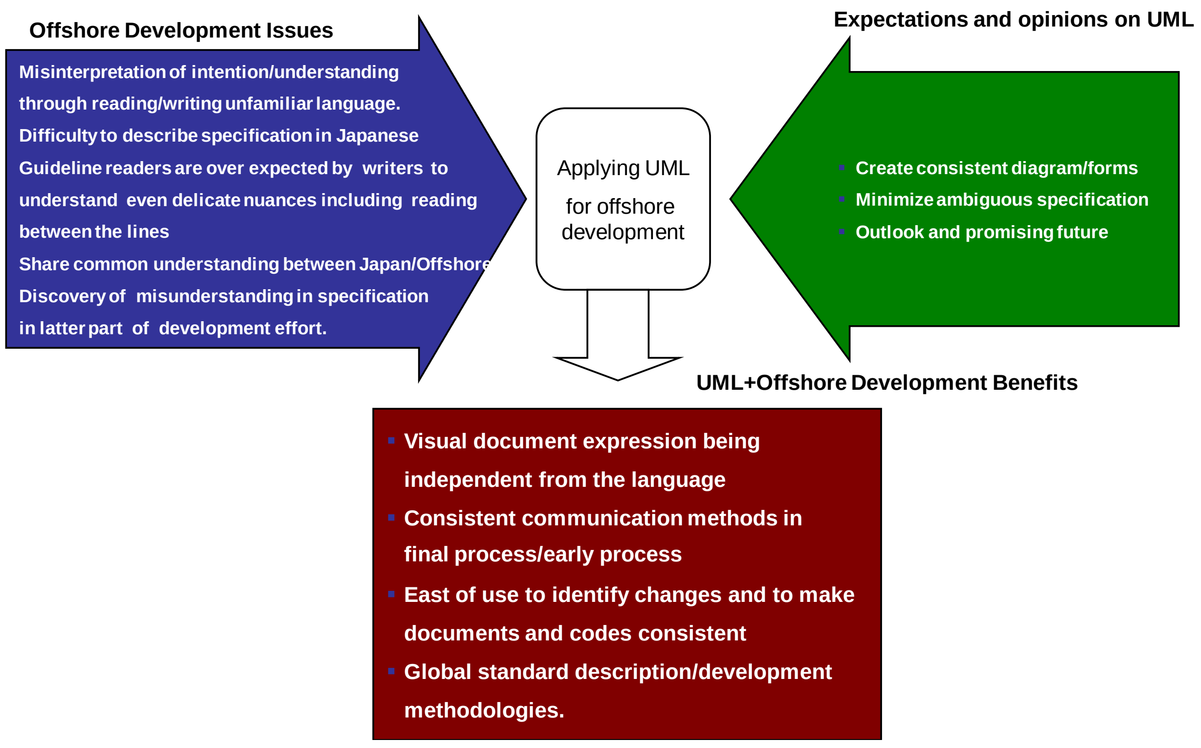
This guideline is limited to describing “Requirements, specification, and test issues in offshore development using UML”. Therefore, it does not explain how to use UML, modeling guide, or detailed development methods.

The readers of this guideline are supposed to already understand UML, modeling, development method, and software development elements. Please apply your knowledge to this guideline.
（This guideline does not aim 「to explain UML, modeling, development method, quality management, or project management or to explain know-how of offshore software development unless they are part of the offshore development scope.）



2. Offshore Development Issues and Solutions

We had a survey targeting the offshore development clients. The survey covered offshore development issues, outlook and opinions about UML, and the benefits from offshore development using UML. The result of the survey is as shown below. Most of the developers have high expectations of UML and believe that it will ameliorate many issues.


Offshore Development Issues
Misinterpretation of intention/understanding through reading/writing unfamiliar language.
Difficulty to describe specification in Japanese
Guideline readers are over expected by writers to understand even delicate nuances including reading between the lines
Share common understanding between Japan/Offshore
Discovery of misunderstanding in specification in latter part of development effort.
Applying UML for offshore development
Expectations and opinions on UML
Create consistent diagram/forms
Minimize ambiguous specification
Outlook and promising future
UML+Offshore Development Benefits
Visual document expression being independent from the language
Consistent communication methods in final process/early process
East of use to identify changes and to make documents and codes consistent
Global standard description/development methodologies.

(1) Overview of Questionnaire

<Questionnaire to Owners (Japan)>
Target: UMTP member companies and their group companies
Date of Survey: November 2006
Qualified Response: 70
(Primarily from PM/PL. Ave. years of experience: 16.2 Offshore years of experience: 2.2)

<Questionnaire to Contractor (offshore party)>
Target: Companies in China of which UMTP member companies and their group companies contract with.
Date of Survey: March 2007
Qualified Response: 91 (Primarily from PM/PL. Ave. years of experience: 4.4)
* China was a chosen targeted country for this particular survey because it is assumed that most of the offshore partnering country is China for Japanese companies.

(2) Extract of the questionnaire response

We introduce the outcome of questionnaire by extracting specific questions regarding issues in offshore development, expectation and comments to UML, and advantages using UML for offshore development.

・Offshore development issues

Table 2.1.1 Offshore development issues and how serious these issues are – Owner (Japan)

		Extremely serious	Very serious	Serious	A little serious	Not serious	More than "serious" (%)
1	Different language/culture causes misunderstanding.	4	19	14	13	10	38
2	It takes a long time to communicate exactly with offshore partners.	1	15	16	18	10	27
3	Ambiguous specification description causes misunderstanding.	1	11	14	22	13	20
4	Discrepancy between the original specification and the source code provided by the offshore could happen	4	17	21	14	4	35
5	Some specification can be missing.	2	13	19	16	10	25
6	Too much time can be spent fielding questions from the offshore company.	6	19	19	13	2	42
7	Serious quality issues can appear during the test phase.	3	14	11	26	6	28
8	It is difficult to describe specification in details.	3	14	22	14	8	28
9	Test from offshore partner can be unreliable.	5	13	14	22	6	30
10	Offshore partner's resignation can cause unreliable project.	12	22	13	9	3	58
11	Debugging lead time can be long if offshore partners do maintenance.	11	23	16	7	2	58
12	Efficient information sharing is difficult with offshore partners.	7	27	17	6	2	58

Table 2.1.2 Offshore development issues and how serious these issues are – Offshore partner

(Remarks: Numbers of questionnaires are not sequential because some questions are not identical to the questionnaires to Owner.)

		Extremely Serious	Very Srious	Serious	A little serious	Not serious	More than "serious"(%)
1	Different language/culture causes misunderstanding	6	7	11	31	38	14
2	It takes too much time to communicate with the Owner (Japan)	2	6	14	32	37	9
3	Ambiguous specification causes misunderstanding	9	28	24	28	2	41
4	Discrepancy between the original specification and the source code provided by the offshore could	13	12	10	27	28	28
5	There are missed description in specifications	12	23	33	19	4	38
12	Can not share information effectively with Owner based on the same understanding and interpretation	7	27	17	6	2	58
13	Communication between project members are not satifactory	4	15	16	23	32	21

It is now understood that both Owners and Offshore parties recognize #12, which is “cannot share information effectively based on the same understanding”, a serious problem. Offshore parties believe it is because specifications provided by Owners are ambiguous, which is shown in #3. Owners, however, do not recognize it a problem. It is assumed that Owners have difficulties to communicate with offshore parties fully by their system specifications and Owners end up spending too much time responding so many questions from offshore parties, which is shown in #6.

Solution to the offshore development issues

It is recognized that 74% of people, shown in table 2.2, are using not just UML but any diagrams and charts to review status or write specifications to avoid such issues as that Owners and Offshore parties could not communicate effectively based on the same understanding indicated in #12 in the table on the previous page. It is interpreted that most of the developers using diagrams and charts actively.

Table 2.2 Percentage of using UML or other diagrams in review process, specifications – Owner (Japan)

	Case	Average (%)	
(a)	UML (at least a diagram) is used.	19%	74%
(b)	Same diagram format excluding UML is used in organization.	38%	
(c)	Identified diagram format is used personally - no (a) and (b) cases.	17%	
(d)	Random various diagram formats are used.	14%	
(e)	Only characters but not diagrams are used.	5%	
(f)	No written format is used.	3%	
(g)	Others.	4%	

Further, another survey shown below explains how UML is recognized. It seems that UML is recognized that it is more specific and clear than other diagrams and charts.

Table 2.3.1 Expectations and Comments to UML – Owner (Japan)

	Comments	Totally Disagree	Rather Diagree	Neutral	Rather Agree	Totally Agree	% of "Agree"
1	UML should only be known by those who get involed in high level system design	29	28	8	2	0	3
2	UML is not much help to reduce cost	3	9	28	21	6	40
3	UML expertise are not there around us	10	17	9	22	9	46
4	UML training is a tough part. It costs us a lot, as well	4	10	17	31	5	54
5	UML increases productivity	1	4	38	19	5	36
6	UML reduces reversal processes in programinning phase	2	5	30	24	6	45
7	UML brings more quality	2	4	27	29	5	51
8	UML is too professional and is difficult to use	9	22	24	10	2	18
9	UML makes difficult to quote volume of work and cost	5	20	32	8	2	15
10	UML makes project management difficult	6	20	37	4	0	6
11	UML makes specifications less ambiguous	1	2	18	37	9	69
12	UML makes easier to understand how changes impacts the other areas	2	6	17	32	10	63
13	UML helps better and more effective communication in review process	2	4	15	35	11	69
14	UML creats mess in development processes	12	29	20	6	0	9
15	There are more suitable tools in high-level design phase rather than UML	5	15	36	9	2	16
16	UML accelerate communications between Owners and Offshore partners	2	3	30	23	9	48
17	UML reduces cost in maintenance	1	13	36	12	5	25
18	UML is effective to automate development process	3	15	26	18	5	34
19	UML is more used in offshore development rather than in Japan	3	6	42	12	3	23
20	UML better matches with offshore development rather than development in Japan	4	6	29	24	3	41

Table 2.3.2 Expectations and Comments to UML – Offshore Partner

(Remarks: Numbers of questionnaires are not sequential because some questions are not identical to the questionnaires to Owner.)

	Comments	Totally Disagree	Rather Diagree	Neutral	Rather Agree	Totally Agree	% of "Agree"
1	UML should only be known by Owners (Japan)	51	22	4	4	1	6
2	UML is not much help to reduce cost	7	23	35	15	2	21
3	UML expertise are not there around us	27	19	23	9	4	16
4	It costs a lot to learn UML	4	18	26	27	6	41
5	UML increases productivity	0	0	14	47	21	83
6	UML reduces return processes from Owners	1	6	21	33	18	65
7	UML brings more quality	0	1	26	36	19	67
8	UML is too professional and is difficult to use	10	21	20	23	8	38
9	UML makes difficult to quote volume of work and cost	4	32	39	7	0	9
10	UML makes project management difficult	6	45	27	3	1	5
11	UML makes specifications less ambiguous	1	3	12	45	21	80
12	UML makes easier to understand how changes impacts the other areas	0	3	18	37	24	74
13	UML helps better and more effective communication in review process	1	4	19	35	23	71
14	UML creats mess in development processes	24	37	15	5	1	7
16	UML accelerate communications between Owners and Offshore partners	0	3	21	40	17	70
18	UML is effective to automate development process	1	3	23	43	12	67
19	UML is more used in offshore development rather than in	2	12	44	20	4	29
20	UML better matches with offshore development rather than development in Japan	4	4	43	27	6	39
21	There are many UML textbooks available	0	13	16	30	23	65
22	UML helps to develop career and helps promotion	1	4	25	28	24	63

Both Owners and Offshore Partners recognize such advantages as "#11, makes less ambiguous", "#12, easier to understand how changes impact", or "#13, helps effective communications in review process". They both also believe that "#16 UML accelerate communications", "#7 UML bring more quality". It is also recognized that level of expectations of Offshore Partners to UML is very high. On the other hand, Owners recognize high training cost and UML is difficult to educate (#4) .

Benefits of UML being used for Offshore Development

Table 2.4.1 Benefits of UML being used for Offshore Development – Owner (Japan)

	Benefits	Not important	Important	Very important	Weighted Total
①	Global standard documentation	13	26	22	**131**
②	Visual documentation	10	26	24	**134**
③	Client(User) friendly documentation	13	27	20	**127**
④	Easy maintenance by using specification without program source code	14	31	15	**121**
⑤	Seamless development through the entire process	16	27	18	**124**
⑥	Same tool can be used by offshore and local developments.	9	21	30	**141**
⑦	Identified communication tool is used from the beginning and ending processes.	8	26	26	**138**
⑧	Global standard development methods can be used.	10	27	24	**136**
⑨	Developed process can be reused.	15	28	17	**122**
⑩	It does not have to rely on natural language.	13	30	17	**124**
⑪	Specification and source code should be matched.	8	31	21	**133**
⑫	Offshore developer's language(ex. Chinese) can be used.	24	22	14	**110**
⑬	Quality can be assured.	13	21	26	**133**

Table 2.4.2 Benefits of UML being used for Offshore Development – Owner (Japan)

	Benefits	Not important	Important	Very important	Weighted Total
①	Global standard documentation	7	41	43	**218**
②	Visual documentation	5	44	42	**219**
③	Client(User) friendly documentation	3	43	45	**224**
④	Easy maintenance by using specification without program source code	5	43	43	**220**
⑤	Seamless development through the entire process	2	48	41	**221**
⑥	Same tool can be used by offshore and local developments.	17	42	32	**197**
⑦	Identified communication tool is used from the beginning and ending processes.	7	39	58	**259**
⑧	Global standard development methods can be used.	5	46	40	**217**
⑨	Developed process can be reused.	5	36	50	**227**
⑩	It does not have to rely on natural language.	10	53	28	**200**
⑪	Specification and source code should be matched.	5	26	60	**237**
⑫	Owner's language (Japanese) can be used	5	35	51	**228**
⑬	Quality can be assured.	5	20	66	**243**

Communication with Offshore Partners is very important in Offshore Development. Having said that, "#12, Other party's language can be used" would be extremely important. Between China and Japan offshore relationship, Chinese offshore partners typically understand Japanese so that Owners do not recognize significant advantages using UML. However, Offshore Partner does recognize significant advantage using UML. It is assumed that most of high-level system design is done by Owners (Japan) and development effort is done by the contractors (Offshore Partners). In that sense, both parties recognize benefits of having common communication tool (#7).

Also, such unique benefits as "#2, Visual Documentation", "#1, Global Standard Documentation", "#8, Global Standard Development Methods" are all important in the offshore development.

Further, such recognition as "#11, consistency between source code and specification is successfully maintained" is supported by many people involved in the offshore development.

(3) Voices and Comments

The following hearing process was implemented to support outcome of the questionnaire above. It is then confirmed that this particular guideline is convincing enough from Offshore Partners point of view, and that it could possibly be used for other offshore development partners' country such as India.

<Hearing from PM/PL in China (partially done by written questionnaire format)>
Hearing Objective: To better understand the reality which many not be discovered by quantitative questionnaire survey.
Date: November 2007

<.Hearing from Architects in India>
Hearing Objective: To understand how UML is being used in US/India offshore relationships which is ahead of Japan/China offshore relationships
Date: March 2008

3. UML Modeling

3.1 UML Features

(1) What is UML?

UML (Unified Modeling Language) is a standard by the OMG (Object Management Group) to specify and make diagrams for software development.

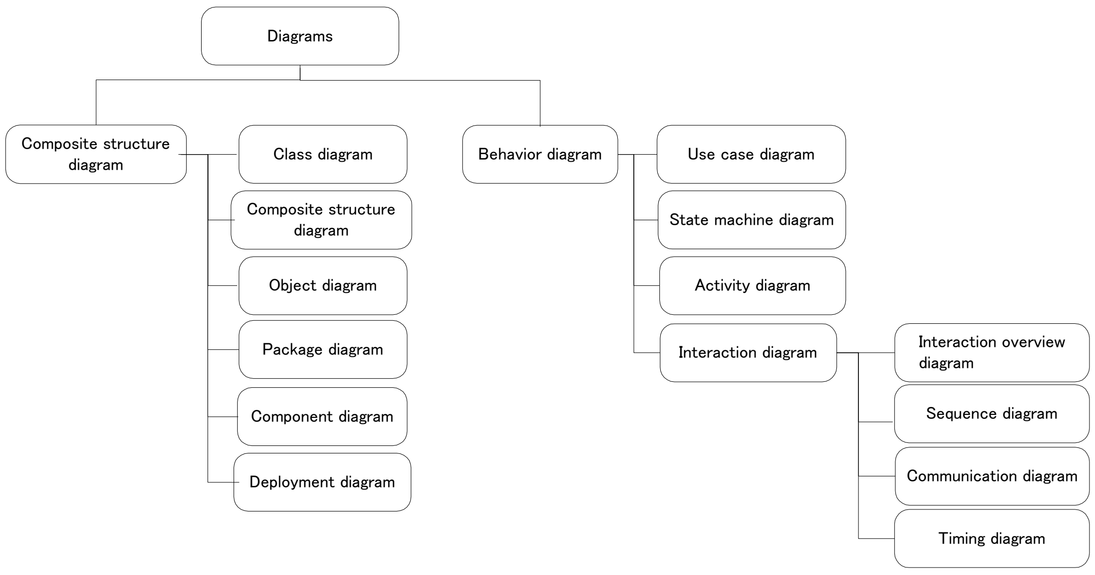
(2) UML Features

UML has the following features.

- Global standard.
  - Easy communication between countries and different projects.
- Easy to display and understand.
  - UML provides various diagrams to analyze and display designs in intuitive and easy to understand ways.
- Identified method through the entire development process.
  - Each development is traceable and communication between developers in each process is easy.

(3) UML Diagrams

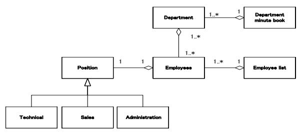
UML has various diagram types as shown below.



▪Component structure diagrams

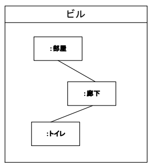
● Class diagram

Objects in analysis and design area are analyzed conceptually and displayed by class and the relationships.



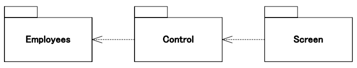
● Composite structure diagram

Internal structure of class or component is displayed by class.



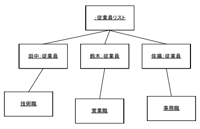
● Package diagram

Package relationships with UML elements are displayed.



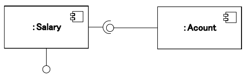
● Object diagram

A system state at a moment is displayed in object or the relationships.



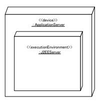
● Component diagram

Software component (reusable) configuration is displayed.



● Deployment diagram

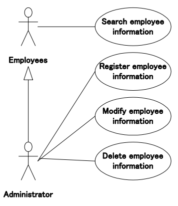
System running environment is displayed.



▪Behavior diagrams

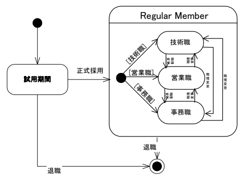
● Use case diagram

Functions and the external relationships provided by the system are displayed.


Employees
Administrator
Search employee information
Register employee information
Modify employee information
Delete employee information

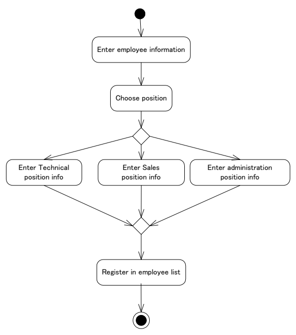
● State machine diagram

An object state change by time is displayed.



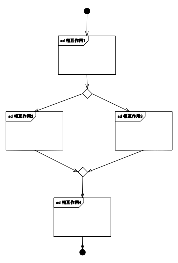
● Activity diagram

Business /event/algorithm flows are displayed.


Enter employee information
Choose position
Enter Technical position info
Enter Sales position info
Enter administration position info
Register in employee list

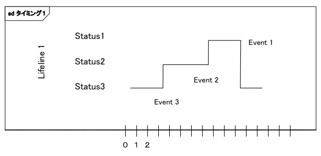
● Interaction overview diagram

Various interactivity relationships are displayed.



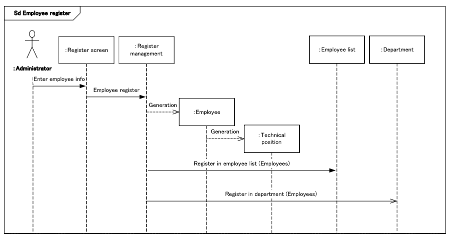
● Timing diagram

Lifeline changed by time and message exchange are displayed.



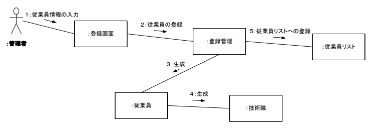
● Sequence diagram

Interaction (object or person exchange in analysis and design) is displayed by time.



● Communication diagram

Interaction (object or person exchange in analysis and design) is displayed by relationships with object or person.



3.2 Required UML Modeling Skill Level

Both Offshore Owners and Offshore Partners need to maintain qualified level of UML modeling skill for them to maintain successful and effective offshore development relationship using UML. If this particular skill set is not available yet, required training and education are highly recommended prior to the offshore development project starts. It should also to be understood that required UML modeling skill level would be varied from such positions as Analyst, Designer, Production Engineer, Architect, Project Manager and Project Leader.

UML Modeling skill level defined here in this section is only for UML Modeling, and it presumes that all such engineers as Analyst, Designer, Production Engineer, Architect, Project Manager and Project Leader have had respective technical skill required for such respective positions. Therefore, this particular section does not define any of these generic skill sets.

<Objective>

It is expected that both Owners and Offshore Partners exchange information of artifact created by UML. If both parties have qualified UML modeling skill, communications between them will be successful by minimum effort of exchanging artifact. If, however, UML modeling skill developers are not enough, communication between Owners and Offshore Partners using artifact of specifications will not be accurate. It will further cause problems such as that Owners development engineers will need to explain what UML modeling is to Offshore Partners' engineers. This will be a waste of time and effort.

<Details/Supplements>

➢ How to judge the skill level

UMTP defines modeling skill level from L1 through L4 as shown below.

	Modeling Skill	Description
L4	Can train other engineers and help them to improve their modeling skill based on his/her own modeling expertise	Require L3 skill level. Also required qualified number of years modeling experience or number of projects experience in software development projects.
L3	Can implement UML models in the production environment	Can provide high-quality models with scalability and ease of changes capabilities.
		Has professional knowledge (professional fileds can be chosen) for business modedling, analysis, architectual design and embedded development.
L2	Can write/read UML models without difficulties (qualified level of modeling literacy)	Can be involved in the modeling project being responsible for a part of the targeted development area.
		Can interpret the models created by other engineers.
L1	Can interpret simple UML models	Has minimum level of knowledge of modeling to creat models by a technique such as UML modeling.

UMTP offers qualified tests for L1 and L2 levels (L3 test is expected to be available from April 2008) This test qualifies your skill level appropriately.

There are two levels of test for L1, L1-T1 and L1-T2.

Qualified levels are:

L1-T1: Understands how to write UML models.

L1-T2: Able to read UML models and successfully judge the quality of the model.

* L1-T2 and L2 tests are available in China.

A real development project requires the following skill level depending on the roles.

Production Engineers:	Create source code from UML models (level L1-T2).
Analysts:	Create UML analytic models by himself/herself directed by senior analysts (Level L2)
Designer:	Create UML design models by himself/herself by the architecture provided. (Level L2)

Senior Analyst:	Create initial analytic model (Level L3)
Architect:	Create architectural model (Level L3)
Project Leader:	Review model analysis, design rules, and determine what to do (Level L3)
Project Manager:	Highest level of project management (Level L1, L2)

	L1-T1	L1-T2	L2	L3
Production Engineer	Yes	Yes	No	No
Analyst	Yes	Yes	Yes	No
Senior Analyst	Yes	Yes	Yes	Yes
Designer	Yes	Yes	Nice to have	No
Architect	Yes	Yes	Yes	Yes
Project Leader	Yes	Yes	Yes	Yes
Project Manager	Yes	Yes	Nice to have	No

➢ Skills Development

Getting familiar with UML (L1-T1) and learning entry level of modeling (L1-T2) can be done by a text book available in book stores. However, extra effort might be required for L2, which is to develop UML model without help. It requires specific training or mentoring to get to the point for better model development, and further requires practical modeling development experience for more comprehensive understanding.

➢ Reference Information

UMTP Modeling Standard Based

Some text books available at book stores have levels as UMTP Modeling Standard Based. This indicates that UMTP verifies that these books follow UML Modeling terminologies (Ver.2). If successfully followed, UMTP authorizes them.

All UMTP Modeling Standard Based books can be found at the following URL.

http://www.umtp-japan.org/modules/data4/index.php?id=7

- Training Text Authorization

It is reviewed in detail during UMTP authorization process whether or not the training text follows UMTP standard terminologies (Ver.2) and whether or not the training text addresses knowledge required for UML modeling and is explained in detail. If successfully followed, UMTP authorizes it.

UMTP authorized training courses can be found at the following URL.

http://www.umtp-japan.org/modules/data4/index.php?id=8

< Relevant Information >

Not available

4. UML Application Scope and Development Know-how (Hints & Tips)

According to the survey in the chapter 2, it is understood that most people recognize the advantage of using UML for offshore development. It is,however,very important in which status UML is applied because the effect is not immediate. This particular chapter explains know-how to succeed in offshore development for each process. It will be focused on UML but not limited to it.

Development flow and each process purpose, task, staff and output are described in figures 4-1 and Table 4-1 respectively. Figure 4.1 only shows the important artifacts. Detail of that should be referred to Table 4.1. Figure 4.1 also shows which tasks may require special attention in each step of development process (know-how) by point number.

The know-how will be described in the following formats.

- ･Point numbers and titles
- ･Contents
- ･Purpose
- ･Detailed description
- ･Relevant information

Besides, there is another category, which is “optional”(can be chosen depending on the situation), for know-hows that is not necessarily executed in the initial stage of offshore development although it is typically effective if all other conditions are satisfied.

Figure 4.1 Development workflow

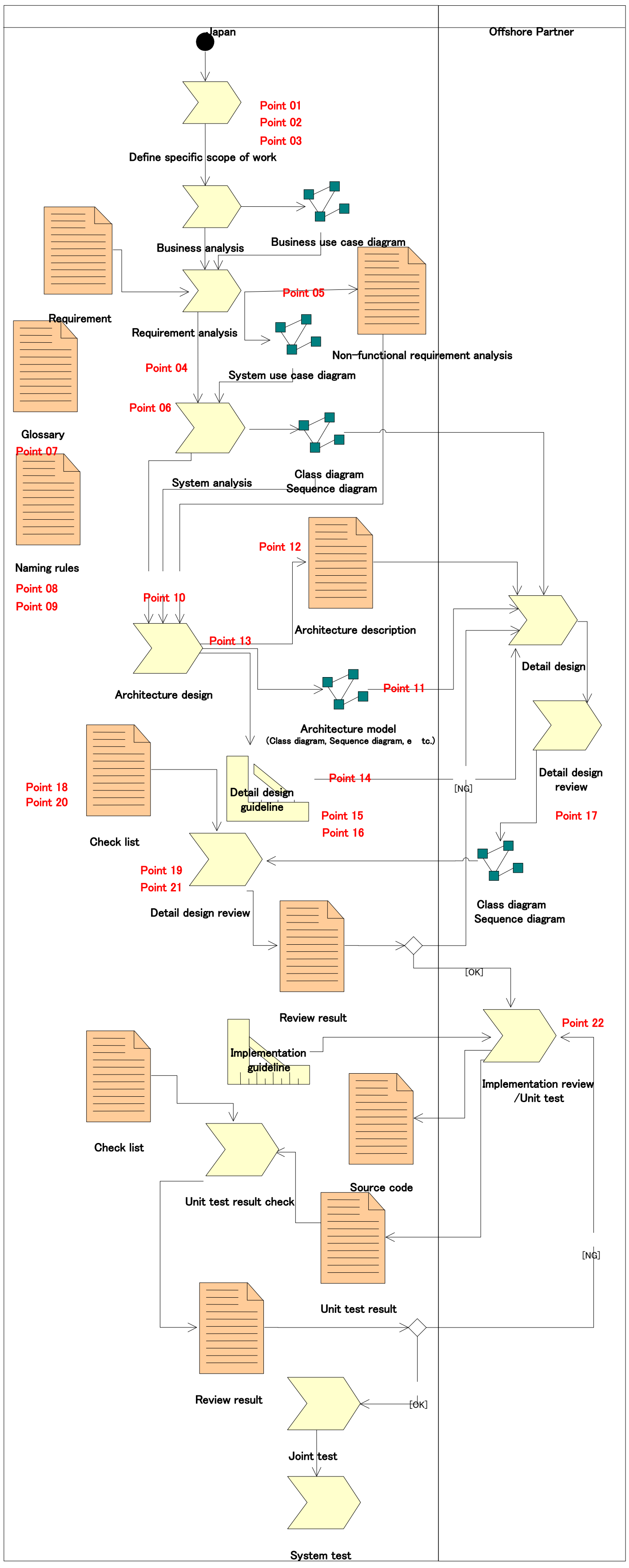


Table 4.1 Process description

Process name Business Analysis	
Purpose(Japan)： Understand business flow of the system to be developed in detail and clarify it. This helps further clarify to define which particular part of the business process to be targeted for system development.	Purpose(Offshore Partner)： Understand targeted business process to be developed. This should support any successive processes to be moved on smoothly.
Task(Japan) ： Understand business process in detail and clarify it. Organize business flow once again to make sure if necessary.	Task(Offshore Partner) ： Understand targeted systems development scope and associated business processes. (in case participated in high-level system design).
Developer(Japan) ： PM Analyst	Developer(Offshore Partner) ： PM System analyst (if s/he participates in high-level system design phase)
Input required： ･Business flow	Input required： ･Business Use Case Diagram ･Business Flow ･Business Use Case Description ･Conceptual Class Diagram
Output： ･Business Use Case Diagram ･Business Flow ･Business Use Case Description ･Conceptual Class Diagram	Output：
Reference Information ･Glossary ･Naming Rules	Reference Information ･Glossary ･Naming Rules

Process name Requirement analysis	
Purpose(Japan): Verify the functional requirements and the non-functional requirements of the system.	Purpose(Offshore Partner): The process assigned after detailed design is not using this process output directly. Understand system function as this system development object in this process output. Recognize the design purpose for the better communication after detailed design.
Task(Japan): Verify the requirement of clients.	Task (Offshore Partner): Refer to the output and understand the system function. (If partiticpated in high-level design process.)
Developer(Japan): Requirement analyst	Developer(Offshore Partner): System analyst Detail system analyst (if participated in high-level design process.)
Input required： ・Business use case diagram ・Business flow ・Business use case description ・Requirement Detail	Input required： ・System use case diagram ・System use case list ・Busines rule definition ・System sequence diagram ・System use case description ・Screen transition diagram ・Screen and design documentation ・Non-functional requirement definition
Output : ・System use case diagram ・System use case list ・Business rule definition ・System sequence diagram ・System use case description ・Screen transition diagram ・Screen and design documentation ・Non-functional requirement definition	Output :
Reference Information ・Glossary ・Naming Rules	Reference Information ・Glossary ・Naming Rules

Process nam System analysis	
Purpose(Japan): Analyze system development object domain in natural form and make model not relying on development and operation system to reuse and maintain.	Purpose(Offshore Partners): The process assigned after detailed design is not using this process output directly. Aim to acquire system development object knowledge from this process output and understand the design purpose for better communication after detailed design.
Task(Japan): Define system context displayed brief system structure and relevance. Then, analyze system behavior with summary to understand system geographic dispersion or complex operation.	Task (Offshore Partners): Refer to the output and understand the system development area. (If participated in high-level system design process)
Developer(Japan): Requirement analyst	Developer (Offshore Partner): System analyst Detail system designer (if participated in high-level system design process).
Input required： ･System use case diagram ･System use case list ･Business rule definition ･System sequence diagram ･System use case description ･Screen transition diagram ･Screen and design documentation	Input required： ･Analysis class diagram ･Analysis sequence diagram ･Object diagram (･Communication diagram) (･State machine diagram)
Output : ･Analysis class diagram ･Analysis sequence diagram ･Object diagram (･Communication diagram) (･State machine diagram)	Output :
Reference Information ･Glossary ･Naming Rules	Reference Information ･Glossary ･Naming Rules

Process name Architecture design	
Purpose(Japan): Verify the fundamental system structure.	Purpose (Offshore Partner): Use as input to the detail design.
Task(Japan): Define a candidate architecture including reusable modeling elements based on analyzed result. Achieve architecture analysis in detail and distinguish class, subsystem and subsystem interfaces in component design and	Task (Offshore Partner): Refer to the output to detail design (If participated in high-level system design process).
Developer(Japan): Architect	Developer (Offshore Partner): Architect Detail system designer (If participated in high-level system design process.)
Input required ： ･Analysis class diagram ･Analysis sequence diagram ･Object diagram (･Communication diagram) (･State machine diagram) ･Non-functional requirement definition	Input required ： ･Architecture class diagram ･Architecture sequence diagram ･Object diagram ･Package diagram ･Component diagram ･Deployment diagram ･Architecture description ･Detailed architecture guideline ･Detailed architecture checklist ･Data model diagram (E-R diagram) ･Analysys class diagram ･Analysys sequence diagram
Output : ･Architecture class diagram ･Architecture sequence diagram ･Object diagram ･Package diagram ･Component diagram ･Deployment diagram ･Architecture description ･Detailed architecture guideline ･Detailed architecture checklist ･Data model diagram (E-R diagram)	Output :
Reference Information ･Glossary ･Naming Rules	Reference Information ･Glossary ･Naming Rules

Process name：Detail Design	
Purpose(Japan): Assign the large tasks to China.	Purpose (Offshore Partner): Detail the model to implementation
Task(Japan): Verify the task from China is within the architecture scope.	Purpose (Offshore Partner): Input the architecture model and description to control subsystems and detail the implementation model.
Developer(Japan): Reviewer	Developer (Offshore Partner): Architect (if participated in high-level system design process).
Input required： ・Detailed architecture class diagram ・Detailed architecture sequence diagram (・Object diagram) ・Detailed architecture review result (Offshore Partner)	Input required： ・Architecture class diagram ・Architecture sequence diagram ・Object diagram ・Package diagram ・Component diagram ・Deployment diagram ・Architecture description ・Detailed architecture guideline ・Detailed architecture checklist ・Analysys case diagram ・Analysys sequence diagram
Output : ・Detailed architecture review result (Japan)	Output : ・Architecture class diagram ・Architecture sequence diagram (・Object diagram) ・Detailed architecture review result (Offshore Partner)
Reference Information ・Glossary ・Naming Rules	Reference Information ・Glossary ・Naming Rules

Process name	**Implementation/Review/Unit Test**
Purpose(Japan): Assign the large tasks to Offshore Partner.	Purpose(Offshore Partner): Build the source program which runs the system.
Task(Japan): Review test status as required.	Task (Offshore Partner): Make implementation and individual test based on the detail design output.
Developer(Japan): Reviewer	Developer (Offshore Partner): Person in charge of implementation
Input required： ・Unit test result	Input required： ・Detail design class diagram ・Detail design sequence diagram (・Object diagram)
Output : ・Unit test review result	Output : ・Source program ・Each test result
Reference Information ・Glossary ・Naming Rules ・Implementation Guideline ・Check List	Reference Information ・Glossary ・Naming Rules ・Implementation Guideline ・Check List

Process name：　Unit Test Result Review	
Purpose(Japan):	Purpose (Offshore Partner):
Task(Japan): Review unit test result.	Task (Offshore Partner): Revisit unit test result as required.
Developer(Japan): Reviewer	Developer (Offshore Partner): Person in charge of implementation
Input required： ･Unit test result	Input required： ・Review result
Output : ・Review result	Output : ・Unit test result
Reference Information ・Glossary ・Naming Rules ・Implementation Guideline ・Check List	Reference Information ・Glossary ・Naming Rules ・Implementation Guideline ・Check List

Process name：Join Test	
Purpose(Japan): Perform join test on the source program through the Unit Tests.	Purpose (Offshore Partner): Improve source program quality.
Task(Japan): Perform Join Test.	Task (Offshore Partner): Fix bugs in the detail design and implementation.
Developer(Japan): Tester	Developer (Offshore Partner): Person in charge of detail design Person in charge of implementation
Input required： ･Source program ･Each test result	Input required： ･Detail Design class diagram ･Detail Design sequence diagram ･Source program ･Bug list
Output : ･Join test result	Output : ･Detail Design class diagram ･Detail Design sequence diagram ･Source program ･Bug list ･Checklist
Reference Information ・Glossary ・Naming Rules	Reference Information ・Glossary ・Naming Rules

Table 4.2 Example for each process output

Process	Output set	Output	UML diagram
Business Analysis	Business process model	Business use case diagram	Use case diagram
		Business flow	Activity diagram
		Business use case description	
	Domain model	Concept class diagram	Class diagram
			Object diagram
Requirements Analysis	Use case model	System use case diagram	Use case diagram
		System use case list	
		Business rule definition	
		System sequence diagram	Sequence diagram
			Interaction overview diagram
		System use case description	Activity diagram available
	User interface model	Screen transition diagram	State machine diagram
		Screen, Design book	
	Non-functional requirements	Non-functional requirements definition	
	Glossary	Glossary	
System Analysis	Analysis model	Robustness diagram	
		Analysis class diagram	Class diagram
			Object diagram
		Data model diagram（E-R diagram）	
		Interaction diagram	Sequence diagram
			Communication diagram
		State diagram	State machine diagram
Architecture Design/ Detail Design	Architecture model	Architecture description	
		Software configuration	Package diagram
			Component diagram
			Composite structure diagram
		Hardware configuration	Deployment diagram
		Fundamental process method	Sequence diagram
	Design model	Architecture class diagram/Detail design class diagram	Class diagram
			Object diagram
		Data model diagram（E-R diagram）	
		Interaction diagram	Sequence diagram
			Communication diagram
		State diagram	State machine diagram
			Timing diagram
		Class specification	
Embedded	Embedded set	Implementation code	
Unit test/ Join test Process	Test set Output set	Test plan	
		Test specification	
		Output	UML diagram

Table 4.3 Development Know-How (Hints and Tips)

Process	Point No.	Know-How	
Define specific scope of work	01	Define specific scope of work, roles and responsibilities	(∗)
	02	Determining UML diagram to be used	
	03	It does not have to be UML at all times	(∗)
Business Analysis / Requirement Analysis	04	Participate early development process *(optional)*	
	05	Non-functional requirement definition	
	06	Understanding business process by analysis models	(∗)
	07	Creating glossaries	
	08	Create naming rules	
	09	Creating modeling rules	
System Analysis/ Architecture Design	10	Explicit common function	
	11	Architecture Model	
	12	Create output of architecture description	
	13	Using patterns	
	14	Level of description of specification, format specification	(∗)
	15	Creating detail design guideline	
	16	Clarifying areas of which specifications are not determined	
Detail Design / Implementation	17	Reviewing output by the offshore partner	
	18	Repeating check by the owner (Japan)	
	19	Validation and Verification	
	20	Reviewing detail design by the models	
	21	Consistency between UML diagrams	(∗)
	22	Using code generation feature in the tool for production (Optional)	
Remarks		Unique nature of systems development in Japan	

(Optional): Expected to be very effective if it is executed under well-prepared appropriate situation. However, it is not necessary required in the early stages of offshore development.

(∗): Samples are available. See Appendix A for explanation of the samples.

【Point 01 Define specific scope of work, roles and responsibilities.】

Each process should be assigned clearly. Output (with UML or non-UML) for each process should be decided in advance. Modeling and test scopes need to be distinguished precisely in offshore development.

【Purpose】

A system development project can be processing in several offices/branches (local or worldwide) together. Every project has different task and output in each process. Task for each process should be correctly recognized to avoid any confusion when project starts. Client and offshore members verify their outputs in each process to avoid issues after completion.

【Details ・ Supplementation】

Offshore task for each process should be correctly assigned for efficient processing. In many cases offshore members are not available to deal with environmental aspect, operational requirements, troubleshooting requirements, and security issues. Therefore, it is helpful to divide the tasks for offshore team and client (Japanese project company) in advance.

<General process>

- ・ Describe outcome for each process.
- ・ Verify input/output for each process.
- ・ Produce the guideline for output forms and contents granularity.
- ・ Decide how to verify output to avoid inconsistency between related outputs.

<Architecture design ・ Detail design（modeling）>

- ・ Decide the fundamental development and other development scopes to be provided.

<Implementation ・ Unit test （preparation and production）>

- ・ Development environment requirements（including server, software, and version）
- ・ Scheduling to provide components （available date）
- ・ Components provided（summary, external requirements）
- ・ Development tools

<Test>

- ・ Consider development environmental condition
- ・ Choose if test and sample data will be provided （Sample data will be helpful for better understanding）
- ・ It is expected to provide test data as early as possible if test data is provided

Offshore development party should particularly consider the followings in the early stage of development.

- ・ Any constraints in the offshore development environment and test environment

For example, it is required to check in an early stage if documents in Japanese/Chinese will be used, large server/computer can be prepared, or files can be shared.

・ Any constraints on non-functional requirements implemented by offshore architecture

For example, some tests need to have an embedded verification function and the other tests rely on the performance during the test.

・ Limited test data in offshore environment

For example, secured client data or some test data in Japanese or Chinese is difficult to share because of the language barrier.

It should be recognized that communication is extremely important in the offshore development project. Detail and specific communication planning will be required. Specifically, who should communicate with whom, either by weekly or any other cycle.

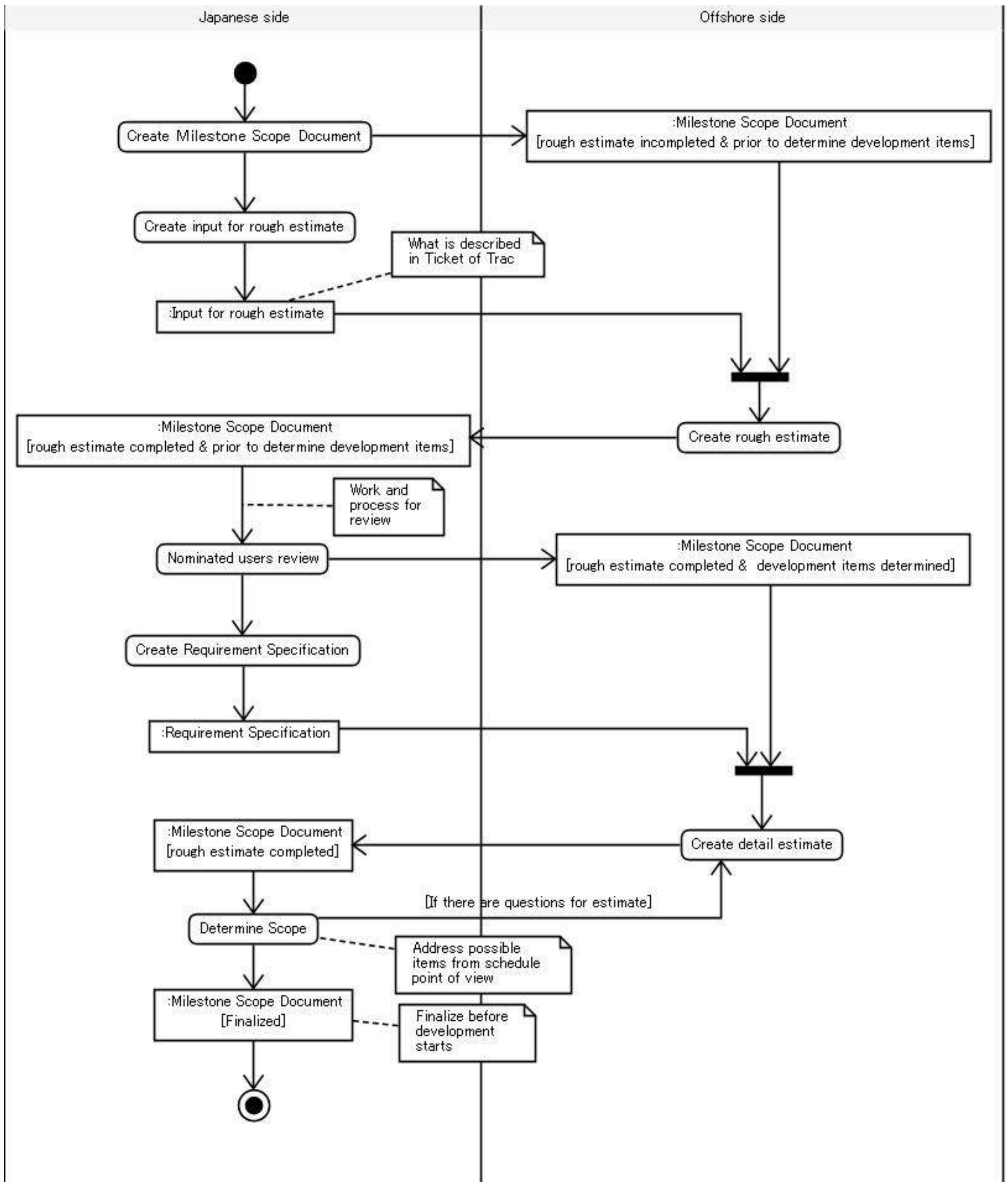
Not just scope of work and associate responsibility, It is necessary to clearly define the development organizations in the offshore team.

【Relevant information】

Figure 4.1 Development Process

Table 4.1 Description of Each Process

<Example of using activity diagram to clearly define works and responsibilities>
Swimlane in the activity diagram is used to identify the works of Japan and offshore partner.


Japanese side
Offshore side
Create Milestone Scope Document
:Milestone Scope Document [rough estimate incompleted & prior to determine development items]
Create input for rough estimate
What is described in Ticket of Trac
:Input for rough estimate
Create rough estimate
:Milestone Scope Document [rough estimate completed & prior to determine development items]
Work and process for review
Nominated users review
:Milestone Scope Document [rough estimate completed & development items determined]
Create Requirement Specification
:Requirement Specification
Create detail estimate
:Milestone Scope Document [rough estimate completed]
[If there are questions for estimate]
Determine Scope
Address possible items from schedule point of view
:Milestone Scope Document [Finalized]
Finalize before development starts

【Point 02 Determining UML diagram to be used.】

Clarify which UML diagrams and supplementation will be used for each process.

【Purpose】

Use appropriate UML diagram in each process development. Decide which UML will be used by process when project starts.

【Details ・ Supplementation】

Decide which UML diagrams will be used in each process to avoid missing any steps. Analyze UML diagram link such as diagrams by random/compulsory classification to explain the basis of the model. Consider using UML diagram as shown below for business analysis ~ detail design in each process.

<Task analysis>	
・Use case diagram	△（depending on status）
・Activity diagram	○
・Class diagram	△（depending on status）
・Object diagram	△（depending on status）
<Requirement analaysis>	
・Use case diagram	○
・Activity diagram	△ （depending on status）
・Sequence diagram	△ （depending on status）
・Interaction overview diagram	△ （depending on satus）
・State machine diagram	△ （depending on status）
<System analysis>	
・Class diagram	○
・Object diagram	○
・Sequence diagram	○
・Component diagram	△ （depending on status）
・State machine diagram	△ （depending on satus）
<Architecture design、Detail design> （To reassure the diagrams drawn up in previous process and to add more details)	
・Component diagram	△ （depending on status）
・Composite structure diagram	△ （depending on status）
・Class diagram	○
・Communication diagram	△ （depending on status）
・State machine diaram	△ （depending on status）
・Sequence diagram	○
・Object diagram	○
・Package diagram	○
・Deployment diagram	△ （depending on status）
・Timing diagram	△ （depending on status）

【Relevant information 】

Refer to Table 4.2 Example for each process output.

【Point 03　It does not have to be UML at all times.】

The entire output of system development does not have to be in UML only. Use non-UML diagram if it suits your purpose.

【Purpose】

The purpose of architecture is to deliver development objects more precisely and efficiently. It is not necessary to use only UML diagrams in system development to produce architecture documentation.

【Details ・ Supplementation】

Recognize UML diagram features and use appropriate diagram to demonstrate information. Use UML or non-UML appropriately because not every UML supports every output. In this case, it is not necessary to use UML. For example, some tables demonstrate clearer information depending on context. All the information does not have to be in UML model. The document should be in proper format whether it is UML or not.

Consider the following for using non-UML diagrams.

・ Screen transition diagram
・ Screen design documentation
・ Data model diagram（E-R diagram）
・ Data flow diagram（DFD）
・ Robustness diagram (UML extension)
・ Network diagram

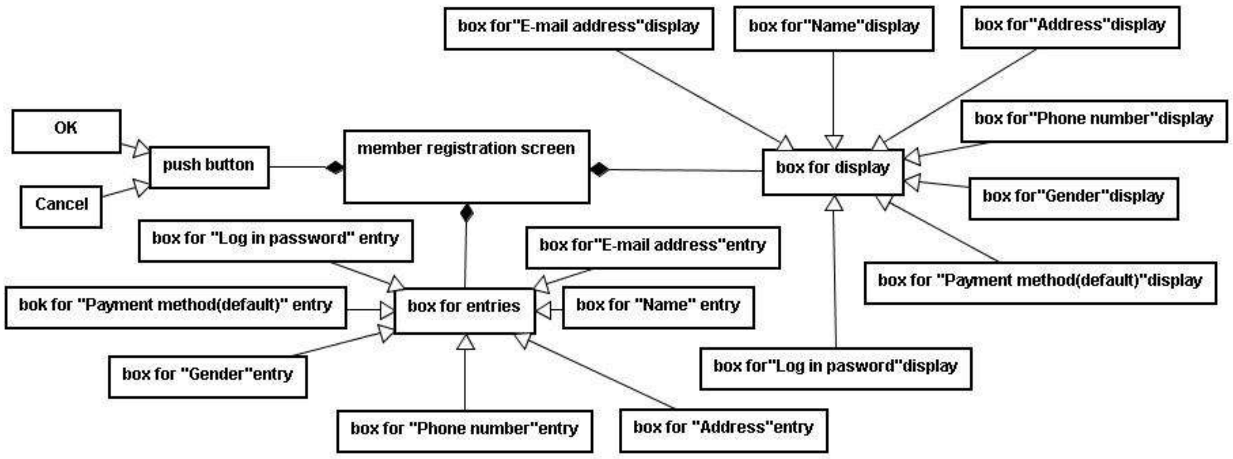
【Relevant information 】

Table 4.2 Example for each process output.

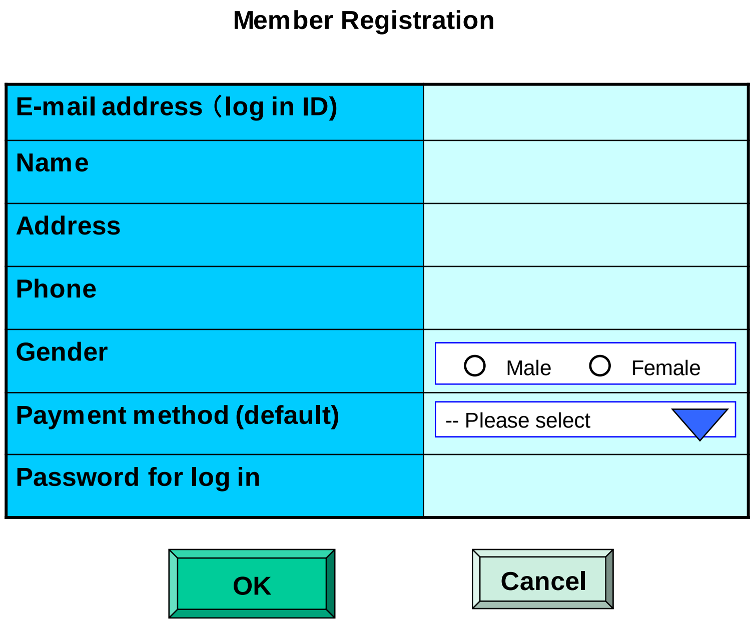
【Example that the technology other than UML may be more appropriate】

When you design the member registration screen, how you could do it UML class diagram is shown in the sample 1. Sample 2 is the screen sample designed and developed by the other screen design tools and Excel. It is obvious that the sample 2 is much easier to understand rather than the sample 1 when you design screen layouts, when you make sure of the design or review them. It is suggested not to use UML to design screens and reports, and use a diagram in the sample 2 alternatively.

<Sample 1>


box for"E-mail address"display
box for"Name"display
box for"Address"display
box for"Phone number"display
OK
push button
member registration screen
box for display
Cancel
box for"Gender"display
box for "Log in password" entry
box for"E-mail address"entry
box for "Payment method(default)"display
bok for "Payment method(default)" entry
box for entries
box for "Name" entry
box for"Log in pasword"display
box for "Gender"entry
box for "Phone number"entry
box for "Address"entry

<Sample 2>



Member Registration

E-mail address（log in ID)	
Name	
Address	
Phone	
Gender	○ Male ○ Female
Payment method (default)	-- Please select ▼
Password for log in	

OK **Cancel**

Option

【Point 04 Participate early development process (optional) 】

It is desirable that offshore system engineer (SE), who’s command of Japanese language are high, to participate high-level system design process which is conducted in Japan.

【Purpose】

Drive after detail design efficiently if offshore members develop process after detail design.

【Details ・ Supplementation】

Offshore members have difficulty in understanding the entire system because The Japanese project team performs the high-level system design process and offshore members perform the detail design. Therefore, it is necessary that offshore members participates the high-level system design process and understand the purpose of the development system and the contents to avoid any mistakes in the detail design process and its successive processes. It would also be possible to request the Offshore Partner to execute Join Test partially as some of the Offshore Partner members understand the entire system design. In this scenario, it would be ideal that just some of offshore members participates the high-level system design process would be depending on the system size.

Most importantly, Owner (Japan) should make a fair judgment whether or not the Offshore Partner has qualified engineers who could possibly participate the high-level system design process. Then, the Owner could consider expanding the areas that the Offshore Partner could help.

【Relevant information 】

Point 01: Define specific scope of work, roles and responsibilities

Point 06: Understanding business process by analysis models

Point 11: Architecture model

【Point 05 Non-functional requirement definition】

Define the non-functional requirements. Non-functional requirements are important factor to determine how to make required functions (system features) available.

【Purpose】

Non-functional requirements are important for architecture design and have to be defined clearly before the architecture design. Non-functional requirements are entered when the architecture is designed.

【Details ・ Supplementation】

There are functional requirements and non-functional requirements in the system requirements. Indicate functional requirements clearly in usage case diagram and define non-functional requirements. Non-functional requirements are described as shown below.

・ Development/system requirements (hardware/software)
・ Performance requirements
・ Reliability requirements
・ Cost (short/long terms)
・ Security
・ Maintenance
・ Operational requirements

This particular section covers non-functional requirements in the system, not just what to be covered by the Offshore Partners. How project members typically view things may not be equally interpreted by the Offshore Partners. Therefore, it is extremely important to define clearly what the output goals should be including non-functional requirements.

【Relevant information 】

Point 11: Architecture model

【Point 06 Understanding business process by analysis models】

Create Analysis Model to help project members understand business process visually.

【Purpose】

Using Analysis Model helps engineers to understand business process visually at high-level. So that, it is expected that Owner (Japan) should create the Analysis Model as the Owner is supposed to be familiar with system requirements in detail. Model description can be written in Japanese. However, it is expected to be brief, not using long sentence. Using models help Offshore Partners' engineers who may not be fluent in Japanese, and it will help both Owner and Offshore Partner to communicate accurately.

【Details ・ Supplementation】

Analysis Model uses mainly class diagram and presents business processes. Analysis Model does not explain how business processes can be systematically design. It does not necessarily have to be in detail. Models should be developed for common understanding among project members. This also intends to avoid anything ambiguous. It will help any successive design processes to avoid any step-backs. Offshore Partners should be advised to ask questions as early as possible if models cannot be interpreted completely or they have any other questions.

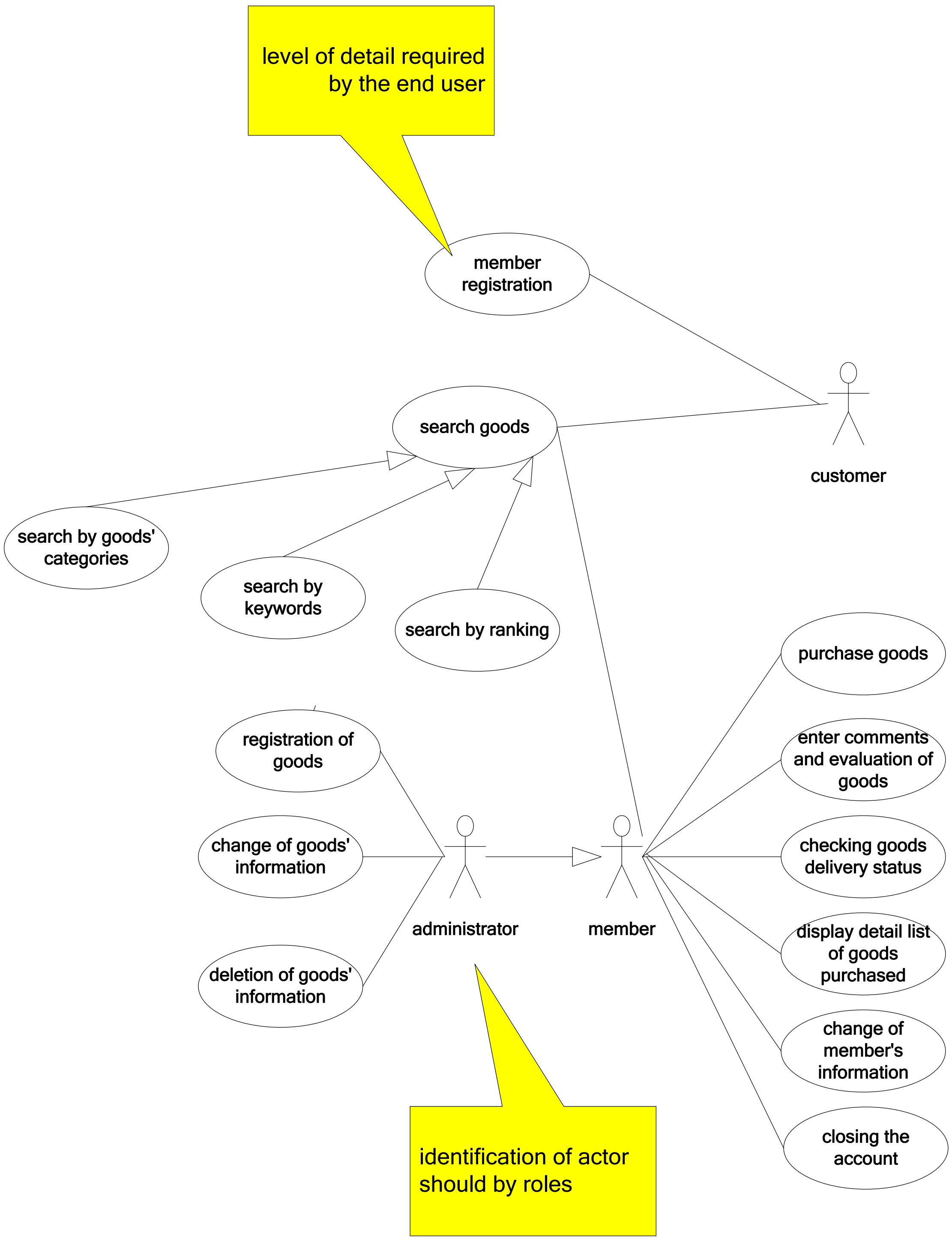
【Relevant information 】

Appendix, Sample Output: (1) Output of each process ③Process: Systems Analysis

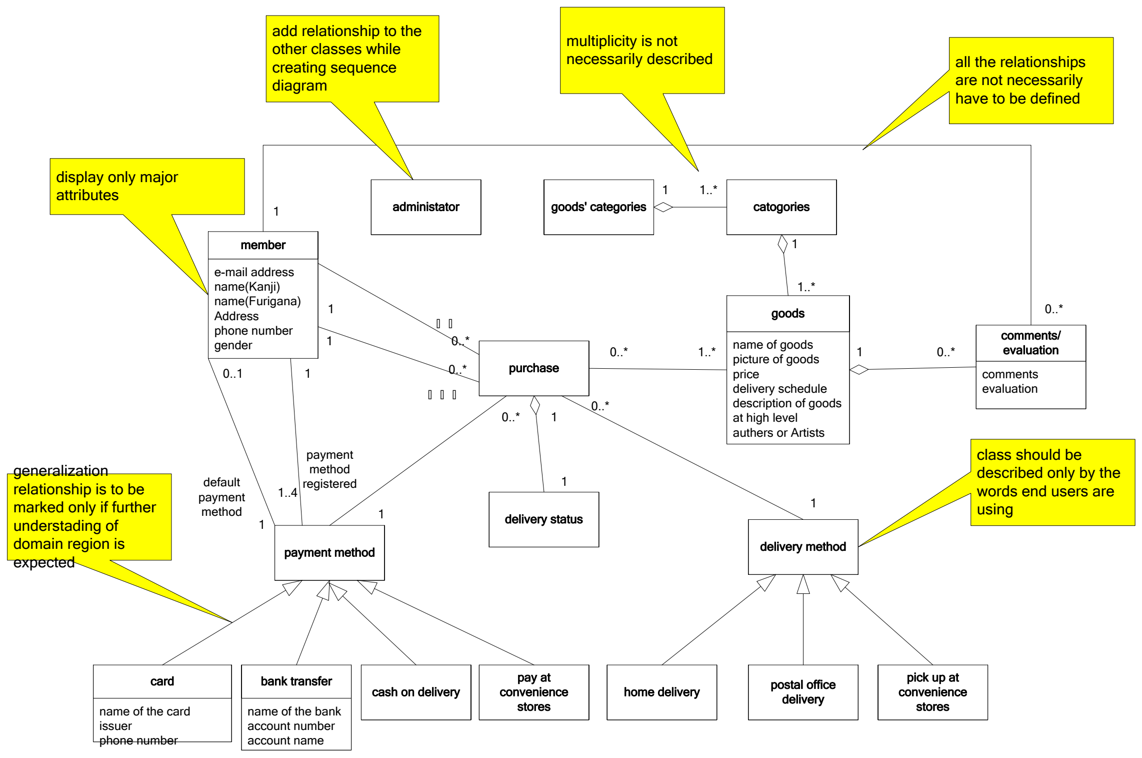
【Sample of analytic model to understand business operation】

Understanding business operation by use case diagram and class diagram at analytical level.

Example of use case diagram


level of detail required by the end user
member registration
customer
search goods
search by goods' categories
search by keywords
search by ranking
purchase goods
enter comments and evaluation of goods
registration of goods
change of goods' information
deletion of goods' information
administrator
member
checking goods delivery status
display detail list of goods purchased
change of member's information
closing the account
identification of actor should by roles

Sample of class diagram


add relationship to the other classes while creating sequence diagram
multiplicity is not necessarily described
all the relationships are not necessarily have to be defined
display only major attributes
member
e-mail address
name(Kanji)
name(Furigana)
Address
phone number
gender
administator
goods' categories
catogories
goods
name of goods
picture of goods
price
delivery schedule
description of goods at high level
authers or Artists
comments/ evaluation
comments
evaluation
purchase
delivery status
payment method
delivery method
default payment method
payment method registered
generalization relationship is to be marked only if further understading of domain region is expected
class should be described only by the words end users are using
card
name of the card issuer
phone number
bank transfer
name of the bank
account number
account name
cash on delivery
pay at convenience stores
home delivery
postal office delivery
pick up at convenience stores
1
1..*
0..*
0..1
1..4

＜Notice＞

- Should be created from end user point of view (operational point of view), and should not be too much detail.
- Name of use case should be the name of operation in use, and name of class, name of attributes should be the words used in the real operation.

【Point 07 Creating glossaries】

Create glossaries. It is efficient to classify common terminology and specified business terminology. Generate it not only in Japanese but also in the offshore member's language (for example, Chinese or English in the case of China). Include offshore member's language for specified business terminology to deliver the meaning more precisely.

It is strongly suggested to use English, and not to use Katakana.

【Purpose】

Improving offshore project members understanding of tasks and consistency. The offshore project team will have clear naming rule to improve the followings.

・Understand design documentation from the Japanese project team

・Prepare specification generation ability

・Use naming rule when the process is accepted and verified

【Details ・ Supplementation】

Terminology and contents are used to improve comprehension of tasks. Update version if glossary is changed and notify all members of the update. If possible, the Japanese project team and the offshore project team are informed that the glossary version is updated in real time.

【Relevant information 】

Point 08: Creating naming rules

UML Modeling Glossary

http://www.umtp-japan.org/modules/data4/index.php?id=2

【Point 08 Creating naming rules】

Create naming rules. Use naming rule standard needs to be created in the system analysis stage at least.

【Purpose】

Produce the standard terminology in order to achieve the efficient requirement analysis and system analysis process output in development and improve the later process output. Offshore project members recognize naming rules and improve the following.

・Understand the design provided by the Japanese project team

・Specification quality of offshore project team

・Use naming rule when the process is accepted and verified

The Japanese project team also improves their review and design with misusing Japanese and lack of description.

【Details ・ Supplementation】

It is convenient to generate rule to decide which language (Japanese or Chinese), digits/figures, and row classification will be used. It is also effective to describe them in both Japanese and English. The following items decides the naming rule.

・ Subsystem/component name

・ Package name

・ Use case name

・ Class name

・ Attribute name and property name

・ Operation name and method name

・ Event name

・ Message name

・ Status name

・ Screen ID、screen transition ID

and more.

【Relevant information 】

Point 07: Creating glossaries

【Point 09 Creating modeling rules.】

Create modeling rules same as coding rules on implementation.

【Purpose】

Generate modeling rule for consistency of models made by the project members. It helps maintenance.

【Details ・ Supplementation】

UML elements (class, package, and message etc.) are classified by explicit modeling rule and the meanings are more accurate. For example, class is divided into BCE (Boundary、Control、Entity) to perform system analysis, architecture design and detail design generally. Boundary indicates other system mediating class; control indicates flow control class; and entity indicates existing class in reality. In addition, development project or exclusive rule in a business unit such as data delivery class, database exchange class, or time constraint message can be generated for better communication between project members.

It is also advised to define level of authorization for changes in Models. For example, it could be authorized for Offshore Partners to use “private” operation, subject to that Offshore Partners needs approval from Owner (Japan) to use “public” operation.

UML extension function can be used when a modeling rule is generated if necessary. UML has UML profile as a UML extension tool. UML extension by UML profile uses stereotype, limitation, and meta-attribute specifically.

【Relevant information 】

Point 07 Creating glossaries.

Point 08 Creating naming rules.

【Point 10 Explicit common function】

Check for common functions in UML modeling process. The offshore project team can produce generalization based exclusively on the architecture by Japanese project team when Japanese project team produces the design/implementation.

【Purpose】

Improve quality and development efficiency. Solution is required for the intention such as the followings of offshore project members.

・ Offshore project members copy and modify samples provided, or from first development output repeatedly and consider the modification as another output.

・ Similar functions considered as common component/function/module can be generated from the result of copied design or implementation.

【Details ・ Supplementation】

Offshore project team generates common functions based on sample exclusively for efficient internal development. It means they copy samples and produce again after generating the task sequence. The Japanese team should provide architecture and guideline in advance and perform an analysis review in an early stage. Verify if common functions are generated concisely and the components are assigned properly. For this, include the following.

・ Task item to verify common function in planning task process.

・ Task item to check common function included in reviewing.

【Relevant information 】

None

【Point 11 Architecture model】

Architecture Model means specific design of hardware and software basic structure. It is strongly advised to create comprehensive architecture model and make sure that the models are equally understood among project members.

【Purpose】

Understand tasks to verify that the hardware and software architectures satisfy the client’s business request. That being a situation, it is advised that Owner (Japan) creates Architecture Model. Architecture should be designed with future system extension in mind or change in a long term because architecture change causes design/specification change as well.

There is a chance that any system enhancement or maintenance in the future may ends up failing because of the insufficient understanding of system architecture of Owner (Japan) if Offshore Partner is assigned to design system architecture.. In this situation, appropriate changes and enhancement in the future may not be possible.

【Details ・ Supplementation】

The Japanese project team prepares the system hardware/software/architecture for the development project. In this case, consider the followings.

・Is the architecture reasonable for functional/non-functional aspect?
・How to implement the requested performance/capacity?
・How to solve the restraints from operational/test environment?
・How to perform operational requirements including trouble shooting?
・How to solve the security issues?
・How to manage development member’s skills?

The Japanese project team designs the architecture model and forwards it to the offshore project members as output. It will be helpful to prepare a guideline including ‘development sequence according to hardware architecture and software architecture’ in case of need.

【Relevant information 】

Point 12: Create output of architecture description
Appendix, Sample output: (1) Output of each process ③Process: Architecture design

【Point 12 Create output of architecture description】

Draw up output of system software architecture description.

【Purpose】

The Japanese and offshore project team members both equally understand the development system software architecture when the process moves from requirement analysis ~ architecture design of the Japanese project team to detail design of offshore team in offshore development. The Japanese project members should send information about the structures and/or ideas that the development system is based upon.

【Details ・ Supplementation】

Use supplementation as process input to verify the software architecture and draw up output from the architecture result. The architecture description output consists of class diagrams and other supplementary documents in system analysis ~ architecture design level.

Each diagram demonstrates ground generating design, decision points, or the intention/purpose of the hardware/software architecture of development system in functional aspect. You can use component diagrams, package diagrams, and/or deployment diagrams if necessary. Description in Japanese can be delivered concisely to offshore project members if you use diagrams of models.

The followings are as outputs on architecture modeling process.

・Input：non-functional requirement definition (Supplementary specification)

・Output：architecture description, detail design guideline

It is not satisfactory if Architecture Model and Architecture Description have just been created. It is even more important and required to make sure that these are in place and used.

【Relevant information 】

Point11 Architecture model

Appendix, Sample output: (1) Architecture description

【Point 13　Using patterns】

It is recommended to use any Patterns, which are known well in architecture or design, either in architecture design phase or detail design phase.

【Purpose】

Using Patterns commonly known among project members accelerates understanding of each other, and helps providing more information for purpose and responsibilities even without sufficient explanation. This equally means there are more information for better communication. This is extremely effective to communicate intention of architecture in the architecture design phase, or to communicate roles and intentions of Components and Classes in the detail design phase.

【Details ・ Supplementation】

It is often happens that design patterns in the software development is explained in combination with UML. Know-how in system design will be incorporated into design patterns which are typically shared by engineers. Re-use or application of these design patterns contribute s sustaining quality and brings common understanding successfully by design patterns in the communication in design phase or any review phases. Such architectural level patterns in the architecture design phase as MVC (Model, View and Control) helps project members responsible for detail design to better communicate what the design intends. Such patterns used in the detail design phase as GoF (Gang of Four) helps engineers to better assume roles and responsibilities of each part of the process designed in any review processes. This also helps checking actions as the points to be checked are effectively summarized. As such, suing patterns are always helpful for all the project members to better understand each other's work and outcome of it.

Mastering Patterns and usage of it is not mandatory in the Offshore Development. However, it is truly believed that it helps communications among project members and fulfills skill level gap among them. So, it is viewed that learning Patterns is certainly recommended.

【Relevant information 】

Point 06: Understanding business process by analysis models

Point 11: Architecture model

Point 12: Create output of architecture description

【Point 14 Level of description of specification, format specification】

Specify level of description of UML and format specification for each respective specification which is supposed to be the outcome of off-shore partner.

【Purpose】

Sample artifacts are often required in the off-shore development, not just in UML. Lack of detail direction from Japan to the off-shore partner's project members regarding how they should describe specification will create inconsistent way of description and inconsistent format among the off-shore partner's project members. It is also recognized that there is not that much effort by the off-shore partner's project members to try to make description level and format consistent.

It should be considered depending on the situation that the artifacts are to be presented with the customers or to be reviewed by the Japanese project members.

【Details ・ Supplementation】

Regarding UML contained in the artifact, Japan members create samples based applying the detailed description and granularity of the models required by the Japanese members, and provides them as part of the guideline. It is desirable to direct specific purposes of the model such as what is expected to be done by the UML and what is to be shown by the UML. This helps the team to be consistent in the level, accuracy, and granularity of UML.

Processes handling exceptions and errors need to be included in the sample appropriately. (The off-shore partner expects that it is included in the sample at all times. So that it is not likely included in the final artifact if not included in the sample.)

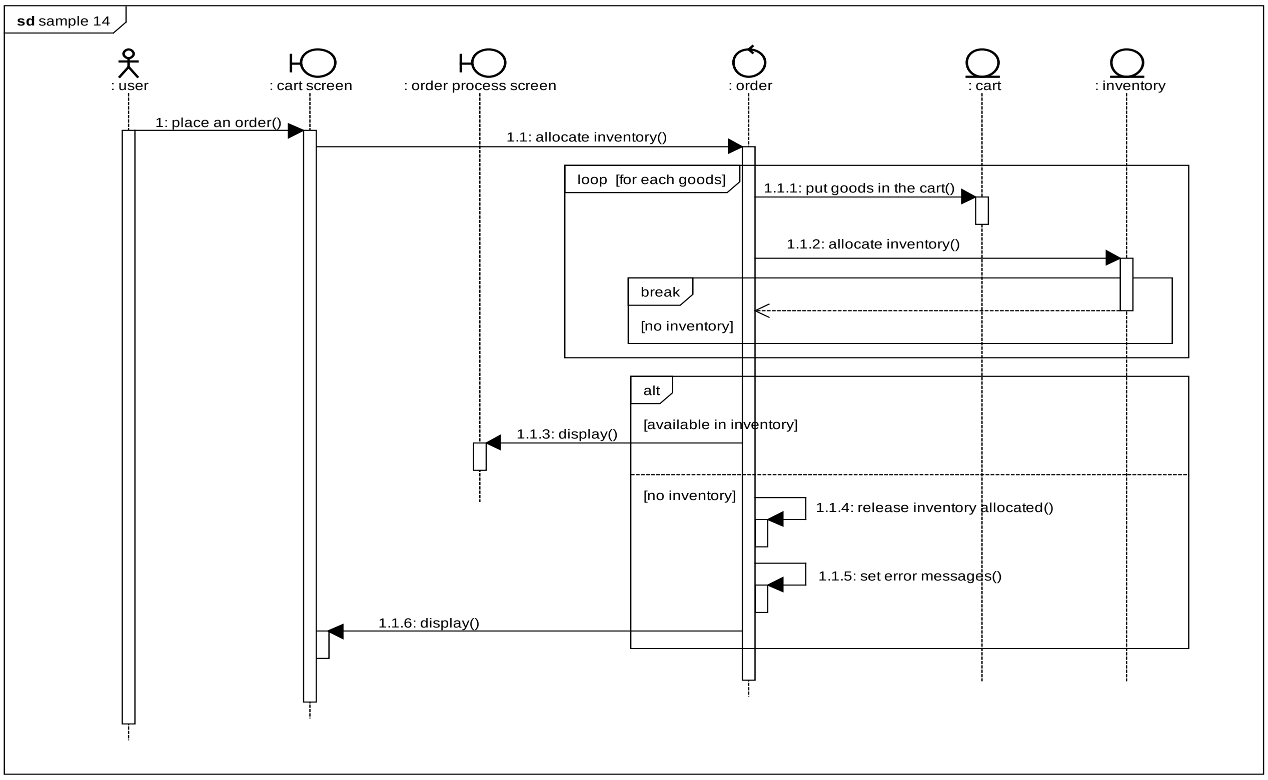
Such other points in descriptions for the artifacts to be cared should include 1) any expressions should be specific, not vague, 2) words and terms used should be consistent, and 3) words and terms in Katakana should be described in English.

【Relevant information 】

None

【Example of exception/error handling】

This system allocates the inventory when button of placing order is pressed, not when the goods are put in the cart. If inventory are allocated to all the goods in the cart, it proceeds to the order confirmation screen. If inventory is not allocated to even one item in the cart, all allocated inventory should be released, return to the original screen and error message should be shown.



・Notice

1) It happens that the process satisfies specific condition is described (i.e., process inventory is allocated in the case above) and that process not satisfying specific condition is not described (i.e. process is not allocated in the case above). Accordingly, sample specification and diagram should clearly state that both processes are required to be described.
2) It needs to be recognized that error handling process (release of inventory allocated in the example above) is not developed for production if it is not described in the specification.
3) Error handling process requires detail and specific description what to do (setting error messages before returning to the cart screen, in the example above)
4) Production environment and process depended on implementation should also be described as required (such as stating error message id clearly, adding additional object where error messages are sent)

【Point 15 Creating detail design guideline】

Present undecided specification of output precisely to offshore members.

【Purpose】

The Japanese project team sends a development request with some undecided specifications for 're-quirement analysis ~ system analysis ~ architecture design process' to offshore project team due to the lack of time in some cases. The undecided specification can result in lots of questions from the offshore project members because the output of the Japanese project team is inconsistent in many cases.

【Details ・ Supplementation】

Define which specification has been undecided for the offshore project members to avoid any specifi-cation loss. It is often connected to specification change and specification missing.

Use notes or annotation including the following about undecided specification that is changeable or undefined in UML output.

・Which domain has undecided specification and which parts can be affected?

・What kind of change do you expect in the current specification? (Will it be more specific or changed to different contents?)

The Japanese project members should manage each item to avoid any loss or misleading change of offshore development parts when specification is decided or changed. In addition, check how offshore members reflect the current change or later decided specification through the offshore company's minute book and meeting minutes to confirm the development progress.

【Relevant information 】

Appendix, Sample output: (1) Detail design guideline

【Point 16 Clarifying areas of which specifications are not determined】

Areas in the output of which specifications are not yet determined should be clearly presented to the Offshore Partners. Offshore Partners should not proceed if specifications are not clearly com municated and should confirm any undefined areas of specifications with the Owner (Japan).

【Purpose】

Verify output review. If you inform the offshore members that the Japanese project members are going to review the offshore output, sometimes the offshore members just send their output without re-viewing it because they think 'it is pointless to review it twice in both Japan and offshore sites'.

【Details ・ Supplementation】

Make agreement between the Japanese and offshore teams on output review including object to be reviewed, check point, and acceptance level performed by offshore project team. At the time, define the difference of the review purposes between offshore company and The Japanese project team to im-prove the review of the offshore company explicitly.

Basic review points are;

・ Check whether all the functions on development function plan are included.
・ Check whether all the specification in the entire design process is described.
・ Check whether the designated description level and notation are followed.
・ Check whether the consistent terminology is used precisely.
・ Check whether the expected static development structure is correctly delivered, designed, and de-scribed.
・ Check whether the expected dynamic development structure is correctly delivered, designed, and described.
・ Check whether the expected physical alignment is correctly delivered, designed, and described.
・ Check whether the class responsibility is unclear.
・ Check whether the diagram consistency is ensured.
・ Check whether the specification change is reflected.

【Relevant information 】

None

【Point 17 Reviewing output by the offshore partner】

Start to check when the offshore project members kick off their development.

【Purpose】

The following can happen because communication is not normally in the offshore project member's language in offshore development.

・ The Japanese project team has difficulty ascertaining how much the offshore project members understand the project.

・ The offshore project team cannot recognize their misunderstanding.

Prepare against any misunderstanding between the Japanese and the offshore project members due to miscomprehended and unconfirmed issues.

【Details ・ Supplementation】

The Japanese project team checks the offshore team's status including the following often after the offshore project members starts their development.

a) Check whether the output description is accurate.

b) Check whether the output notation standard is followed.

c) Check whether the design contents are understood correctly.

The Japanese project members should verify continuously the review results of the offshore project members in the beginning of the process and during the process.

Furthermore, give advice to the offshore project members to verify development output and comprehend each other during the development period. Offshore Partners should create Review Reports at all times and the Owner (Japan) should review it as required.

【Relevant information 】

Point18 Repeating check by the owner (Japan)

【Point 18　Repeating check by the owner (Japan)】

The Owner (Japan) should implement appropriate checking process immediately after the offshore development effort has been started.
*It is to be noted that “check” hereby means, level of understanding and recognition, not just any sampling checks or as opposed to “review” which is typically conducted for more formal authorization and confirmation..

【Purpose】

You can see inconsistent output often in offshore development even though it is not in UML. Offshore project members frequently ask about inconsistent output from the Japanese project team. Sometimes both teams cannot review their outputs due to the inconsistent output. It is required that all the necessar y information should be contained in output without expecting readers to guess.

Offshore Development projects need to be closely reviewed as there might not be any output in time or quality of outputs are not satisfactory. These effort needs to be closely monitored. Checking process by the Owner (Japan) helps better quality and save workload. It is highly recommended that the Owner should allocate appropriate number of days retain resources for this particular checking effort.

【Details ・　Supplementation】

In dynamic aspect

・Mutual class diagram/sequence diagram/communication diagram
・Mutual sequence diagram/state machine diagram/ activity diagram

In static/physical aspect

・Mutual class diagram/package diagram/object diagram/ component diagram/ deployment diagram

It is suggested that the Owner (Japan) executes review process of which done by Offshore Partners in early stage of projects. The Owner (Japan) should check appropriate outputs at appropriate timing of the project rather than checking all the detail outputs.
The Japanese project team should verify through detail design review the diagram consistency with counterparts in several relevant diagrams as above. Also ensure if each diagram uses consistent terms for the same subjects. Inconsistency, as shown below, causes implementation failures.

・ Some sequence operation causes unreadable recorded objects from class diagrams.
・ Some state transition trigger is unreadable from sequence diagram or activity diagram.
・ Component to be aligned in some node is unreadable from class diagram or component diagram.

【Relevant information 】

Point 17 Reviewing output by the offshore partner

【Point 19　Validation and verification】

It is recommended to have two views. One is to validate the project for the Owner (Japan) to achieve its goal, and the other is to verify if things are successfully implemented by the Offshore Partner.

【Purpose】

Roles and responsibilities are clearly defined in the Offshore Development project. Accordingly, it is expected that the Owner (Japan) tends to focus on customer's business analysis through system analysis, while Offshore Partners focus on detail design and unit test to satisfy needs defined by the Owner. This different view occasionally creates misunderstanding in testing scope and functional requirement. It is important to verify whether or not end customer's needs are all satisfied. It is equally important to validate if implanted system satisfies end customer's needs.

【Details・ Supplementation】

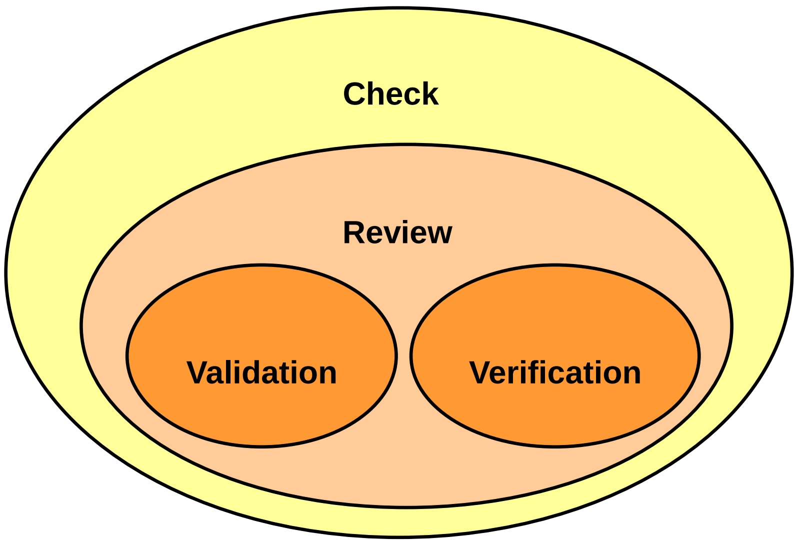
The Offshore Partner tends to focus on verification to make sure that the system was developed to exactly meet with customer's needs, required specifications and functions are all satisfied by cross checking programs and result of running those during unit test or join test processes. On the other hand, the Owner (Japan) wants to validate many points whether or not the production system satisfies end customer's business needs.
These different viewpoints between the Owner and Offshore Partner should be recognized and following actions will be required in addition to the review done by the Offshore Partner which is primarily "verification".

- The Owner should let the Offshore Partner know as early as possible during the project what exact tests the Owner wants Offshore Partners execute, from validation point of view
- The Owner should prepare for validation viewpoint tests to execute which should be conducted right after the Owner has received the outputs from the Offshore Partner.

For example, Traceability Matrix, which is to verify how each required functions are implemented and incorporated in the system by streamlining from needs analysis through implementation, is very useful and it should be very effective once project members have been qualified from skill point of view.

This guideline defines "review", "check", "validation" and "verification" as below.

Review process which targets to authorize the output can be interpreted either in “ it is ideal to be reviewed by such two points of view as validation and verification, or “it is ideal to execute variety of checking process to better monitor the progress of development project coordinated re motely, not just defined process of output review.”.



【Relevant information 】

Point 01:	Define specific scope of work, roles and responsibilities
Point 17:	Reviewing output by the offshore partner
Point 18:	Repeating check by the owner (Japan)

【Point 20 Reviewing detail design by the model】

Class Diagram, Sequence Diagram, Object Diagram, and Package Diagram are to be reviewed for detail design.

【Purpose】

Directions such as system sequence and business flow need to be confirmed by the Model before making it production. It is not necessary to follow source code. Review can be completed by the Model (such as sequence diagram) because it is visual and is easy for every project member. Owner (Japan) can also review the detail design by the Model, which will improve quality of the system in production.

【Details ・ Supplementation】

Reviewing in offshore development is more important than any other usual development. However, the Japanese project members can have many numbers of review processes if you make too many detail items to check. Therefore, it is useful to use modeling.

The Owner's (Japan) project members create Architecture Description and Architecture Guideline. The Offshore project members follow them and proceed to detail design. Then, they create such outputs as Sequence Diagram, Class Diagram, Object Diagram and Package Diagram.

The Owner (Japan) needs to review such outputs as Sequence Diagram, Class Diagram, Object Diagram and Package Diagram created by the Offshore Partner.

Important point in this particular review process is to refer Architecture Description to make sure that Sequence Diagram, Class Diagram, Object Diagram and Package Diagram created by the Offshore Partner are consistent with the architecture specified.

【Relevant information 】

Point 09: Creating modeling rules
Point 12: Create output of architecture description
Point 15: Creating detail design guideline
Point 21: Consistency between UML diagrams

【Point 21 Consistency between UML diagrams】

Consistency between UML diagrams created by the offshore partner need to be carefully reviewed.

【Purpose】

It is common that there are inconsistency between outputs from the Owner and the Offshore Partner. This could happen to any outputs, not just UML. Inconsistency in the output from the Owner (Japan) creates confusions and questions from the Offshore Partner. Inconsistency in the output from the Offshore Partner creates difficulties for Japanese project members to understand the output correctly. Both organizations end up not being able to review their output each other. So, it is extremely important to describe as much detail as possible in the output. Creators should not ever expect Readers to understand the output in the way Creators expect without complete description.

【Details ・ Supplementation】

In dynamic aspect

・Consistency between Class Diagram/Sequence Diagram/Communication Diagram

・Consistency between Sequence Diagram/State Machine Diagram/ Activity Diagram

In static/physical aspect

・Consistency between Class Diagram/Package Diagram/Object Diagram/ Component Diagram/ Deployment Diagram

The Japanese project team should verify through detail design review the diagram consistency with counterparts in several relevant diagrams as above. Also ensure if each diagram uses consistent terms for the same subjects. Inconsistency, as shown below, causes implementation failures.

・ Some sequence operation causes unreadable recorded objects from class diagrams.

・ Some state transition trigger is unreadable from sequence diagram or activity diagram.

・ Component to be aligned in some node is unreadable from class diagram or component diagram.

【Relevant information 】

None

【Example of integrity between UML diagrams】

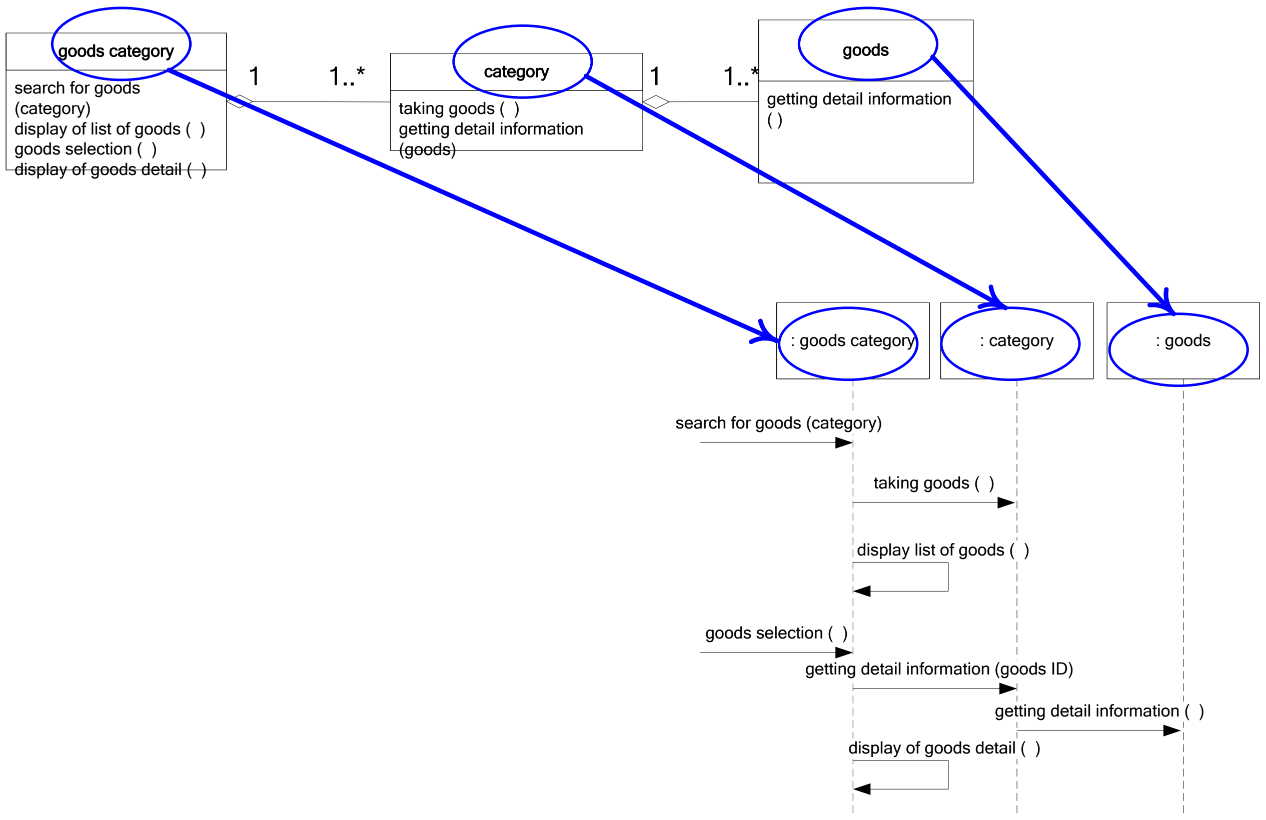
> Sequence diagram and class diagram

Special attention needs to be paid the following three integrities for sequence diagrams and class diagrams.

Example: Search for goods by goods category

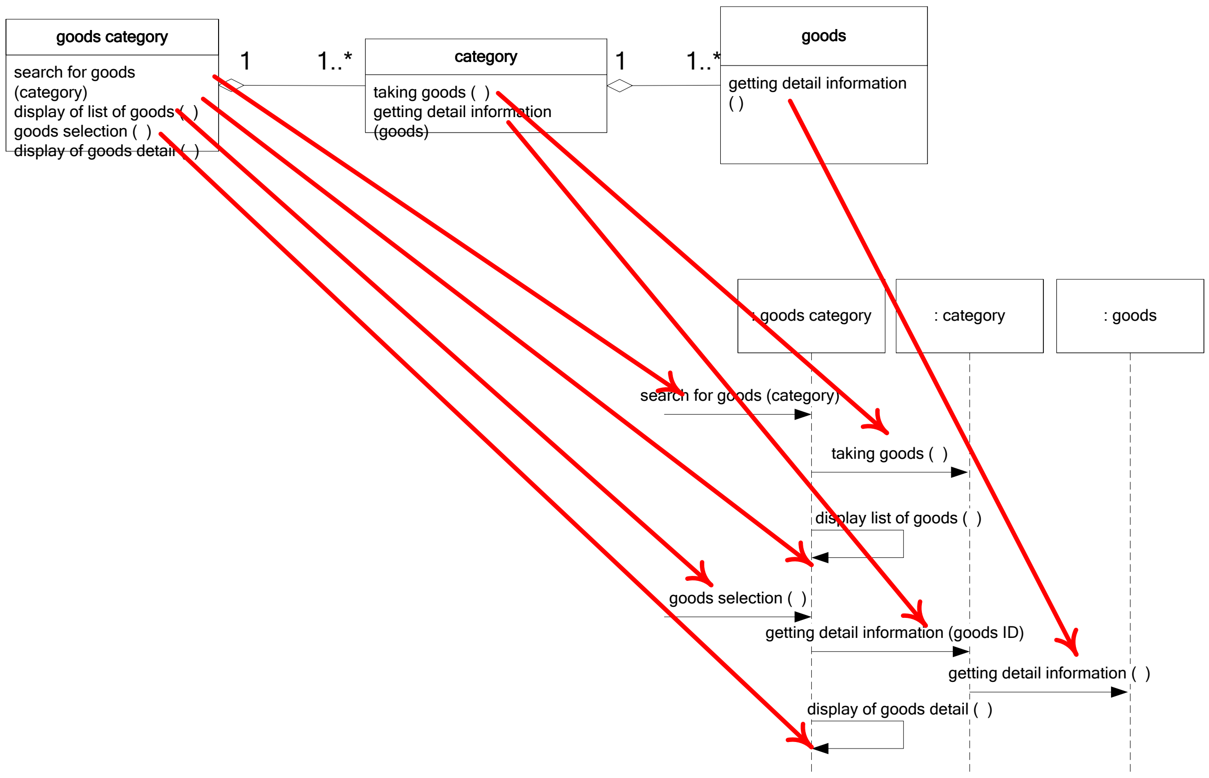
・Lifeline and class

Class in the class diagrams corresponding to the lifeline of sequence diagrams need to be defined.


goods category
search for goods (category)
display of list of goods ( )
goods selection ( )
display of goods detail ( )
1
1..*
category
taking goods ( )
getting detail information (goods)
1
1..*
goods
getting detail information ( )
: goods category
: category
: goods
search for goods (category)
taking goods ( )
display list of goods ( )
goods selection ( )
getting detail information (goods ID)
getting detail information ( )
display of goods detail ( )

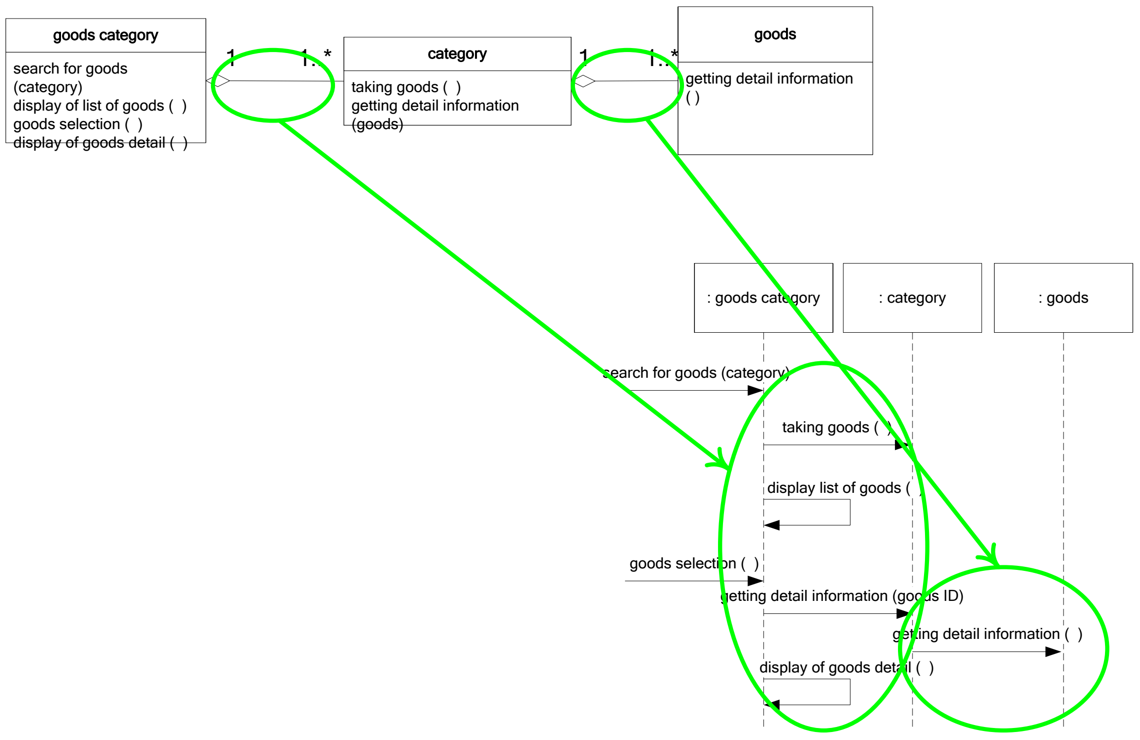
> Message and operation

In sequence diagrams, operation needs to be described in the class of the class diagram which corresponds to the lifeline where a message is called. Message call is possible by operation call.


goods category
search for goods (category)
display of list of goods ( )
goods selection ( )
display of goods detail ( )
1
1..*
category
taking goods ( )
getting detail information (goods)
1
1..*
goods
getting detail information ( )
: goods category
: category
: goods
search for goods (category)
taking goods ( )
display list of goods ( )
goods selection ( )
getting detail information (goods ID)
getting detail information ( )
display of goods detail ( )

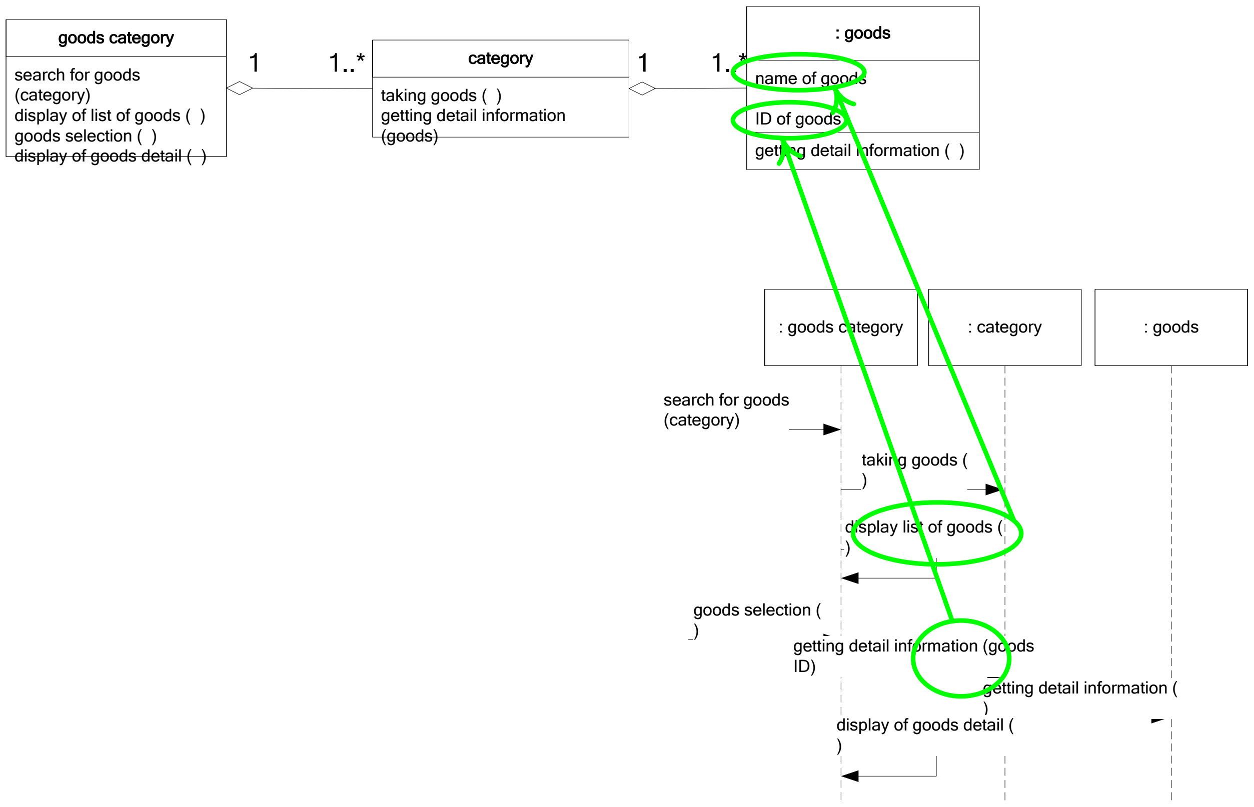
> Relationship with messages

In sequence diagrams, some relationships need to be described in between the class of the class diagram which corresponds to the lifeline where a message is called.



goods category
search for goods (category)
display of list of goods ( )
goods selection ( )
display of goods detail ( )
1
1..*
category
taking goods ( )
getting detail information (goods)
1
1..*
goods
getting detail information ( )
: goods category
: category
: goods
search for goods (category)
taking goods ( )
display list of goods ( )
goods selection ( )
getting detail information (goods ID)
getting detail information ( )
display of goods detail ( )

In addition to the three examples introduced earlier, necessity of attribute in the class diagram may be recognized occasionally.


goods category
search for goods (category)
display of list of goods ( )
goods selection ( )
display of goods detail ( )
1
1..*
category
taking goods ( )
getting detail information (goods)
1
1..*
: goods
name of goods
ID of goods
getting detail information ( )
: goods category
: category
: goods
search for goods (category)
taking goods ( )
display list of goods ( )
goods selection ( )
getting detail information (goods ID)
getting detail information ( )
display of goods detail ( )

Option

【Point 22 Using “code generation feature” in the tool for production (optional)】

It is suggested to create source code using the source code template created by the code generation feature in the tool for production, if appropriate models are available. Tools do not have to be used at all times. However, using tools brings higher productivity and more efficiency.

【Purpose】

In the process of detail design, the Owner (Japan) project team should review such outputs as Class Diagrams or Sequence Diagrams created by the Offshore Partner. This process makes the model more reliable as it was already reviewed and confirmed that there is no error in the model. Then, it is advised to use the code generation feature in the tool so that it makes the source code even more reliable.

【Details ・ Supplementation】

It would be possible to generate skeleton of source code by using code generation feature in the modeling tool for the Class. This makes possible to identify the context in the source code by the model (Class Diagram) because such information as class name, attribute name, operation name will be available in the source code.

It requires checking the process flow (primarily Sequence Diagram) manually as generic models typically do not generate source code specifically for the sequence diagram (some of the modeling tools provide the feature to run the source code in production and reversely generate the sequence diagram)

【Relevant information 】

None

【Supplement: Unique nature of systems development in Japan】

This particular section is to make Owner’s (Japan) unique nature clear in the systems development environment and intends to get rid of the gap between the Owner and the Offshore Partner.

【Purpose】

Systems development environment in Japan demonstrates the following unique nature of it.

・Project proceeds subject to repeated specification changes (people tend to believe that it would happen and unavoidable)

・Test coverage and management method must be complete and strict.

It is important to make the Owner’s (Japan) unique nature of systems development environment clear and reduce issues in the project by aligning the understanding of the Owner (Japan) and the Offshore Partner.

【System Design】

It is important to create common understanding between the customer and developer (Owner and Offshore Partner) that changes in specification may impact significantly negatively to the project q uality.

【Detail / Supplement】

The following might be considered reasons why changes in specification might happen.

① Common business practice or unique conditions in the customer industry (so to speak, common understanding in the customer industry) are occasionally not described in the specification. These typically become visible at delivery after all tests have been completed.

② Japanese customers tend to request in detail for screen layout and operation of the system.

③ Japanese customers typically request changes in specification even once it has been agreed.

④ The Owner (Japan) may occasionally ask the Offshore Partner to start their task (detail design) even before the final specification gets completed.

【Development/Test】

【Detail / Supplement】

The following request for the test should be satisfied.

① Test for all paths will be required

Variation or matrix of conditions, screen sequence and flow, complete code coverage (100% coverage expected).

② Evidence (test result) is fundamentally required.

・Evidence is required to clarify process pattern.

・Evidence is required if issues are recognized during development process.

③ Standard rates for tests items will be specified.

・Number of steps, number of screens, number of DB tables will be the base for standard rates for the test items..

・Rate of each such case as Normal, Abnormal, Boundary will be specified.

④ Number of bugs during test process will be specified

・Test will be requested again if number of bugs in normal system, abnormal system are below each respective standard rate.

・It will be requested to review the entire system if numbers of bugs are many.

⑤ Many reports and documents during test process will be requested to present.

・List of bugs, list of issues and problems, bugs creation curve, quality management report of test processes, and so forth.

Appendix A: Explanation of the examples used in the Points

Internet Shopping

A Company is considering starting selling a shopping system on the Internet.

> How to search for the goods

There are three ways to search for the goods.

1) Search by the goods category
2) Search by the keywords
3) Search by the rank (by sales volume)

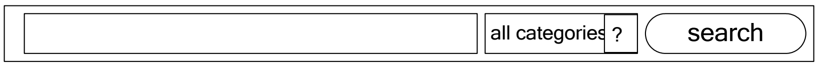
1) Search by the goods category

There are several goods categories (such as books, magazines, CD, DVD), and each category contains the following items

Books	**Magazines**	**CD**	**DVD**
・Animation	・Animation	・J-POP	・Foreign Movies
・Medical	・Music	・Animation	・Japanese Movies
・Entertainment	・Game	・Easy Listening	・TV Drama
・Music Melody Books	・Marriage	・Enka	・Music
・Science Technology	・Health	・Classic	・Animation
・Study Aid	・Commic	・Jazz	・Comedy
・Music	・Computer	・Pops	・Variety
・Education	・Magazines	・Rock	・Hobbies
・Living	・Cars	・Game Music	・Sports
・Economy	・Bikes	・Western Music	・idols
・Game	・Business		・Adults
・Engineering	・Literature		
・Languages	・Life Style		
・Children Books	・Cooking		
・Computer	・Travel		
・Qualification			
・Dictionaries/ Encyclopedias			
・Performer's photos			
・Geography			
・Business			
・History			

2) Search by the keywords

Select what to search, type in keyword, press search button, then goods hit the information will be shown.



The following all categories can be searched.

- All categories
- Books
- Magazines
- CD
- DVD

3) Search by the rank (by sales volume)

Choose what to search, press “rank” button, then goods will be shown by sales order sequence. The following all categories can be searched.

- All categories
- Books
- Magazines
- CD
- DVD

>Display of goods

Choose goods, display goods information, and then the following goods information will be shown. This is common for all the categories. Each category may have more information to be shown.

- goods name
- picture of goods
- price
- delivery schedule
- description of goods at high level
- description about the author or the artist
- customer comments and evaluation

>Member registration

Member registration is required to purchase goods, send comments or evaluations. For member registration, following information needs to be provided from the screen and register button needs to be pressed. The system returns a confirmation e-mail to the e-mail address registered. This confirmation e-mail has a URL which needs to be accessed to confirm the information registered, by pressing confirmation button.

- e-mail address (Log in ID)
- Name (Kanji, Furigana)

- ･address
- ･phone number
- ･gender
- ･default payment method (card, bank transfer, cash on delivery, convenience store)
- ･log in password

When registration is completed, members can log in by log-in ID and log in password.

- ･purchasing goods
- ･input comments and evaluation for the goods
- ･confirmation of goods delivery schedule and status
- ･detail list of goods purchased
- ･change of members information
- ･closing account

>Purchasing goods

Log-in is required prior to purchasing goods. You go to searching goods, list the goods to purch ase, press “cart” button. You can put as many goods as you want in the cart. “What’s in the cart” button can list all the goods in the cart. You can delete the goods that you really do not need. “Order” button will lead you to the ordering process. You can choose delivery method (ho me delivery, post office delivery, and pick up at convenience store) and payment method (card, bank transfer, cash on delivery, or convenience store). Default payment method chosen at member registration screen will be shown on this particular screen. Information typed in will be saved and can be used next time.

To complete ordering process, you should confirm list of goods purchased, total amount, then press “confirmation” button to confirm your order. Then, you will see purchase number on the screen, and receive an e-mail which describes purchase number and list of goods purchased.

If you pick up your goods at the convenience store, you need to inform this purchase number to the store clerk.

>Input comments and evaluations for the goods

Members can type in comments and evaluations for the goods by log-in. This input can be possi ble for any comments and evaluations for the goods which members do not buy. Comments can be up to 300 characters. Evaluations should be 5 scale from 1 to 5, of which “5” is the best evaluation.

>Checking goods delivery status

Members can confirm the delivery status of the goods purchased. After logged-in, member can pop up the goods delivery status check screen, choose the purchase number of the goods that members wants to check, and could see delivery status in more detail.

being prepared for shipping, on its way to the receiving store, delivered to the receiving store
already delivered, being prepared for shipping, being shipped, already delivered

>Display of list of goods purchased
You can display the list of goods purchased. You can even check the goods purchased and alre ady delivered. After logged-in, you can display the list of goods purchased, and choose purchase number of the goods you want to check, then list of the goods selected will be shown.
>Change of member's information
You can change your member's information already registered. After logged-in, choose change of member's information and then screen of member's information is shown. Make any changes you need to make and then update your member's information by pressing "update" button.
>Closing account
If you choose closing account after logged-in, confirmation screen is shown. Then you choose cl osing the account, and then your account will be closed. Once account is closed, all member inf ormation will be completely deleted. You will no longer be able to log in. All the features and fu nctions available for you as a member will no longer be available.

>Registering administrator
An administrator can be registered by the screen not available for public. You need to type in a dministrator ID and associated password.
Administrator can benefit the same features as members can. Administrators can also use some additional features as follows:

- Registration of goods
- Change of goods' information
- Deleting goods information

>Registration of goods
Administrator can register the goods by logging in through the screen not open to public.
Administrator chooses the category and input the following information. Some categories have more information in addition to the following. These information needs to be registered as required.

- name of goods
- picture of goods
- price
- delivery schedule
- description of products at high level
- description about the author or the artist
- customer comments and evaluation

>Change of goods information

Administrator can change the goods information by logging in through the screen not available to the public. First you need to choose the goods to make any changes. There are a couple of ways to make these changes.

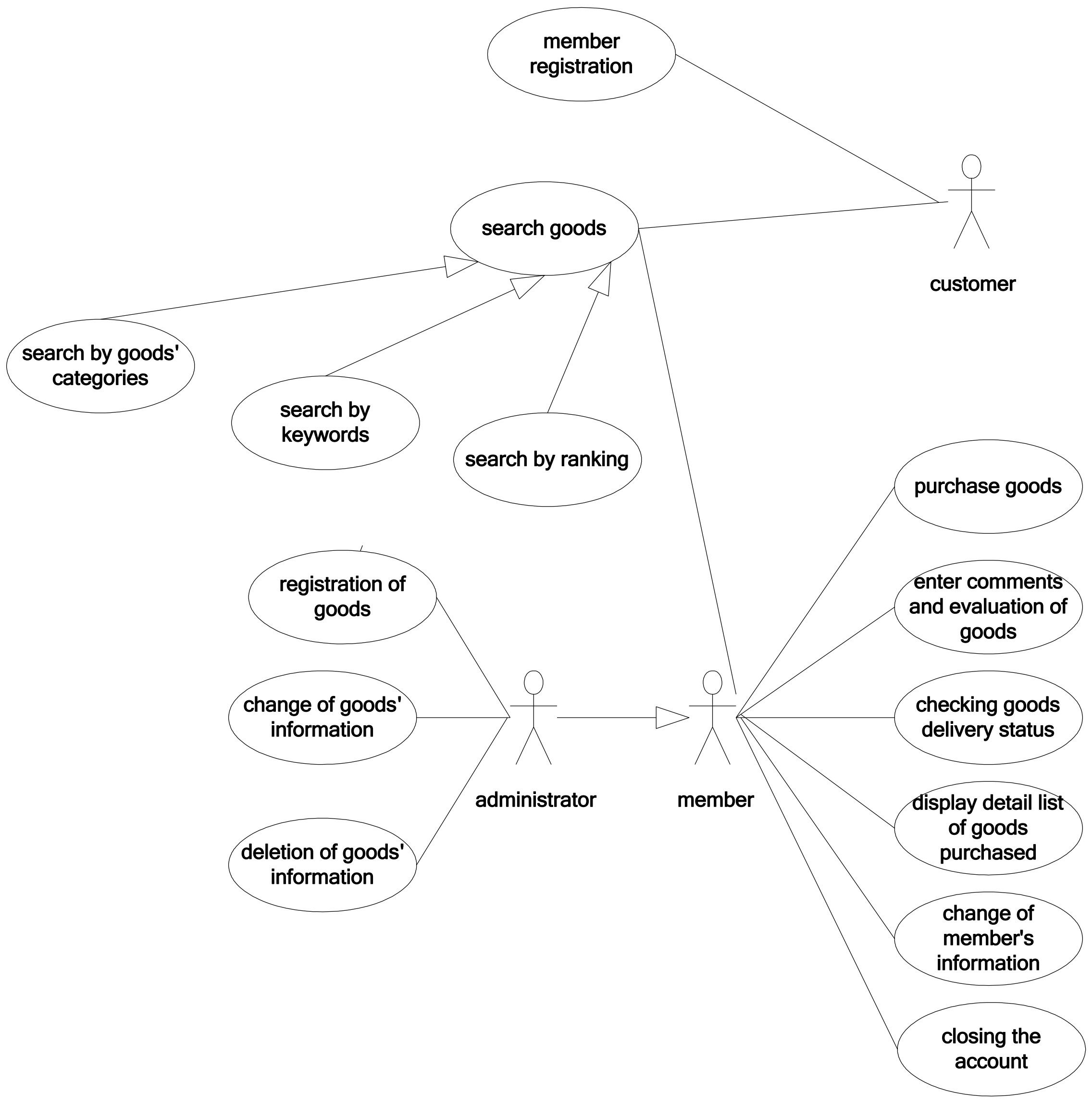
- choose goods category and select targeted goods from the list of goods in the category.
- search for the goods from search screen, and select targeted goods.

Goods already registered will be shown. Make necessary changes and apply these changes by pressing ”update” button.

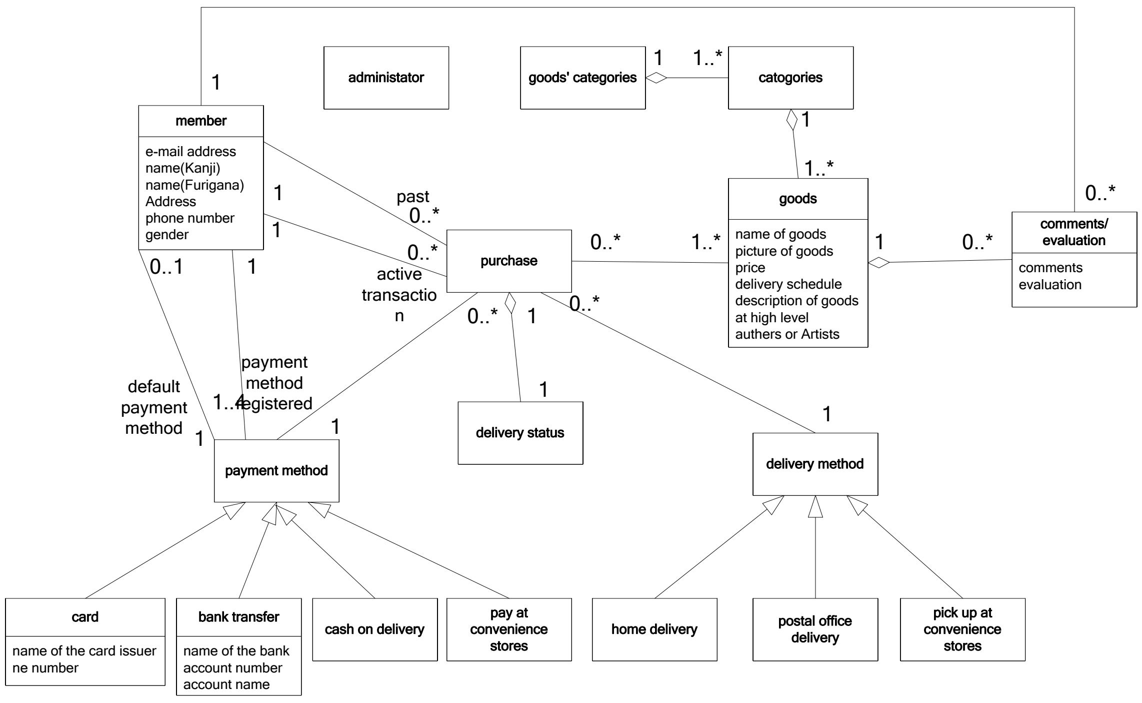
>Deleting goods information

Administrator can delete goods information by logging in through the screen not available to the public. Just like to change goods information, choose the goods and press “delete” button to del ete goods information. Confirmation screen pops up and the goods information gets deleted.

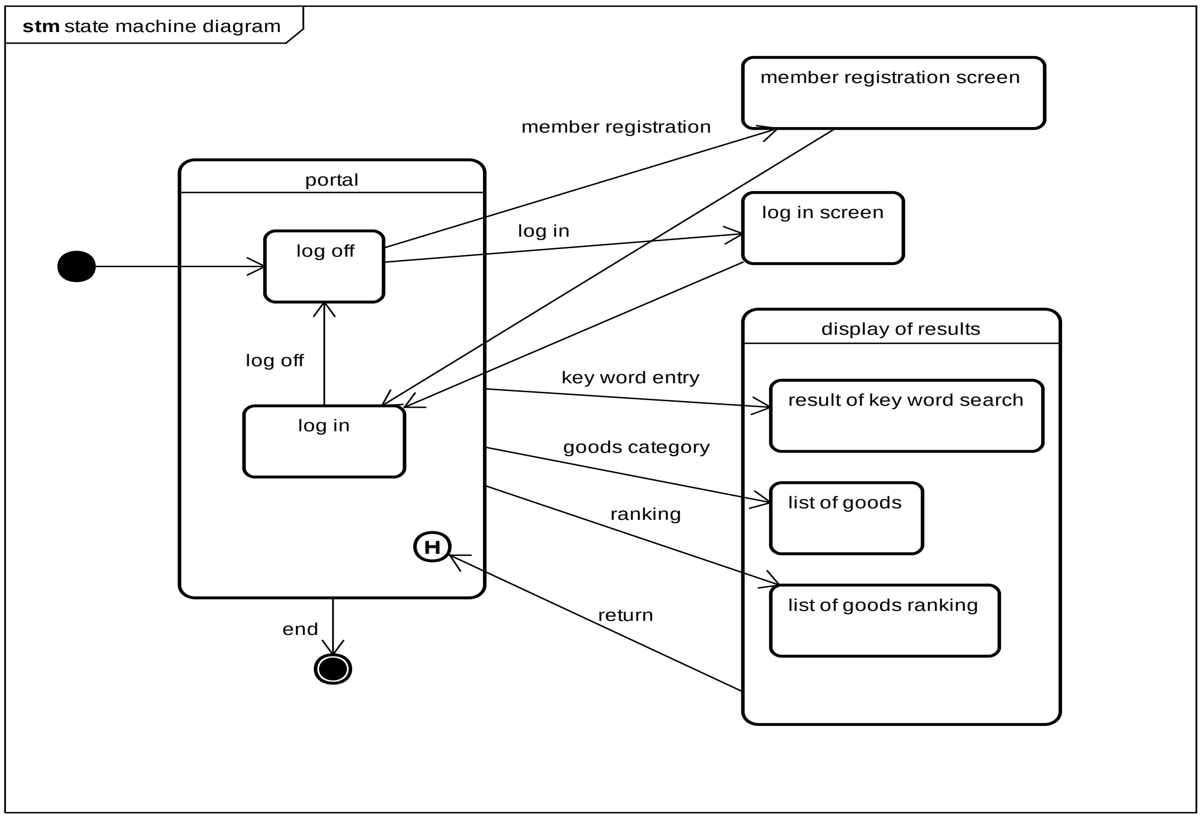
<System use case diagram>


member registration
search goods
customer
search by goods' categories
search by keywords
search by ranking
purchase goods
registration of goods
enter comments and evaluation of goods
change of goods' information
checking goods delivery status
administrator
member
display detail list of goods purchased
deletion of goods' information
change of member's information
closing the account

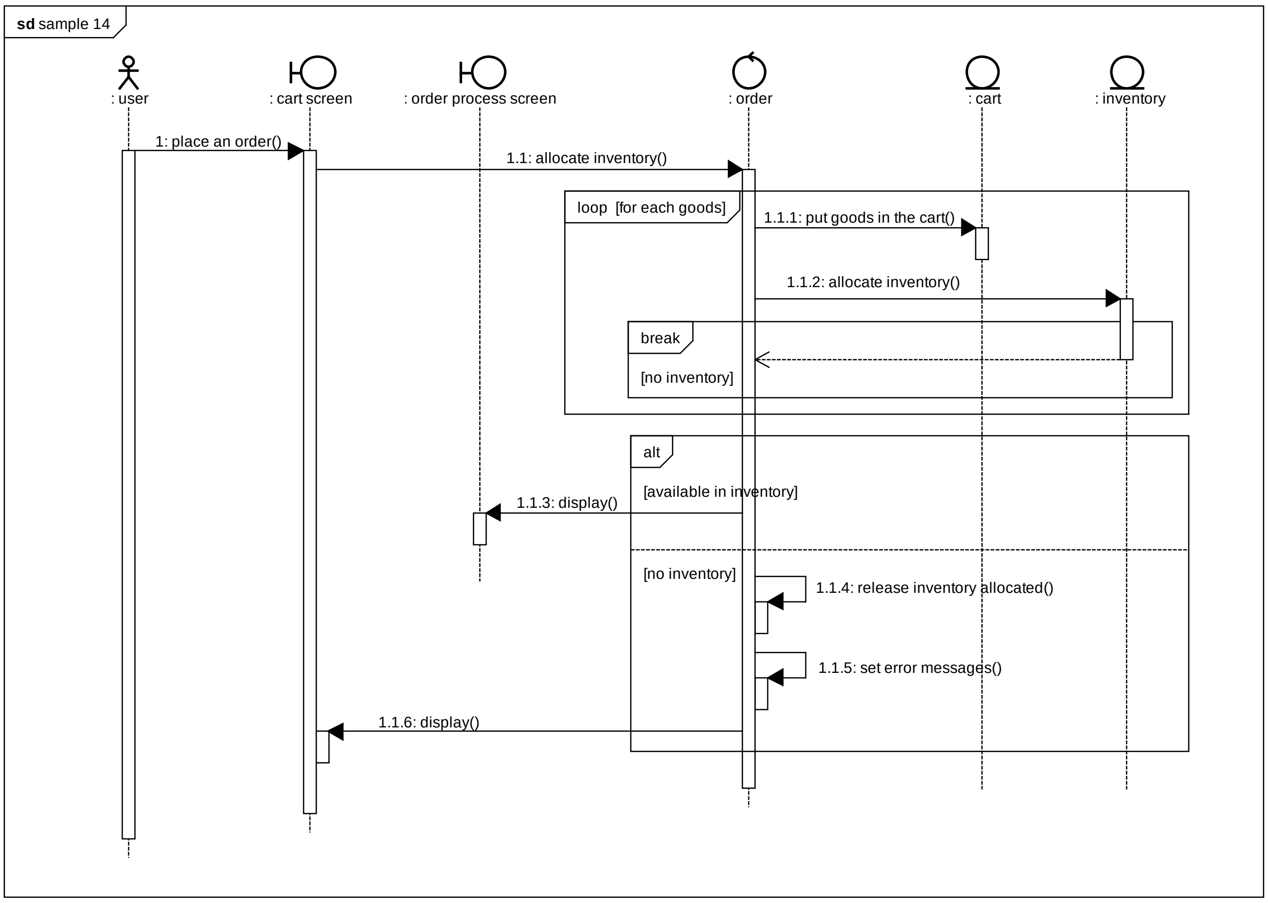
class diagram



state machine diagram (screen transition)　　※a part of screen for customers and members



sequence diagram



Appendix B. Sample outputs

(1) Each process output (partial)

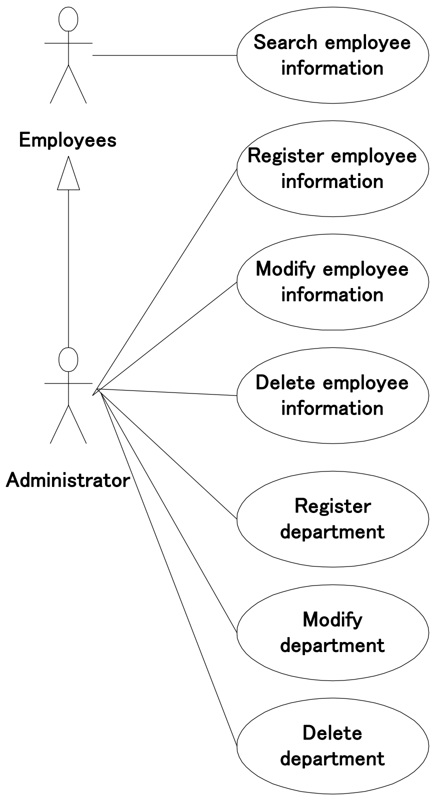
① Requirement definition

Company A is verifying the employee information search system development. If a user enters keywords such as department, age, or address of employees, it displays the employee information. Every employee can use this function.

The company employees are in technical/Technical/administration positions. The company has several departments and the employees belong to departments. Some employees belong to several departments at a time. The administrator registers, modifies, or deletes employee information in the system and registers, modifies, or deletes departments of employees also.

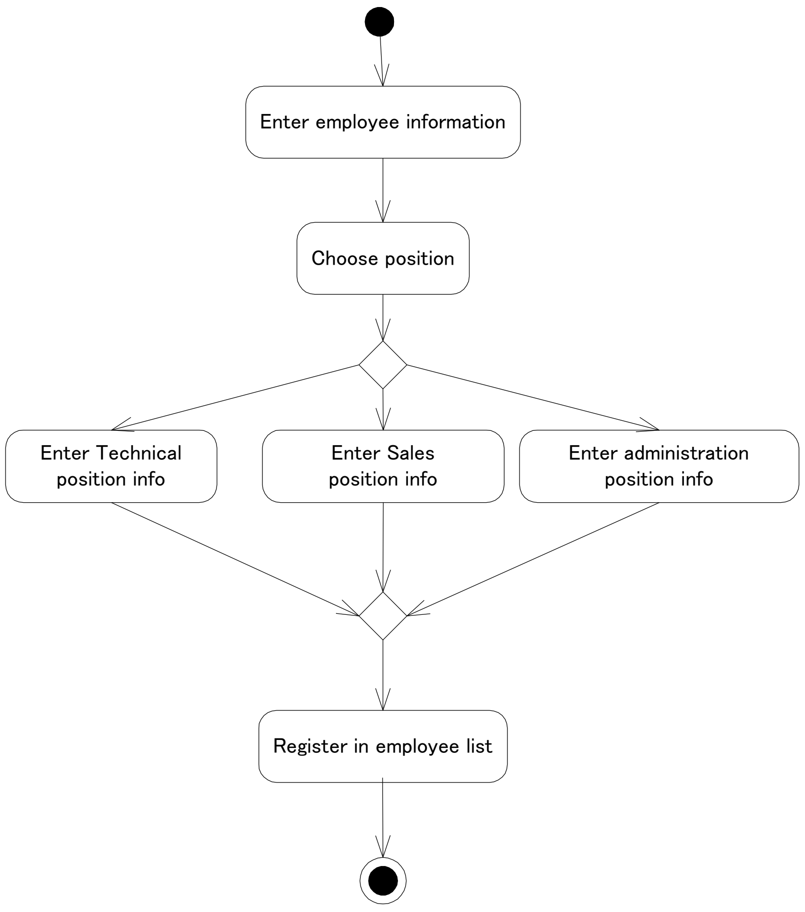
② Process： Requirement analysis (Japan ⇒ Offshore Partner)

●Use case diagram

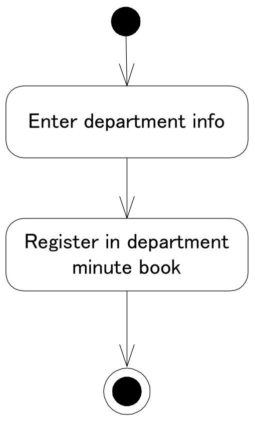

Employees
Administrator
Search employee information
Register employee information
Modify employee information
Delete employee information
Register department
Modify department
Delete department

● Activity diagram of use case (event flow)

Activity(event flow) of use case「Register employee information」


Enter employee information
Choose position
Enter Technical position info
Enter Sales position info
Enter administration position info
Register in employee list

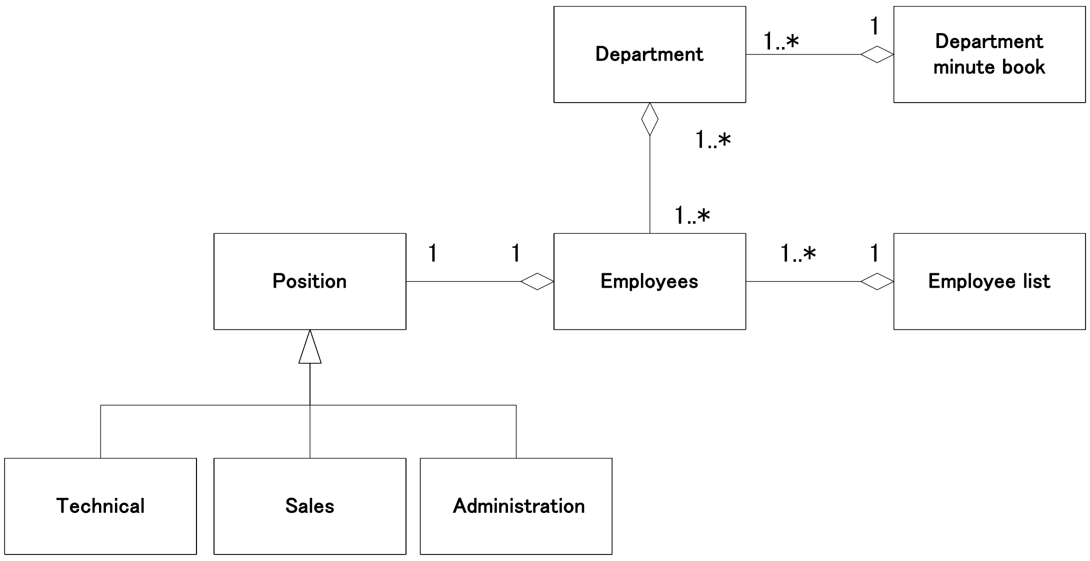
Activity diagram of use case「Register department」(event flow)



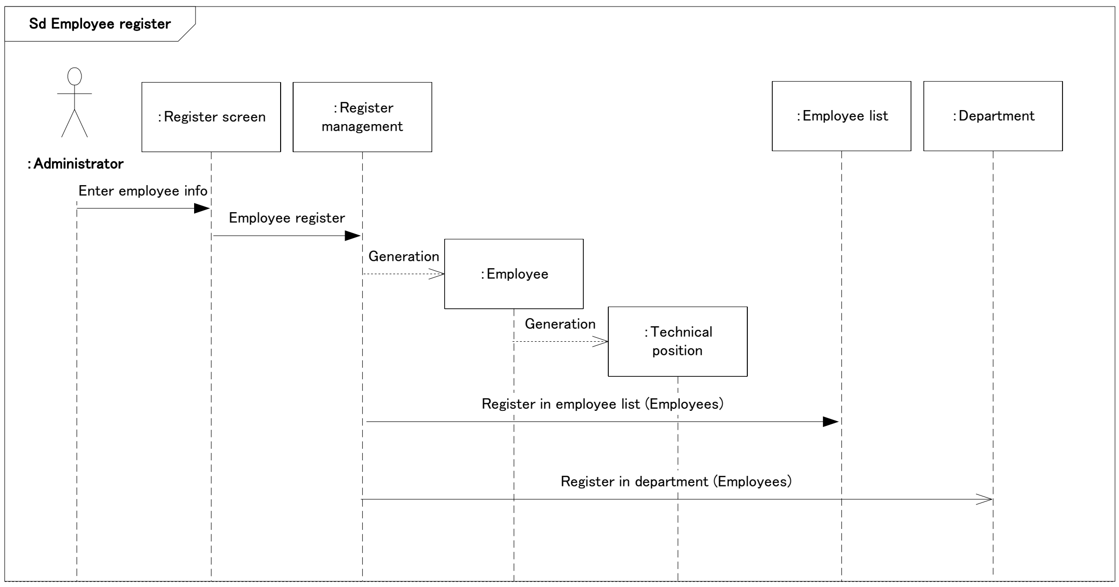
※ Describe activity diagram(event flow) on the other use cases.

③ Process: System analysis (Japan ⇒ Offshore Partner)

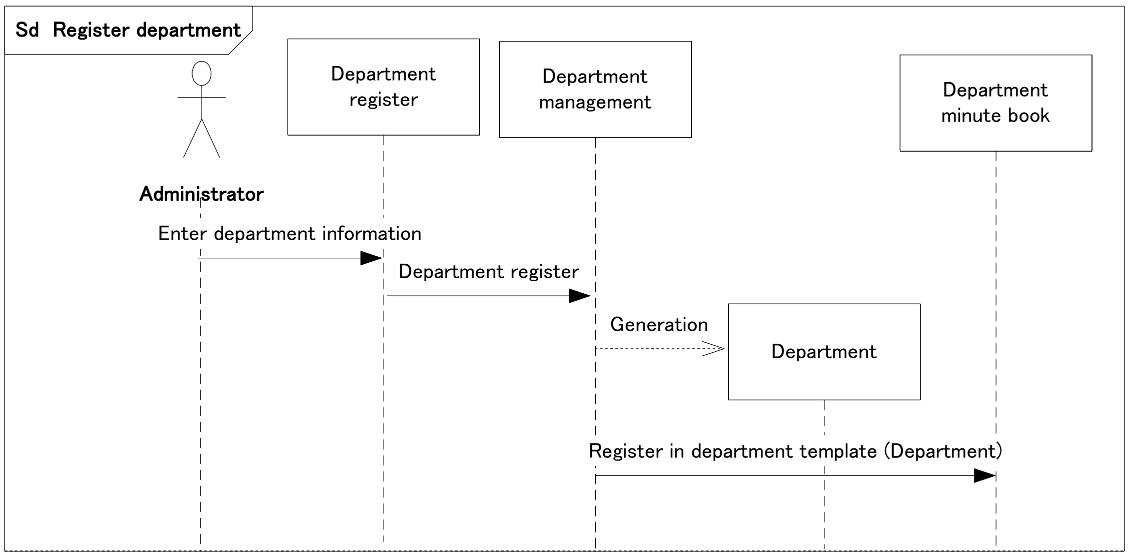
●Class diagram



●「Register employee information」 sequence diagram

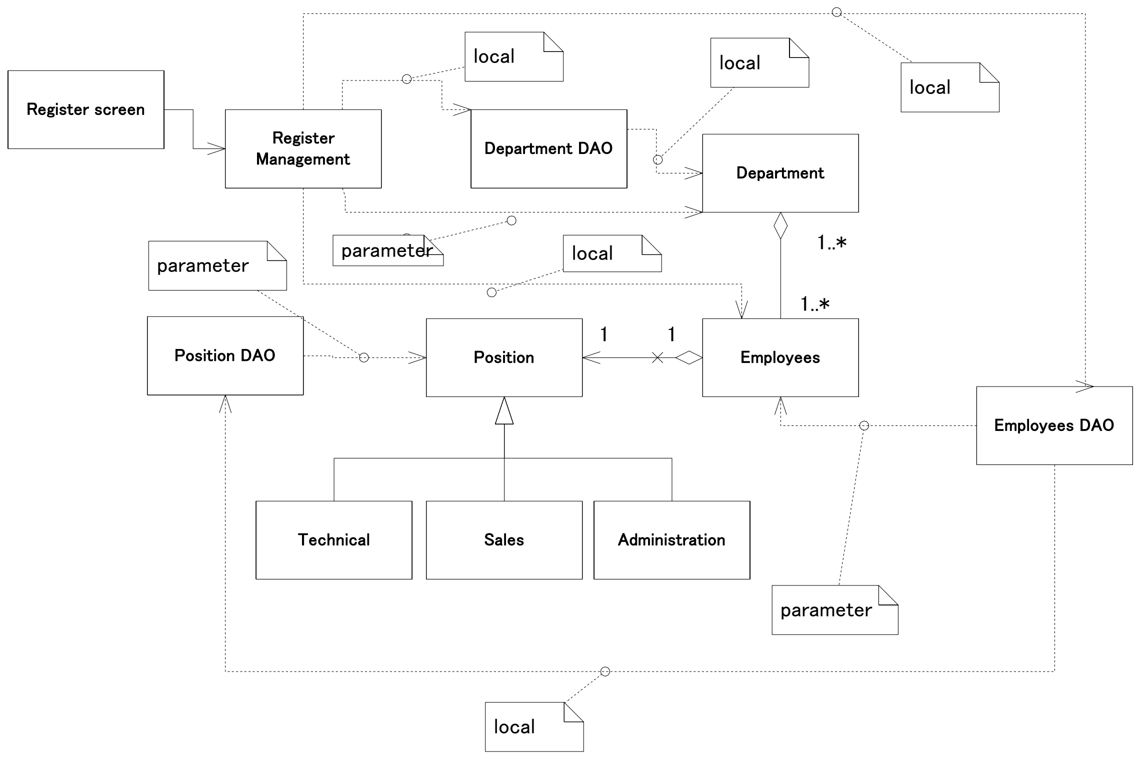


●「Register department」 sequence diagram

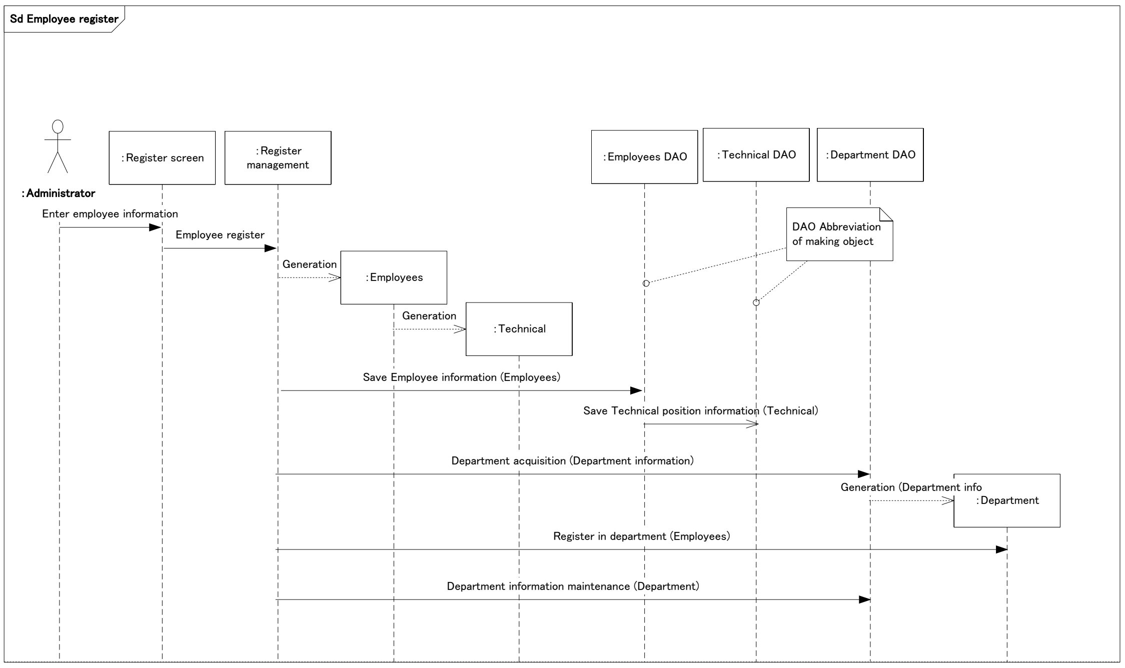


④ Process：Architecture design (Japan ⇒ Offshore Partner)

●Class diagram



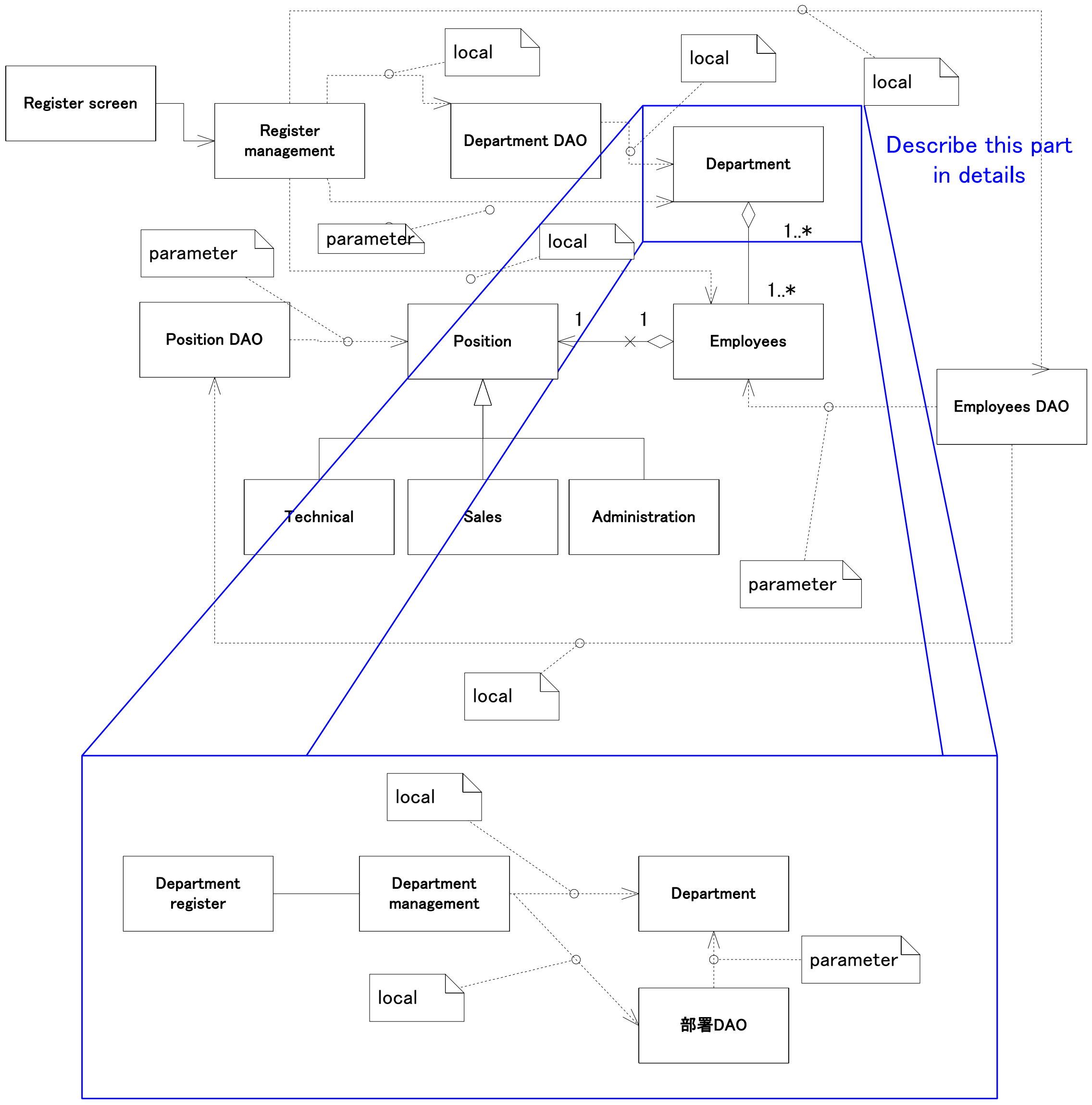
●「Register employee information」sequence diagram



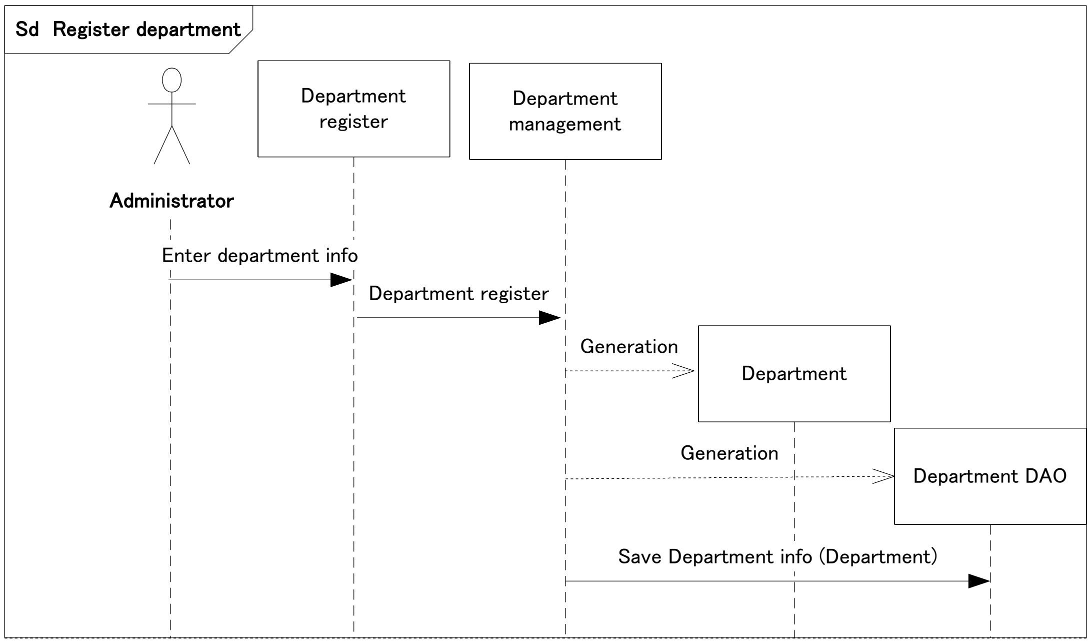
Sd Employee register
:Administrator
:Register screen
:Register management
:Employees DAO
:Technical DAO
:Department DAO
DAO Abbreviation of making object
Enter employee information
Employee register
Generation
:Employees
Generation
:Technical
Save Employee information (Employees)
Save Technical position information (Technical)
Department acquisition (Department information)
Generation (Department info
:Department
Register in department (Employees)
Department information maintenance (Department)

⑤Process：Detail design (Offshore Partner ⇒ Japan)

● Class diagram



● Sequence diagram



(2) Architecture description

Version 0.1

○Persistent use（use case view and part of logic view）

・Data Access Object (DAO) pattern use

Purpose：

Understand the difference between persistent methods and database by classifying business logic and persistent implementation.

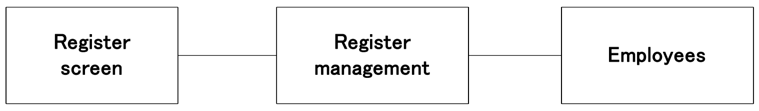
Role of DAO：

DAO is in charge of CRUD of data.

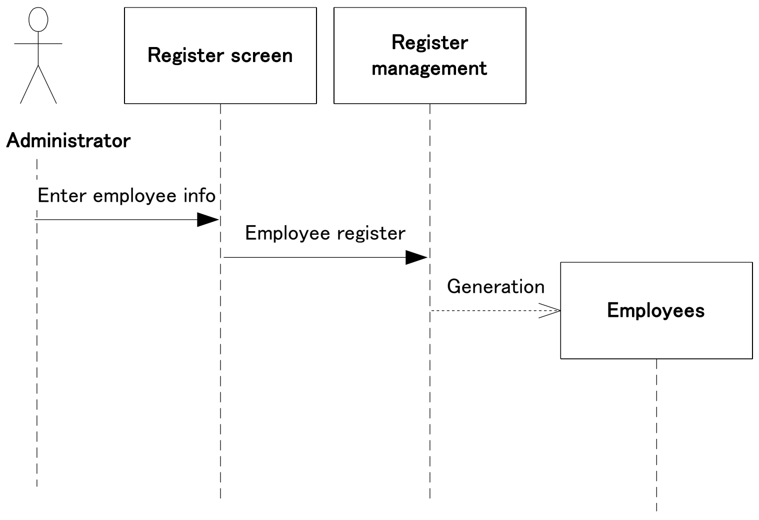
DAO connects to database.

・Example: DAO pattern is introduced for system analysis model by using a simple example (Employee register).

Class diagram of system analysis

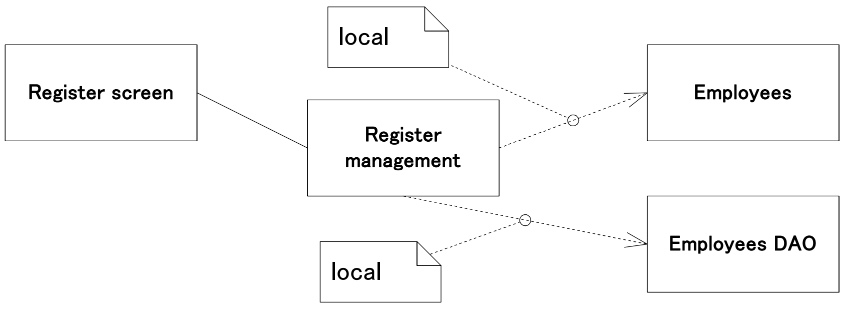


Sequence diagram of system analysis



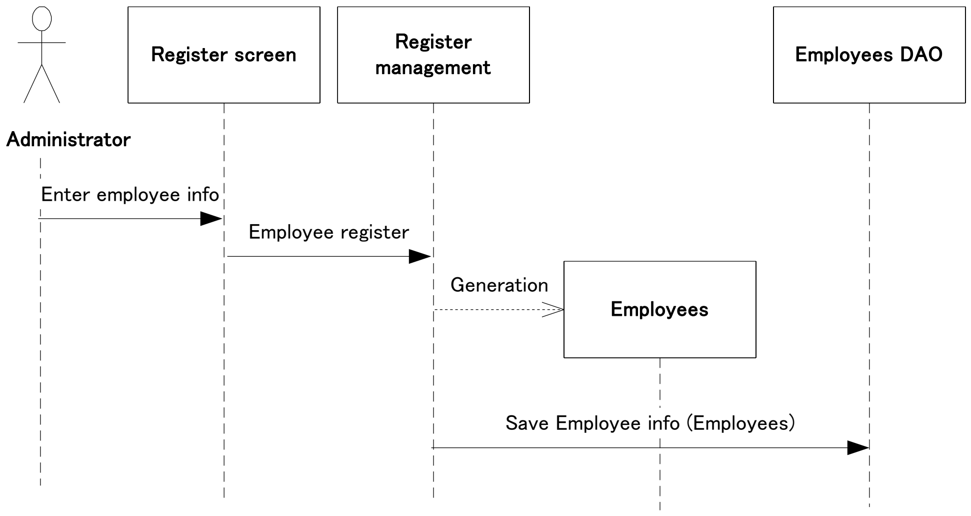
Design model with DAO pattern

Class diagram



※ “Local” means to generate dependency class object among the dependency class operations in dependency relationships.

Sequence diagram



「Employee DAO」class saves and loads「Employee class」 database.

.

.

.

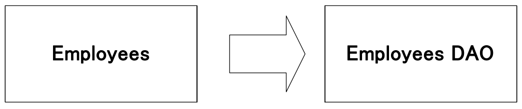
(3) Detail design guideline（partial）

Design class addition

・ Persistent Data Access Object (DAO) pattern

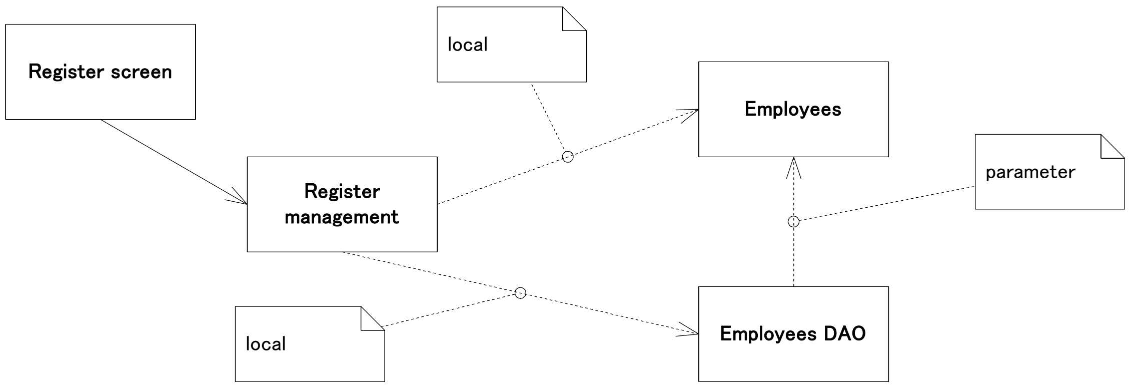
Each DAO class is drawn up for each entity class to be persisted.

DAO class name is entity class name + ”DAO”.



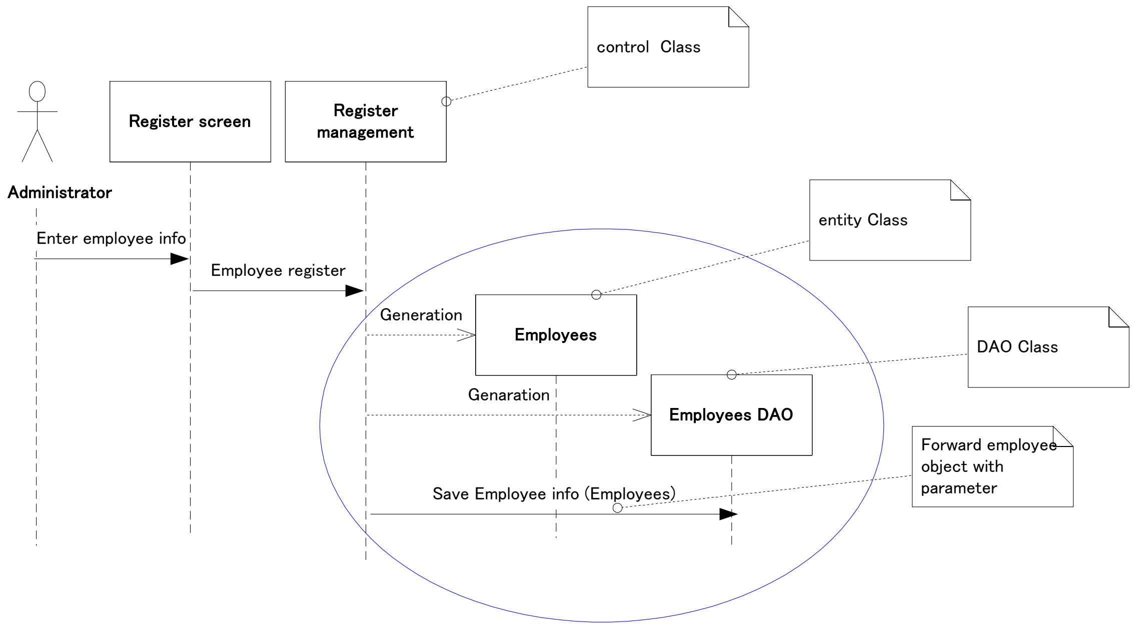
・ Class diagram

Display the dependency relationships and enter the comment「local」because the control class draws up new DAO class object locally and entity class. The DAO class receives the entity class as a parameter so that the dependency relationships need to be displayed with the comment ‘parameter’.



Sequence Diagram

Control class draws up and calls a DAO class object locally and calls it. Entity class is forwarded as a parameter in case of maintenance.



.

.

.

.

UML application guideline for offshore development

NPO UMTP

Offshore Software Development Working Group

Guideline preparation members

Name	Company
Masaru Adachi 足立 勝	Nikon Systems Inc.
Chunsheng Wang 王 春生	Oriental Standard Japan Co., Ltd.
Miho Oshita 大下 美穂	Bab-Hitachi Software K.K
Junichi Oka 岡 純一	Japan Information Co., Ltd.
Hiroshi Ogura 小倉 博	Open-Works Inc.
Naoki Kase 加瀬 直樹	Toshiba Corporation
Taro Kamioka 神岡 太郎	Graduate School of Commerce and Management, Hitotsubashi University
Akira Kamoshita 鴨下 明	Japan Systems Co. Ltd.
Yán Xu 許 炎	TONGHUA TECHNOLOGY (DALIAN) CO., LTD.
Ryuichi Konno 今野 隆一	Japan Systems Co.,Ltd
Hajime Saito 齋藤 肇	Hyron Software Co.,Ltd
Sutherland Maria サザーランド真理亜	Yokohama National University
Rui Shouda 正田 塁	Osaka Gas Information System Research Institute Co.,Ltd
Kazushi Takahashi 高橋 和司	Ad-Sol Nissin Corporation
Akitoshi Takemasa 竹政 昭利	Osaka Gas Information System Research Institute Co.,Ltd
Shigeru Tanaka 田中 繁	Open Works Inc.
Taku Nakajima 中嶋 拓	IT Engineering Ltd.
Toshimasa Nakahara 中原 俊政	Bab-Hitachi Software K.K
Yohei Haba 幅 洋平	Bab-Hitachi Software K.K
Tsuyoshi Hamakawa 浜川 剛	Technologic Arts Incorporated
Tsunahiro Hidaka日高 綱弘	Nippon Information Co.,Ltd
Masahiko Hiroi 広井 政彦	Nikon Systems Inc.
Hiroyuki Fujino 藤野 博之	NEC Nexsolutions, Ltd
Shigetoshi Hosaka 保坂 成利	Chiyoda Corporation
Hiroshi Makino 牧野 宏	Open Resource Corporation
Toshitaka Mori 森 稔貴	Japan Information Co., Ltd.
Shoji Yamaguchi 山口 昭二	Sakura Information Systems Co., Ltd.
Ryo Yoshida 吉田 亮	IBM Japan
Xiǎodōng Liang 梁 暁冬	Meiji Yasuda System Technology Limited
Ryouji Wakabayashi 若林 良二	NEC Nexsolutions, Ltd