國立交通大學

資訊科學與工程研究所

碩 士 論 文

建立無所不在學習之框架式互動學習模型

Building a Frame-based Interaction and Learning Model for U-Learning

研 究 生：施南極

指導教授：曾憲雄 博士

中 華 民 國 九 十 五 年 六 月

建立無所不在學習之框架式互動學習模型
Building a Frame-based Interaction and Learning Model for U-Learning

研 究 生：施南極　　Student：Nam-Kek Si

指導教授：曾憲雄 博士　　Advisor：Dr. Shian-Shyong Tseng

國 立 交 通 大 學
資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年六月

建 立 無 所 不 在 學 習 之 框 架 式 互 動 學 習 模 型

學生：施南極　　　　　　　　　　　　指導教授：曾憲雄 博士

國立交通大學資訊科學與工程研究所碩士班

摘　　要

隨著無線網路技術以及嵌入式系統的發展，線上學習已從 E-Learning 發展到 Mobile Learning 以及最近的 Ubiquitous Learning (U-Learning)。無所不在學習(U-Learning)的特性主要是提供學習的普遍性、不被中斷性，服務的互通性以及以嵌入式裝置為主的服務，除此之外並強調情境感知(Context-Awareness)的特性，依使用者身處地點、情境、時間等之不同來提供更適切的學習服務。而目前絕大部分建立無所不在學習應用程式的方法皆是以技術與功能導向的，導致難以導入老師與專家的知識，因此不能保證學習效果，對教學活動也缺乏了靈活性以及有不容易維護的問題。為了解決這些問題，本研究利用了知識工程的方法，提出一個知識整合架構的模型，稱為框架式互動學習模型(**Frame-based Interaction and Learning Model**, **FILM**)來描述 U-Learning 課程(U-Lesson)。本研究亦針對 FILM 模型提出一套參考設計流程(**FILM Design Process**)供工程師與老師有系統化地設計 U-Learning 課程。FILM 包括 **U-Lesson Behavior Petri Net Model (ULBPN)** 以及 **U-Lesson Material Frame Model (ULMF)** 兩個子模型。ULBPN 子模型為階層式高階派翠網，提供老師描述教學設計知識。ULMF 子模型為框架知識表示法，提供工程師描述底層的情境資訊推論與系統服務。最後我們提出了一個 U-Learning 的課程並以 FILM 模型來描述，以討論 FILM 模型之可行性、描述力以及延伸性。

關鍵字：框架、派翠網、模型、無所不在學習

Building a Frame-based
Interaction and Learning Model for U-Learning

Student：Nam-Kek Si Advisors：Dr. Shian-Shyong Tseng

Institute of Computer Science and Engineering
National Chiao Tung University

ABSTRACT

As Internet usage becomes more popular over the world, the learning technology including online learning, employee training courses, and e-book in the past ten years has been accepted globally. In recent years, Ubiquitous Computing technology is further applied in the domain of learning and is called Ubiquitous Learning (U-Learning). It is a computer supported learning paradigm for identifying learners' context information to provide integrated, interoperable, pervasive and seamless learning experiences. However, the technology and functionality driven development process of U-Learning application results in difficult to construct, maintain and extend the developed applications; moreover, learning efficacy can not be assured. In this thesis, we apply knowledge based approach and propose a **Frame-based Interaction and Learning Model** (**FILM**) consisting of two sub-model, **U-Lesson Behavior Petri Net Model** (**ULBPN**) and **U-Lesson Material Frame Model** (**ULMF**) for the development of U-Learning Lesson (U-Lesson). Based on FILM, a generic reference design process, **FILM Design Process** is also proposed to have a guideline of using FILM to design a U-Lesson. Teachers can use ULBPN to model the instruction design of a U-Lesson and engineers can use ULMF to model the materials used in a U-Lesson. Finally, a U-Lesson is modeled by FILM to evaluate its feasibility.

Keywords: Frame, Petri Net, Modeling, U-Learning

誌謝

首先必須感謝指導教授曾憲雄博士這兩年來的耐心指導，不管是在研究以及處事上都給予了我很多的指導與啟發，使我在碩士兩年內成長了不少，受益匪淺，感激不盡。同時也要感謝我的口試委員，黃國禎教授、孫春在教授與楊鎮華教授，他們給予了我相當多寶貴意見，讓本篇論文更有價值。

在此亦感謝博士班蘇俊銘學長與翁瑞鋒學長，在論文以及處事上給予了許多寶貴的意見。另外也要感謝王慶堯學長、曲衍旭學長與林順傑學長，他們給予了我很多學習與討論的機會。

同時也感謝實驗室的同窗夥伴們，林喚宇、林永彧、羅仁杰、鄧嘉文、葉展彰、張晉璿與蔡昇翰，不管是學業或是生活上，他們都陪伴著我渡過這段碩士生涯，讓這兩年更充實、更精彩。

最後，我要感謝我的父母、弟妹與女朋友潔雯對我的支持與鼓勵。特別感謝父母多年來的辛勞與培養，讓我能夠學業有成。也感謝潔雯在我面對挫折落魄時給予溫馨的支持，讓我不放棄自己，能夠重新振作完成本篇論文。

僅將本篇論文獻給每一位支持與幫助我的人。

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

As Internet usage becomes more popular over the world, the learning technology including online learning, employee training courses, and e-book in the past ten years has been accepted globally. With the development of wireless network and embedded system, the mobile computing technology brings a revolution to free users from the constraints of stationary desktop computing to services anytime and anywhere. In recent years, Ubiquitous Computing (U-Computing) technology is further applied in the domain of learning and is called Ubiquitous Learning (U-Learning) [22] or more specific, Context-aware U-Learning. It is a computer supported learning paradigm for identifying learners' context information such as surrounding environment and social situation to provide integrated, interoperable, pervasive, and seamless learning experiences. The objective of U-Learning is to move web based and mobile learning a step further from learning at anytime anywhere to be at the right time and right place with right learning resources and right learning peers.

U-Computing [24] is a new computing paradigm in which computing system are seamlessly integrated into the users' everyday life. Ubiquitous means that using computing technology should be as natural as using non-computing technology. Appliances should vanish in the background to let the user focus on his tasks rather than dealing with computing devices and technical issues. Many researchers [2][3] believe that context-awareness is an important aspect of the U-Computing vision. For computing systems to be context-aware [4][5][22] means that they are capable of providing users with relevant services and information based on the context information of the users and environment. Context-aware systems are able to adapt their operations to the current context without explicit user intervention and thus aim at increasing usability and effectiveness by taking environmental context into account.

U-Learning is the application of U-Computing research domain. It covers the researches of embedded system, communication technology and implementation of context-aware service. The characteristics of U-Learning are that, in the low level, the categories of context-aware service are education oriented, e.g. Content delivery, Assessment system, and in the high level, each design of a U-Learning application is a plan for a specific learning objective, which is called Instruction Design [15]. To specify our discussion, we used "U-Learning lesson (U-Lesson)" instead of "U-Learning application" in the rest discussion of this thesis. Instruction Design [10] is the systematic process of translating general principles of learning and instruction into plans for instructional materials and learning. ADDIE [10] shown in Table 1 is a general purpose commonly used instruction design process. It is an iterative process consisting of five phases: Analysis, Design, Development, Implementation and Evaluation. According to the evaluation result, we may improve the design by going back to previous phases repeatedly. Comparing to the design of generic U-Computing application, the design of U-Lesson is more dynamic, so that the maintainability and extensibility are important to this domain. Under our observation, a good development process of U-Lesson should satisfy the following criteria. It should be easy to construct, maintainable, extensible and guarantee of learning efficacy.

Table 1: Five steps of ADDIE instruction design process

Phase	Description
Analysis	Defining what is to be learned
Design	Specifying how it is to be learned
Development	Authoring and producing the materials
Implementation	Installing the project in the real world context
Evaluation	Determining the adequacy of the instruction

We review previous researches and observe the following conclusions. The development processes of U-Learning applications were technology and functionality driven. The process flows of U-Learning applications were implemented in programming level by engineer and did not take pedagogical theory into consideration. It results in the difficulty to construct, maintain, extend the developed applications; moreover, learning efficacy can not be assured. Obviously, it doesn't satisfy the criteria mentioned before. Actually, the maintainability and extensibility of U-Learning applications have not been discussed. Nowadays, related researches are confined to context management and a comprehensive method for the entire development process does not exist.

In this research, we apply knowledge based approach and the problem of development of U-Lesson is transformed into a modeling problem. Therefore, the development process is different from the traditional one. To author a U-Lesson, we first acquire the necessary knowledge from pedagogical expert and engineer using the proposed model as the knowledge representation. At run time, the lesson can be provided by executing the elicited knowledge with a specific engine. When the lesson needs to be maintained or extended, instead of rewriting, recompiling and redeploying the code, we only need to modify the model or invoke the knowledge acquisition process again, so that maintainability and extensibility is better than the previous approach.

To model a U-Lesson, we propose **Frame-based Interaction and Learning Model**

(**FILM**). It consists of three layers, **U-Lesson Process Layer** (**ULP**), **U-Lesson Activity Layer** (**ULA**) and **U-Lesson Material Layer** (**ULM**), and two sub models, **U-Lesson Behavior Petri Net Model** (**ULBPN**) which is designed for ULP and ULA, and **U-Lesson Material Frame Model** (**ULMF**) which is designed for ULM. Based on FILM, a generic reference design process, **FILM Design Process** (**FILM DP**) is also proposed to have a guideline of using FILM to design a U-Lesson. Teacher uses ULBPN to model the instruction design of a U-Lesson and engineer use ULMF to model the materials used in a U-Lesson. Finally, a U-Lesson is described and modeled by FILM. It shows that the FILM is a model that could take the knowledge of domain expert into consideration and a more flexible way to develop U-Lesson.

Chapter 2 Related Work

In this chapter, several U-Computing applications related to learning purpose are reviewed. Then, we also have an overview to the current state of U-Learning research. Since context-aware is the emphasized feature in U-Learning domain comparing to traditional learning and E-learning, we also discuss the related works of the problem of context acquiring and processing.

Ubiquitous (Pervasive) Computing

There are a few researches [1][9] in U-Computing domain related to learning purpose. In [1], a prototype of a mobile context-aware tour guide using user's location portfolio was presented to assist user in visiting scenic spots and writing tour diary. Dey [9] described a conference assistant, a prototype of mobile, context-aware application, which assists conference attendees using user's current location and the conference's agenda in order to provide conference guiding and valuable information gathering. In short, existing U-Computing application developments focused on how to use and manage context information and the usage of embedded device without considering the concept of learning activity, e.g., Role Playing, Learning situation. Thus, Lindquist [14] stated that it is not trivial to port the existing U-Computing application directly to U-Learning domain, new scenario and application for U-Learning should be proposed. We may conclude that the ideas of existing U-Computing application for learning purpose are valuable; furthermore, in order to apply those to learning domain, the knowledge of domain expert should be considered.

Ubiquitous Learning

In recent year, researches [6][19][26] in U-Learning centered on proposing innovative U-Learning scenarios and developed the corresponding implementation. Ogata [19] described a context-aware language learning support system for Japanese polite expression learning. The system provides the right polite-expression that is derived from hyponymy, social distance and situation through the identification of the target user and place. Cheng [6] proposed a personalized ubiquitous learning support environment in which an instructor edits learning instructional requirement set. By comparing them with learner behavior obtained, learner's situation can be grasped. Finally, suitable personalized support for the learner's situation can be identified by predefined rules.

We have three observations to the U-Learning researches. 1. The learning activity within the scenario is hard to maintain as the learning activity is embedded in process flow and event control during the design process, moreover, cooperation and interoperability between different applications could not be expected as design pattern and architecture of different applications are various. 2. The purposed scenarios are quite simple and only for special purpose. Actually, there exists a scalability issue and it is hard to overcome because a comprehensive development methodology does not exist. 3. It is hard for teacher to introduce their instructional knowledge since system approach and technology driven is used in the development of U-Learning application.

Context awareness

In order to support the development of ubiquitous series application, various architectures [8][18][20][25] focusing on the ease to manage and use context information have been proposed. Context Toolkit [8] which is a context information architecture consists of three types of components: context widgets, context interpreters and context servers. Widget abstracts the acquiring context details and provides context information to other components and application, the interpreter transforms information between different types of context, the server aggregates related context to obtain high level context information. The architecture also provides callback and resource discovery mechanism to support developing process. Similarly, contextual Information Service (CIS) [20] and HIVE [18], which provide a multi-layer software engineering process in order to separate context management from application development. Yang [25] proposed a context model with learner ontology and service ontology, and context acquisition mechanism for collecting contextual information at run time. We may conclude that their works are confined to the context management, but it is not much help in supporting the scenario development. Therefore, the modification, extension and combination with context-aware application based upon context architecture are still not flexible.

In summary, there are several issues in the domain of constructing U-Learning applications. The development processes of U-Learning applications were technology and functionality driven, so that learning efficacy can not be assured. The maintainability, scalability and extensibility of U-Learning applications have not been discussed. Related researches are confined to context management; a comprehensive method for the entire constructing process does not exist.

Chapter 3 Frame-based Interaction and Learning Model

3.1 U-Lesson Development Process

In the previous researches, the development processes of U-Learning applications were technology and functionality driven. The process flows of U-Learning applications were implemented in programming level by engineer and did not take pedagogical theory into consideration. It results in the difficulty to construct, maintain, extend the developed applications; moreover, learning efficacy can not be assured. To specify our discussion, from a pedagogical point of view, we use "U-Learning lesson (U-Lesson)" instead of "U-Learning application" in the rest discussion of this thesis. In order to assure the learning efficacy of the U-Lesson, apart from the technical point, such as context awareness and service flow control, we must combine the pedagogical knowledge with the engineer knowledge during the development process. The knowledge of teacher and pedagogical expert, e.g. teaching process, activity design, etc, should be stored in advance. To solve the construction, maintainability and extensibility issues, our idea is to apply knowledge based approach to the development of U-Lesson and propose a knowledge representation to model the knowledge involved in the construction of U-Lesson. Firstly, a knowledge acquisition process using the proposed model as the knowledge representation is designed to acquire the knowledge from domain experts and the elicited knowledge can then be executed by a specific engine. In other words, to design a U-Lesson, we first acquire the necessary knowledge from pedagogical expert and engineer using the proposed model as the knowledge representation. At run time, the lesson can be provided by executing the elicited knowledge with a specific engine. When the lesson needs to be maintained or extended, instead of rewriting, recompiling and redeploying the code, we only need to modify the model or invoke the knowledge acquisition process again. The whole idea is shown in Figure 1.

The development process of U-Lesson shown in Figure 1 is bottom-up approach. Conceptually, it consists of four phases, "Material Design", "Activity Design", "Process Design" and "Teaching & Evaluation". In Material Design Phase, Engineer abstracts and characterizes all available materials including services, device, profile and context, and material knowledge of the U-Learning environment is obtained. We observed that activities of instruction design built in one lesson may often be reused in another lessons. Therefore, we could modularize an instruction design into activities and reuse the activities in different instruction design. In Activity Design Phase, engineer and teacher cooperatively design individual activity by composing the materials constructed from the previous phase. The process flow knowledge of activities is acquired. In Process Design Phase, teacher designs the learning process targeting a learning objective by composing the activities constructed before and the process flow knowledge of U-Lesson is acquired. Finally, in Teaching & Evaluation Phase, the constructed U-Lesson is executed and efficacy of learning is evaluated. According to the efficacy, the Activity Design and Process Design phases may be retaken to make improvement to the U-Lesson.

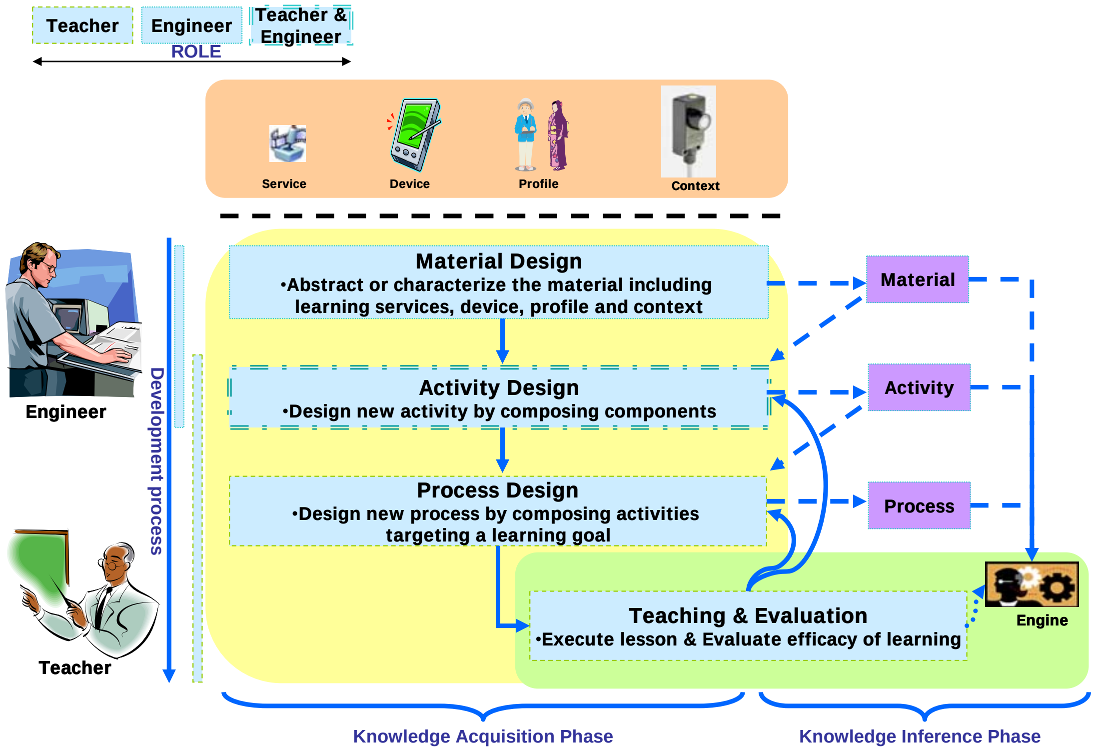

Figure 1: Idea of development and execution of U-Lesson

In the following of this chapter, we investigated the detail of elicited knowledge of each phase acquired during the proposed development process. After that, we discuss the idea of our proposed model which includes choosing suitable knowledge representations to model the knowledge and how the knowledge representations could be integrated into a comprehensive model.

3.2 U-Lesson

Before we discuss U-Lesson, we first have a review to traditional lesson (or Course) in E-learning domain. A lesson comprises two elements: 1) Instruction Design [10][15]--

Activities learners are required to perform. 2) Resource -- Content or resources learners interact with. To design the instruction design, the first step is to identify learning objectives and then design learning activities to help students meet the objectives. These activities will make use of the available resource and education tools (learning services).

Traditional instruction design lacks the consideration of U-Learning environment's feature. In tradition, profile is the only consideration for activity's process flow. But in the case of U-Learning, many kinds of context information are available and they provide the necessary information for learning services to decide when it should be triggered and what the right response is, so that we can provide suitable learning services at right's time, right's place, right's situation and right's location. To utilize the context information, instruction design should take it into consideration, so that context-awareness is the important aspect we should introduce to instruction design. The problem of context-awareness includes how to represent contextual information and how services can use them. Contexts often being discussed are time, location, intention and user profile. Apart from the context raw data, high level context interpretation (Context reasoning) is essential to every context-aware service. The function of a context-aware service depends on the support of context reasoning. Besides the instruction design, resources used in U-Learning are not limited to content and assessment, learning is provided by the cooperation of many kinds of services implemented in various embedded system, e.g., sign, alert, etc. could be used to support the learning.

Following our development process of U-Lesson, three kinds of knowledge can be obtained and the architecture is shown in Figure 2.

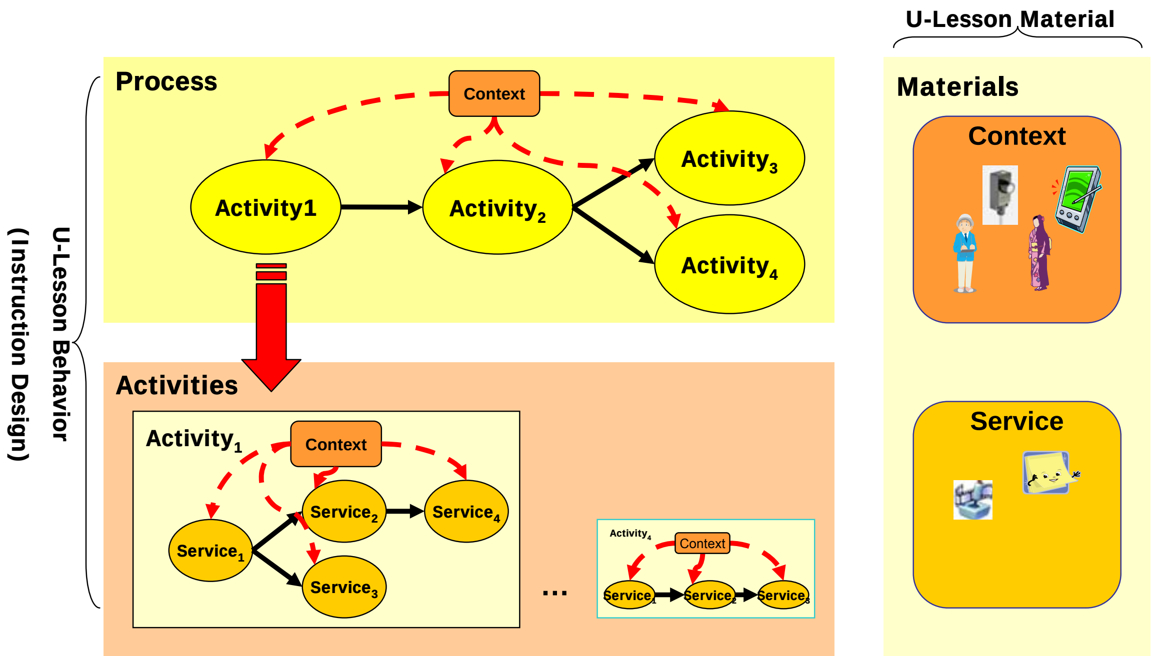

Figure 2: Architecture of U-Lesson knowledge

Each kind of knowledge is corresponding to a phase in the development process, Material is acquired from the Material Design Phase, Activity is acquired from the Activity Design Phase and Process is acquired from the Process Design Phase. We discuss detail of each kind of knowledge below.

Material

The knowledge of Material can be further divided into Context and Service. Context means all kinds of possible information that can be utilized. According to the data characteristics, context can be distinguished into two types: Static (User Profile, Portfolio, Preference) and Dynamic (i.e., Location, Time, Situation). Besides the raw context, high level interpreted context is also contained. Service means applications which can assist learners in the U-Computing environment. Categories of services are organized as service ontology shown in Figure 3.

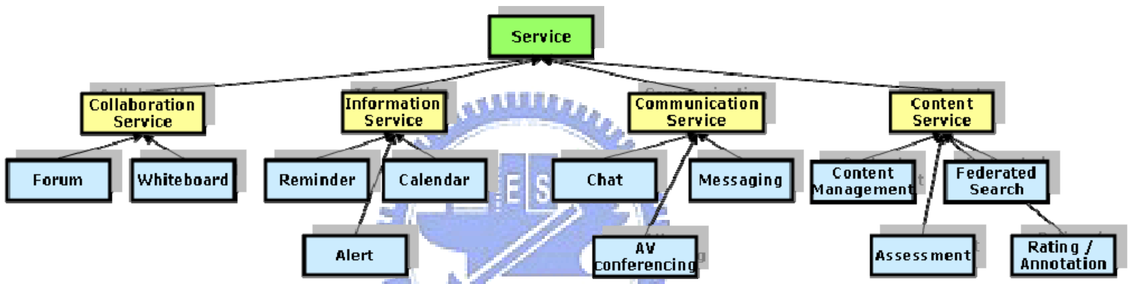

Figure 3: Service Ontology

Activity

The knowledge of Activity is part of instruction design. Each activity is a module of adapted sequence defined to use service available in Material to composite an activity. Each service may have its prerequisite trigger conditions to specify when a service should perform to learners. The conditions may be context or information given from other services.

Content or resources learners interact with in an activity would be specified as parameter in process or activity layer.

Process

The knowledge of Process is similar to Activity. Both of them are instruction design. The difference is that there is only one adapted sequence defined in Process which uses a set of predefined activities to satisfy a designated learning objective. Each activity may have its prerequisite trigger conditions to specify when learners should start the activity. The conditions may be context or information given from other activities.

As both learning process and learning activity focus on the behavior of activities and services respectively, they define the behavior of a U-Lesson, so that we define them as U-Lesson behavior.

3.3 Concept of FILM

Petri Net for U-Lesson Behavior

Petri net [7][12] is a graphical and mathematical modeling tool, which can be used to express process flow, parallelism, concurrent, causality, synchronization and resource sharing. It has a well-defined semantics allowing formal analysis, and consists of places, transitions and arcs where input arcs connect places with transitions and output arcs start at a transition and end at a place. High Level Petri Net (HLPN) is an extension of Petri Nets, which is able to model and validate more complex systems. Conceptually, the migration step from low-level Petri nets to HLPN net is similar to that from assembly languages to modern programming languages with an elaborated type concept. In low-level nets there is only one kind of token and this means that the state of a place is described by an integer. In HLPN, each token can carry complex information or data and may describe the entire state of a process or a database.

In order to develop complex systems, an abstraction concept of structuring and grouping is proposed to work with a selected part of the model without being distracted by the low-level details of the remaining parts. HLPN is suitable for modeling the behavior of U-Lesson, to model how learning process controls the activities and how individual activity controls the behavior (context-awareness feature) of context-aware services. Hierarchical High Level Petri Net (HHLPN) [12] [11] provides the petri net modeler with such abstraction mechanisms. The basic idea behind HHLPN is to allow the modeler to construct a large model by combining a number of small petri net into a larger net. It is similar to the situation in which a programmer constructs a large program from a set of modules and subroutines. In addition, since each HHLPN can be transformed into a non-hierarchical petri net, the previous formal analysis methods applied to HLPN can be also applied to HHLPN. To take advantage of the modularization and reusability of the hierarchical concept, we could model learning process and activities separately by HLPN and model the hierarchical relationship between them by HHLPN. In the case that we are designing learning process, we can give a simple description of the modeled activity without having to consider internal details about how it is carried out, e.g., teacher could model the learning process without considering how the detail is in the individual learning activities. And in the other case that we are designing learning activity, we can specify the behavior in more detail, e.g., teacher and engineer cooperatively model the detail of learning activities. Furthermore, we want to be able to integrate the detailed specification of learning activity with the more high level description of learning process and this integration must be done in such a way that it becomes meaningful to speak about the behavior of the combined system. HHLPN can provide the necessary semantic for the hierarchical model and is responsible for the integration.

Frame for U-Lesson Material

Frame [11][17][21][23] provides a convenient structure with object oriented concept for representing objects that are typical to a given situation such as stereotypes. It is structural knowledge about a limited physical or conceptual entity of the world and the knowledge is organized in large well structured chunks. Basically, frame is a group of slots that define a stereotypical object. Slot is used to store information and a default value could be given to describe the object's attribute; moreover, slot may hold procedural attachments which may attach rule or procedure. Therefore, frame is tightly combining declarative and procedural aspects for proper balance between expressive power and efficiency. There are four types of procedural attachments. The "if needed" rule is triggered while retrieving a slot value but a default value is not available. The "if added" rule is triggered while storing a value in a slot. The "if changed" rule is triggered while editing a slot value. The "if removed" rule is triggered while removing a slot value. The inheritance between entities is represented as a hierarchical relationship between frames which as always structured as ontology.

New Object-Oriented Rule-base Model (**NORM**) [13] is an object oriented rule base model and is based on the concept of learning and thinking behaviors of human. It provides high maintainability and reusability through the object-oriented concept. A **rule base** contains **knowledge classes** (**KC**s). Each KC which represents a kind of concept of the domain knowledge consists of **rules**, **relation** with other KCs and **fact** declarations as shown in Figure 4. A fact which consists of name, value and possibility represents all kinds of appearance in real world and is used when inferring. During inference process, the rules use facts to obtain reasonable conclusion. A rule consisting of condition part, action part, certainty factor, threshold and weight is the basic knowledge element. There are four relations: **Trigger**, **Acquire**, **Reference** and **Extension-of**.

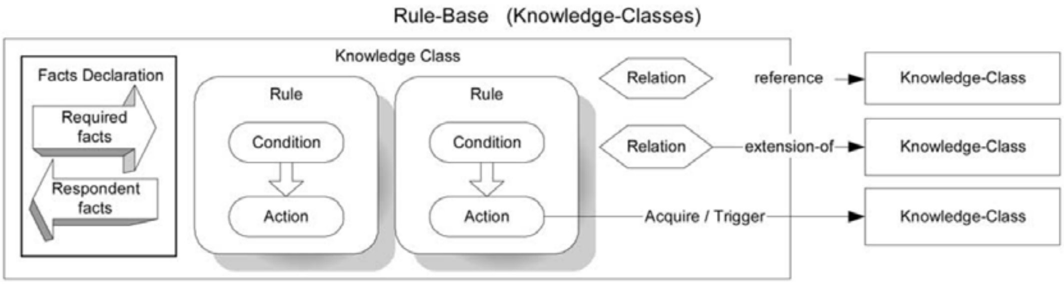

Figure 4: The knowledge class in rule base of NORM

The characteristics of the material of U-Lesson can be defined as follows. 1) Most objects are stereotype; context and service can be represented by a set of attributes (Declarative knowledge). 2) The behavior of the objects (Procedural knowledge) is event driven; the behavior could be implemented with procedure call or rule class inferring. 3) The inheritance relationship of objects can be structured by ontology and take advantage of it. As mentioned above, frame provides the necessary expressive power to model the knowledge of this layer; furthermore, we can construct logical entity and context services using frame. Web Ontology Language (OWL) [16], a similar knowledge representation, also could model stereotype, use ontology to construct the relationship of entities, and attach rules which reason about the ontology. But it doesn't have mechanism to model procedural knowledge. In addition to the frame, NORM is used as the rule model for the attached rule of the frame to ensure maintainability and reusability.

Frame based Interaction and Learning Model

The goal of **Frame based Interaction and Learning Model** (FILM) is to provide a systematic and integrated method to model a U-Lesson. It is a systematic method that we decompose the problem of U-Lesson modeling into three layers and two sub models. The three layers are **U-Lesson Process Layer (ULP)**, **U-Lesson Activity Layer (ULA)** and **U-Lesson Material Layer (ULM)**. The two sub models are **U-Lesson Behavior Petri Net Model (ULBPN)** designed for ULP and ULA, and **U-Lesson Material Frame Model (ULMF)** designed for ULM. We use High Level Petri Net to model the ULP and ULA and use Frame as the knowledge representation of ULM. It is an integrated method that each layer of the model can be one to one related. In the upper layer, the HLPNs of ULP and ULA of U-Lesson behavior are related by HHLPN. The HLPN of ULP is the prime page of HHLPN, and the HLPN of ULA is the subpage of HHLPN; moreover, the transition in HLPN of ULP is substitution transition which substitutes the HLPN of ULA. In the lower layer, the HLPN of

ULA and frame of ULM are related by defining a special transition. The transition substitutes a frame and also defines the operation effect to the specified frame. To bridge with the real U-Learning environment, to control learning service and sensor, a middleware can be defined to communicate with ULMF.

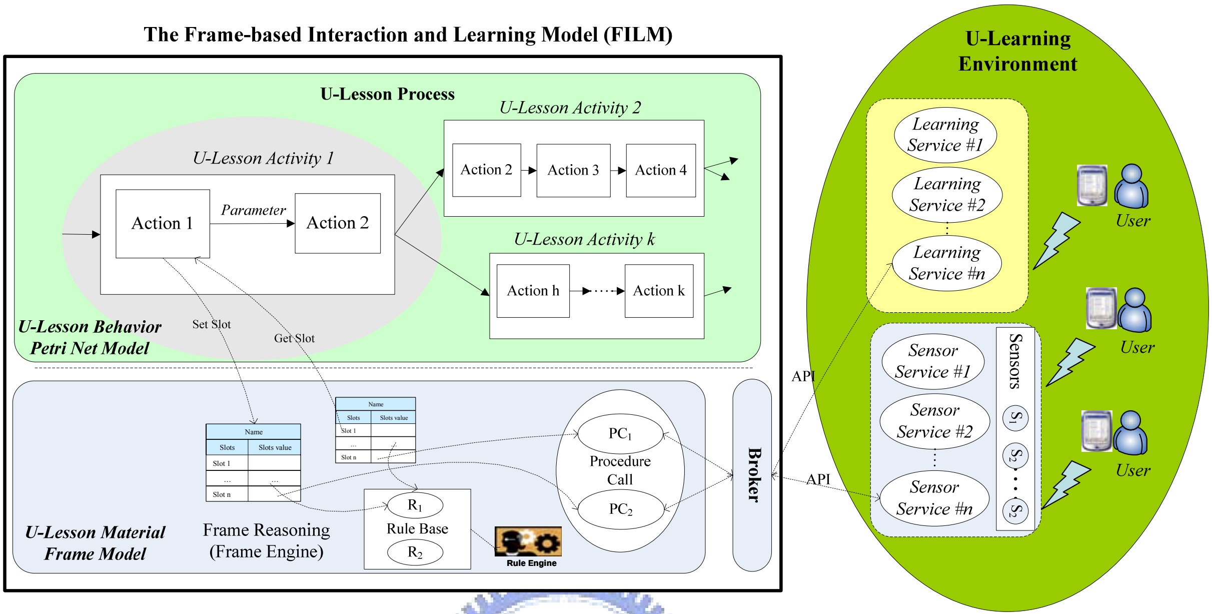

Figure 5: The concept of FILM

Chapter 4 Definition of the Frame-based Interaction and Learning Model

4.1 FILM Definition

Definition 1. The **Frame-based Interaction and Learning Model** (**FILM**) is a 2-tuple: **FILM** = (**ULMF, ULBPN**), where

1. **U-Lesson Material Frame Model** (**ULMF**)

   The core of the FILM architecture is the frame layer. Basically, frame provides a knowledge representation of context and services. Each object, such as person, sensor, device and environment are represented as an individual frame. Their attributes and procedural behavior (Procedure call and reasoning) are represented as frame's slot and slot's procedural attachment respectively. Moreover, context services which provide a high level usage and interpretation of context information are also introduced in frame layer. Finally, the relationship of the objects could be managed by a domain ontology and ontology reasoning can be realized by procedure call or rule class.

2. **U-Lesson Behavior Petri Net Model (ULBPN)**

   The **Hierarchical High Level Petri Net for U-Lesson Behavior (ULBHPN)** is applied in this model for the behavior of U-Lesson, including the learning process and learning activity. In ULBHPN, we use **U-Lesson Process High Level Petri Net (ULPPN)** to model learning process and **U-Lesson Activities High Level Petri Net (ULAPN)** to model several learning activities. The connection between learning process and learning activities is realized by defining substitution transition in the ULPPN to substitute corresponding ULAPN. The duty of ULAPN is to take care of the flow control of frame.

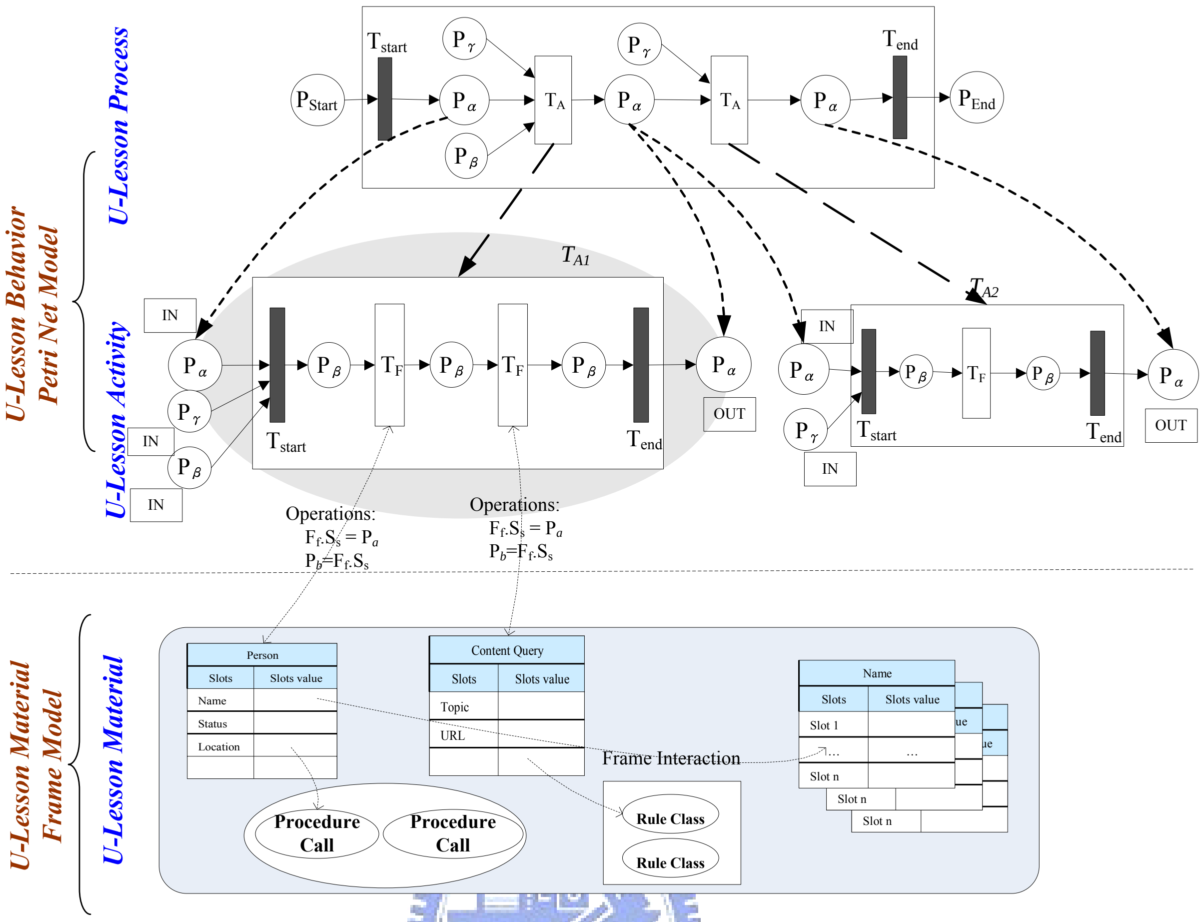

Figure 6: The architecture of FILM

4.2 U-Lesson Material Frame Model

Definition 2. The **U-Lesson Material Frame** (**ULMF**) is a 3-tuple:

ULMF = (**P**, **RC**, **Frame**), where

1. **P** is a finite set of **k** procedure calls, where **P = {(PN$_i$, PB$_i$, PI$_i$, PO$_i$) |** $\forall i, 1 \le i \le k$ }. **PN$_i$, PB$_i$, PI$_i$** and **PO$_i$** are the name, body, input parameter and output parameter of the **i**-th procedure call, respectively.
2. **RC** is a finite set of **l** rule classes, where **RC = {(RU$_i$, FC$_i$)}** $\forall i, 1 \le i \le l$ }. ULMF uses NORM [13] as knowledge modeled rule base. **RU$_i$** is a set of rule in the **i**-th rule class and **FC$_i$** is a set of fact in the **i**-th rule class.
3. **Frame = (FN, I, S)** with **m** slot, **n** procedure calls, **k** rule classes where
   i. **FN:** is the name of a frame. It can be a service name, context name or context service name.
   ii. **I = (R, FN)**, where **R** is relation with other frame specified by frame name **FN**. There are two types of relation. The "*a kind of*" denotes the inheritance relation.

The "*is a*" denotes instantiation relation.

iii. **S** is a finite set of slots of a frame, where **S** = {(**SN_i**, **SV_i**, **ST_i**, **DV_i**, **SR_i**) $|\forall i, 1 \le i \le m\}$

- **SN_i** is the name of the **i**-th slot.
- **SV_i** is the value of the **i**-th slot.
- **ST_i** is the data type of the **i**th slot. Data type could be ordinal type, such as: int, string, it could also be pointer type which point to another frame.
- **DV_i** is the default value of the **i**-th slot. It is assigned to slot value when frame instantiation if available.
- **SR_i** = **{(RT_1, PA_1, IF_1), (RT_2, PA_2, IF_2), (RT_3, PA_3, IF_3), (RT_4, PA_4, IF_4)}** is a set of guard rules which trigger the procedural attachments based on different events. There are four types of guarding rules: **1**: "if needed", **2**: "if added", **3**: "if changed" and **4**: "if removed". The procedural attachments **PA** attached to a guarding rule could be a procedure call **P** or a rule class **RC**.

  If the attachments are procedure call with **x** parameters, **IF={<BD_1, PI_1>, …, <BD_x, PI_x>}** where **BD** could be the form "*FrameInstance.Slot*" or a variable holder. **PI_i** is the **i**-th parameter of the procedure call.

  If the attachments are rule class with **y** facts, **IF={<BD_1, FC_1>, …, < BD_y, FC_y>}** where **BD** could be the form "*FrameInstance.Slot*" or a variable holder. **BD_i** could be null to represent the **i**-th fact is not needed to be set before inferring. **BD_i** is the **i**-th fact of the rule class.

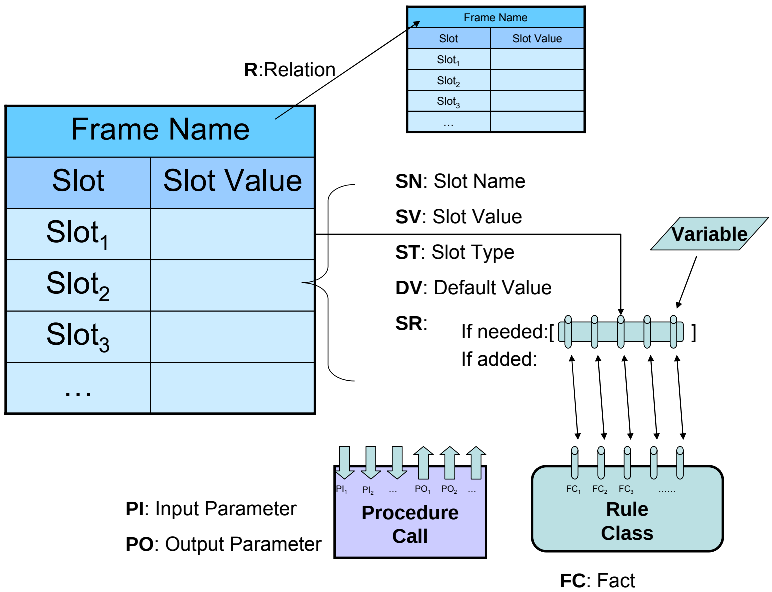

Figure 7: The elements of ULMF

A graphic representation of ULMF is shown in Figure 7. Besides, the elements of traditional frame knowledge representation, the concept of Interface is purposed to ensure the modularization and reusability of rule class and procedure call.

Example

An example using ULMF to model the categories of messaging service is shown in Figure 8. Services in the same category often share some attributes and functionality, by utilizing the frame inheritance mechanism; we can easily construct the frame for a new service by constructing the relationship of the service category. We can organize services into ontology like the service ontology shown in Figure 3. Example of context interpretation is shown in next chapter.

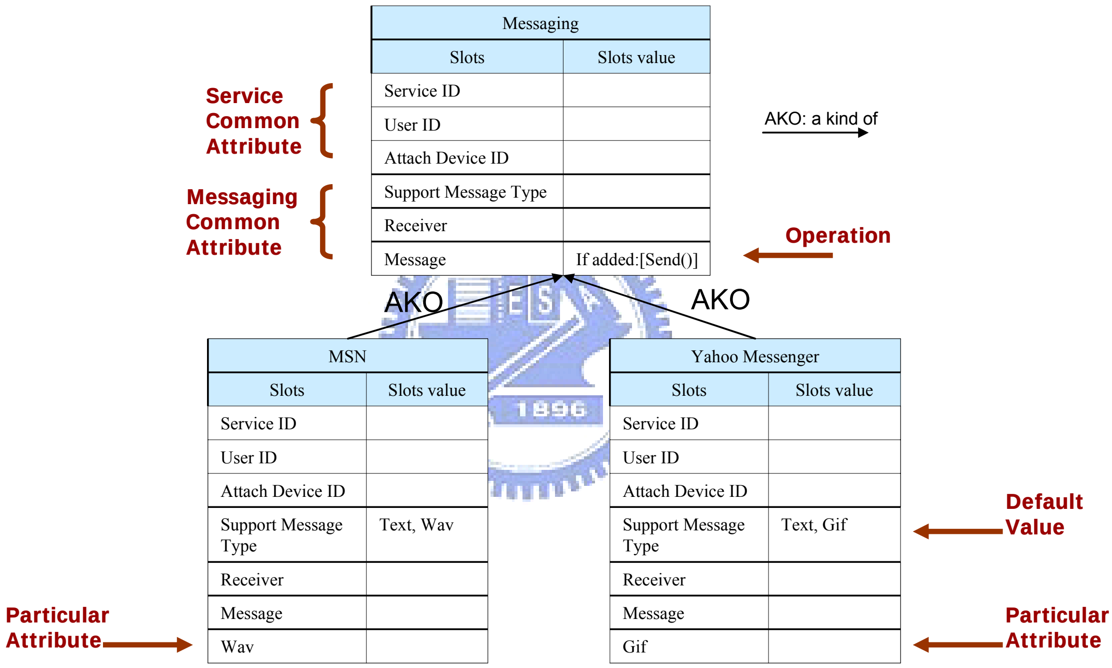

Figure 8: Service modeling with ULMF example

4.3 U-Lesson Behavior Petri Net Model

Hierarchical High Level Petri Net [11][7][12]

Hierarchical High Level Petri Net (HHLPN) contains a set of non-hierarchical Petri Nets which call ***pages***. In order to specify the pages we shall take a very general and simple strategy, a set of start pages, called ***prime pages***, should be specified by user. The most important component of HHLPN is ***substitution transition*** because it allows the user to relate a transition (and its surrounding arcs) to a more complex petri net – which usually gives a more precise and detailed description of the activity represented by the substitution transition. A substitution transition is said to be ***supernode*** of the corresponding ***subpage*** while the page of the substitution transition is a ***superpage*** of the corresponding subpage. ***Page assignment*** function is to map a substitution transition to a subpage. How the subpage should be glued together with the surroundings of the supernode is provided by ***port assignment*** which describes the interface between the superpage and the subpage. The port assignment is to map a ***socket node*** on the superpage (i.e., one of the places that surround the substitution transition) with a ***port node*** on the subpage (i.e., one of the places which is marked with a port type). There are our ***port types*** of the port node, ***In***, ***Out***, ***I/O***, and ***General***. An In type indicates that the port node must be related to a socket node which is an input node of the substitution transition. An Out type indicates that the port must be related to a socket which is an output node, while an I/O type indicates that the socket must be both an input and output node. A General type indicates that the port can be related to all kinds of sockets.

Example

Figure 9 is an example of HHLPN with one prime page (Page1) and two subpages (Page2, Page3), totally three pages. In Page1, there are two substitution transitions, T_{11} and T_{12}, which substitute subpage Page2 and subpage Page3 respectively. To simplify our discussion, we discard page Page3 in the following discussion. Transition T_{11} is supernode of the page Page2 and page Page1 is the superpage of page Page2. Page assignment is defined to map transition T_{11} to page page2. Socket node on page Page1 are places P_{11}, P_{12}, P_{13}, and P_{14}. Port node on page Page2 are places P_{21}, P_{22} and P_{24} and Port types of P_{21}, P_{22} and P_{24} are In, In, Out respectively. Port assignment is defined to map P_{11} to P_{23}, P_{12} to P_{21} and $P_{13}$to P_{24}.

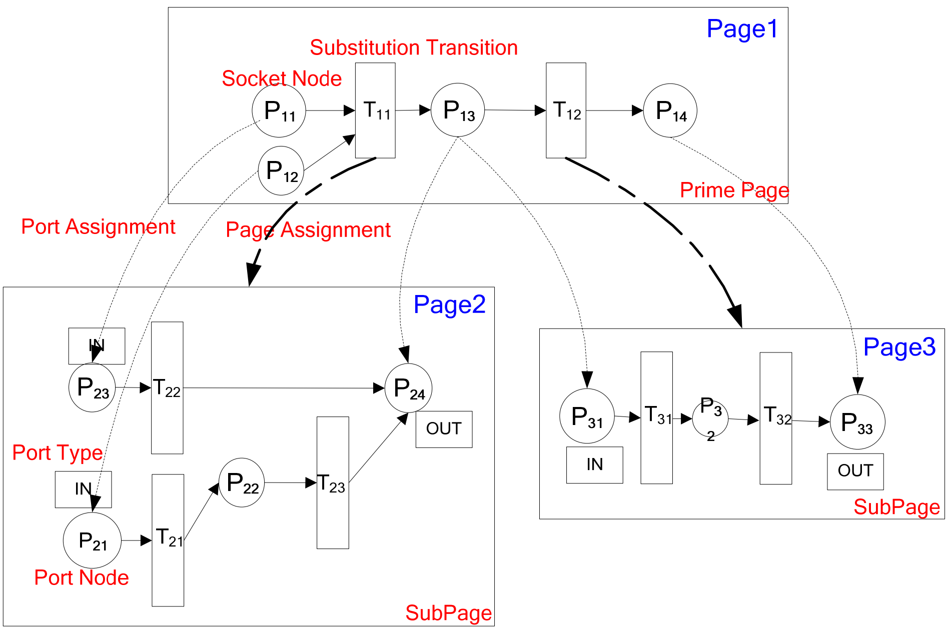

Figure 9: Hierarchical High Level Petri Net example

ULBPN= (**ULBHPN**)

Definition 3. The **U-Lesson Behavior Hierarchical High-Level Petri Nets** (**ULBHPN**) is a 7-tuple

ULBHPN=(**S**, **SN**, **SA**, **PN**, **PT**, **PA**, **PP**), where

1. **S** is a finite set of pages, where **S**=**ULPPN**$\cup$**ULAPN**. Each page $\mathbf{s_i} \in \mathbf{S}$ is a non-hierarchical high level Petri net: $(P_{s_i}, T_{s_i}, \Sigma_{s_i}, C_{s_i}, A_{s_i}, G_{s_i}, E_{s_i}, IN_{s_i})$.

   The sets of net elements are pair-wise disjoint.

   $\forall s_1, s_2 \in S, s_1 \neq s_2, (P_{s_1} \cup T_{s_1} \cup A_{s_1}) \cap (P_{s_2} \cup T_{s_2} \cup A_{s_2}) = \phi$
2. **SN** is a finite set of substitution nodes, where **SN**=$\mathbf{T_A}$, $\mathbf{T_A}$ in **ULPPN**.
3. **SA** is a page assignment function, where **SA**: **SN**→**S**, **S**$\in$**ULAPN**.
4. **PN** is a set of port nodes, where **PN**$\subseteq$**P**, **P**$\in$**ULAPN**.
5. **PT** is a port type function. **PT**: **PN**→{**in**, **out**}.
6. **PA** is a port assignment function, where **PA**: **SN**→Binary relation such that 1) Socket nodes are related to port nodes. 2) Socket nodes are of the correct type. 3) Related nodes have identical colour sets and equivalent initialization expressions.
7. **PP = ULP** is a prime page.

Example of ULBHPN is shown after the definition of ULPPN and ULAPN.

Definition 4. The **U-Lesson Process High-Level Petri Nets** (**ULPPN**) is a 8-tuple:

ULPPN = (P, T, Σ, C, A, G, E, IN), with **m** places and **n** transition. It is the prime page of **ULBHPN**, where

1. **P** is a finite set of places, where **P** = {$p_1, p_2, \ldots, p_m$}. **P** includes three types of places: $\mathbf{P}_\alpha$ denotes the connector or trigger event between learning activities. $\mathbf{P}_\beta$ denotes the parameters set by user or previous activity, that is required in the learning activity $\mathbf{P}_\beta$ connects to. $\mathbf{P}_\gamma$ denotes the specified context constraint activating a learning activity and the data is obtained from the frame of the specified context. Besides, we use $\mathbf{P_{start}}$ and $\mathbf{P_{end}}$ to represent the starting place and ending place of U-Lesson process.
2. **T** is a finite set of transitions, where **T** = {$t_1, t_2, \ldots, t_n$}. **T** includes three types of transitions: $\mathbf{T_A}$ denotes a learning activity and the set of $\mathbf{T_A}$ is **SN** $\in$ **ULHHPN**. $\mathbf{T_{start}}$ and $\mathbf{T_{end}}$ denotes the starting transition and ending transition of U-Lesson process respectively.
3. **Σ** is the non-empty finite color sets of tokens. Token represents the parameter or

context constraint of an activity. $\mathbf{C}_{\alpha}$ is a special colour and denotes the ordinary colour, corresponding tokens without information, which is applied to initialize or trigger a learning activity.

4. **C** is colour function, where **C**: $\mathbf{P} \rightarrow \mathbf{\Sigma}$.
5. **A** is union of input arcs **I** and output arcs **O**, where $\mathbf{A}=\mathbf{I} \cup \mathbf{O}$. **I** is a finite set of the input arcs $\overrightarrow{PT}$ from a place to a transition, where $\mathbf{I} \subseteq (\mathbf{P} \times \mathbf{T})$. **O** is a finite set of the output arcs $\overrightarrow{TP}$ from a transition to a place, where $\mathbf{O} \subseteq (\mathbf{T} \times \mathbf{P})$.
6. **G** is a guard function, where **G**: **T→Boolean expression**. All variables in **G(t)** must have types that belong to **Σ**. The firing rule **G(*t*)** of a transition ($\mathbf{t} \in \mathbf{T}$) is defined as "*if-else*" form. The guard function can generate specific sequencing behaviors. In **ULPPN**, we define the following guard functions:

   G(T_A): define the prerequisite rules of learning activity and specify whether a learner is ready or not to learn the activity according to the context of environment or leaner status.

   G(T_{start}), G(T_{end}): define the prerequisite rules to specify if the information is ready for the learning process to start or finish.
7. **E** is an arc expression function, where **E**: **A→Expression**. It maps arc into an expression which must be of type of its places' colour. It denotes the information that how many and which kinds of token colors should be removed from the input places or added to the output places.
8. **IN** is an initialization function, where **IN**: **P->Initialization Expression**. The initialization expression is defined as "*Colour=Initial Value*" form.

Example

Figure 10 is an example of ULPPN which consists of four activities. The learning process starts in 4th of May and contain sequence of Content Delivery activity which take a parameter P_{β} to specify the content should be delivered. If learners didn't finish the reading until 20:00 on May 10th, they are reminded by a reminder activity.

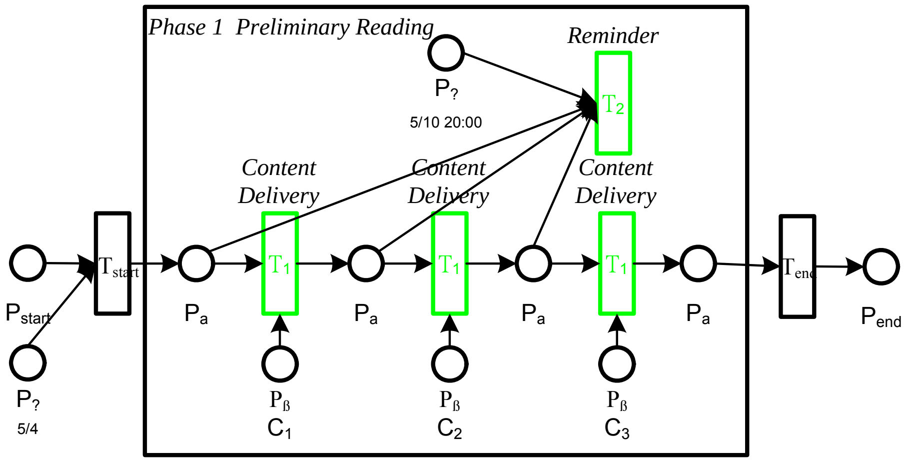

Figure 10: Learning process modeling with ULPPN example

Example

Figure 11 is an example of ULPPN which consists of five activities. The learning process is a collaborative learning instruction design and starts when learners arrive in the exhibition. Firstly, learners are divided into two groups by the grouping function of collaborative supplementary and take “Hall A” or “Hall B” activities according to their group assigned. Then, learners wait for each other to finish the previous activity by the round-up function of collaborative supplementary. Finally, learners take the Discussion activity to have a group discussion to conclude the lesson.

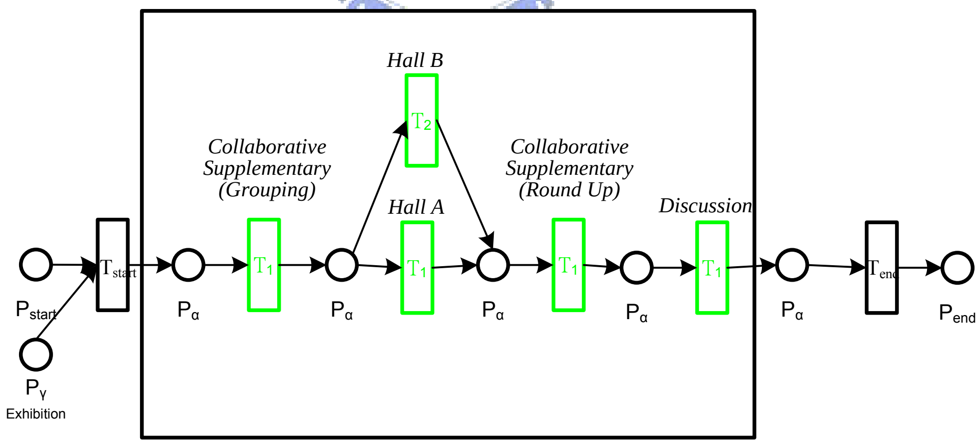

Figure 11: Concurrent learning process modeled by ULPPN example

Definition 5. The **U-Lesson Activity High-Level Petri Nets** (**ULAPN**) is a 8-tuple:

ULAPN = (P, T, Σ, C, A, G, E, IN), with **m** places and **n** transition. It is subpage of **ULBHPN**, where

1. **P** is a finite set of places, where **P** = $\{p_1, p_2, \ldots, p_m\}$. **P** includes three types of places: P_α denotes the connector or trigger event between an actions. P_β denotes the

parameters set by user or previous action, that is required in the action $\mathbf{P}_\beta$ connects to. $\mathbf{P}_\gamma$ denotes the specified context constraint activating an action and the data is obtained from the frame of the specified context.

2. **T** is a finite set of transitions, where **T** = {t_1, t_2,…, t_n}. **T** includes three types of transitions: $\mathbf{T_F}$ represents the frame actor, which perform a designated operation **OP**={"**SetValue**", "**GetValue**"}with a specific frame in a learning activity. The **GetValue** operation is to get value from "*FrameInstance.Slot*" or a variable holder to an output place. The SetValue operation is to set "*FrameInstance.Slot*" or a variable holder by the value of an input place. $\mathbf{T_{start}}$ and $\mathbf{T_{end}}$ denotes the starting transition and ending transition of the learning activity respectively.
3. **Σ** is the non-empty finite color sets of tokens, token represents the parameter data of actions in learning activity, which records the context information, service parameters and learning progress. $\mathbf{C}_\alpha$ which is a special colour denotes the ordinary colour, corresponding tokens without information, which is applied to initialize or trigger an action.
4. **C** is colour function, where **C**: **P**→**Σ**.
5. **A** is union of input arcs **I** and output arcs **O**, where $\mathbf{A}=\mathbf{I}\cup\mathbf{O}$. **I** is a finite set of the input arcs $\overrightarrow{PT}$ from a place to a transition, where $\mathbf{I}\subseteq(\mathbf{P}\times\mathbf{T})$. **O** is a finite set of the output arcs $\overrightarrow{TP}$ from a transition to a place, where $\mathbf{O}\subseteq(\mathbf{T}\times\mathbf{P})$.
6. **G** is a guard function, where **G**: **T**→**Boolean expression**. All variables in **G(t)** must have types that belong to **Σ**. The firing rule **G(t)** of a transition ($\mathbf{t}\in\mathbf{T}$) is defined as "*if-else*" form. The guard function can generate specific sequencing behaviors. In ULAPN, we define the following guard functions:

   G($\mathbf{T_S}$): specify whether the prerequisite context information is ready to apply the action.

   G($\mathbf{T_{start}}$), **G($\mathbf{T_{end}}$):** define the prerequisite rules to specify if the information is ready for the learning activity to start or finish.
7. **E** is an arc expression function, where **E**: **A**→**Expression**. It maps arc into an expression which must be of type of its places' colour. It denotes the information that how many and which kinds of token colors should be removed from the input places or added to the output places.
8. **IN** is an initialization function, where **IN**: **P->Initialization Expression**. The

initialization expression is defined as "*Colour=Initial Value*" form.

A graphic representation of ULAPN is shown in Figure 12.

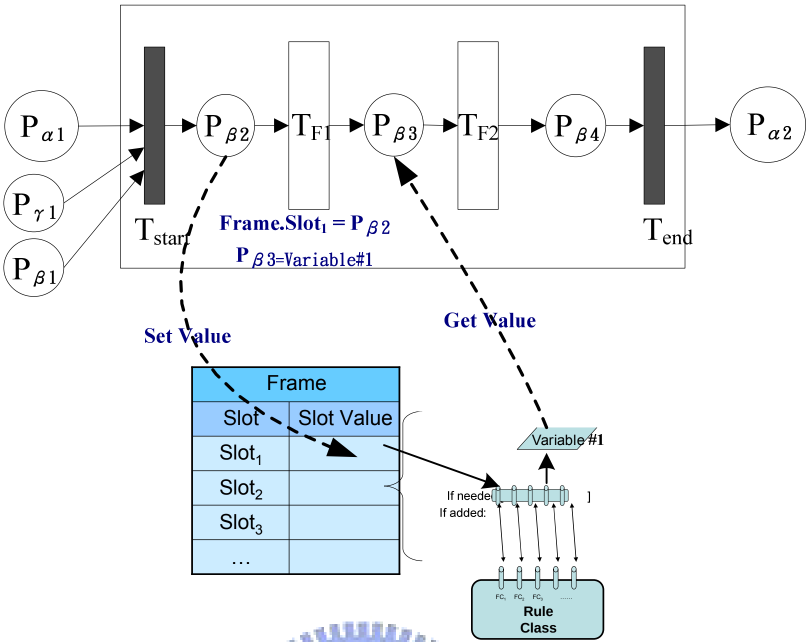

Figure 12: The elements and operations of ULAPN

Example

A discussion activity shown in Figure 13 is modeled by ULAPN. The activity finds the location nearby of the learner, then send short message to the learners in the location nearby to invite them.

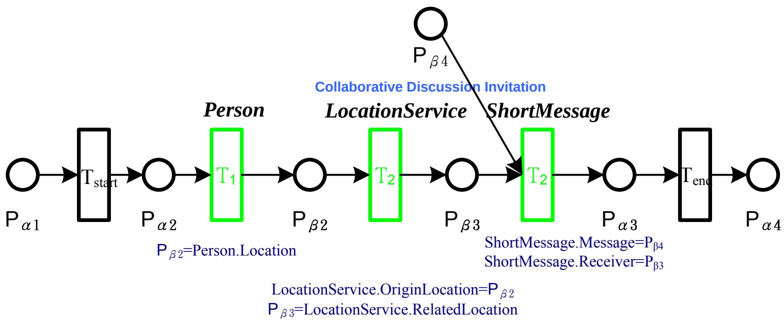

Figure 13: A discussion activity modeled by ULPPN

A graphic representation of the ULBPN is shown in Figure 14. ULPPN and ULAPN model individual learning process and learning activity respectively. ULBHPN relates the activity to the process.

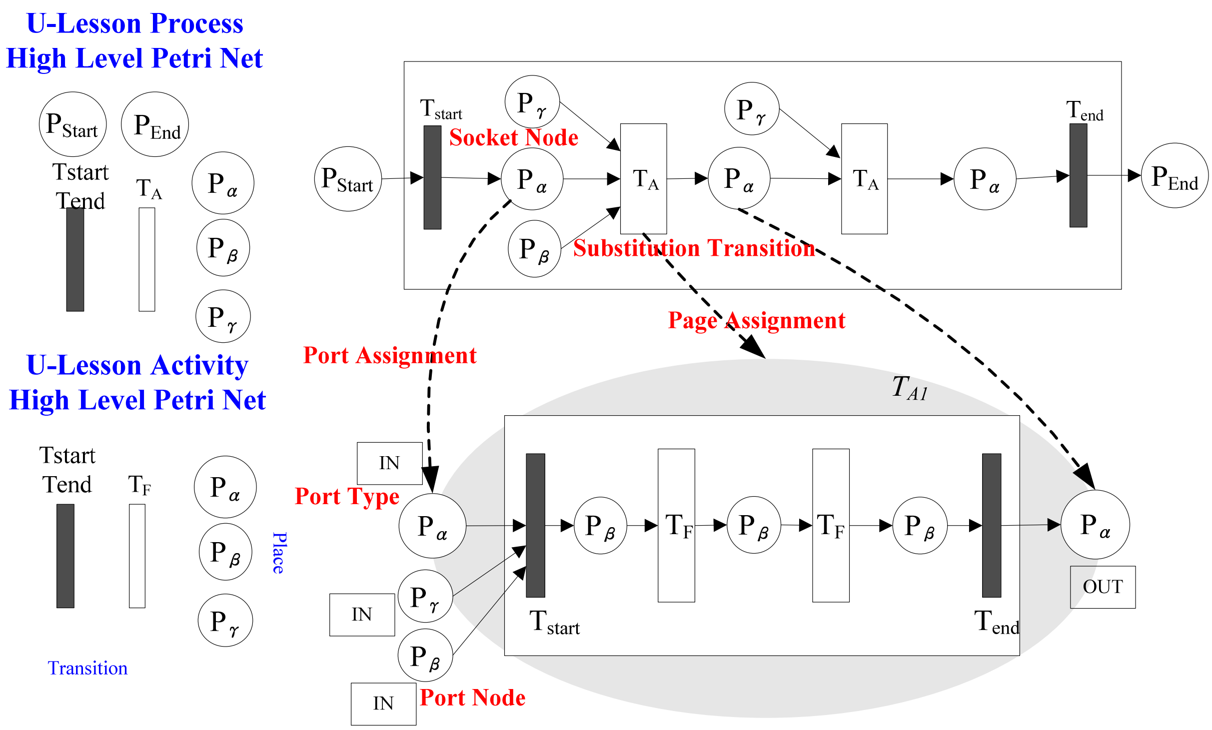

Figure 14: The elements of ULBPN

Chatper 5 U-Learning Scenario Modeling

5.1 FILM Design Process

FILM is a model to design process of U-Lesson and there are many possible ways to use the model. In order to examine how FILM could be used to model the process of U-Learning application, we describe a generic reference design process, FILM Design Process (FILM DP) (See Figure 15). It is a bottom-up design approach consisting of seven phases, “Frame Design”, “Activity Design”, “Learning Objective Analysis”, “Process Design”, “Activity Instantiation”, “Teaching” and “Evaluation”. Among them, “Frame Design”, “Activity Design”, “Process Design” and “Activity Instantiation” are model design phases. “Teaching” is model use phase. Apart from the design phrases, two role, teacher and engineer, are involved in the process. Each phase would be done with one or both of the roles.

At the beginning, Frame Design Phase should be taken at first, engineer designs individual frame for the learning service, sensor and context service. This phase would be repeatedly done each time the service or context is changed. After the Frame Design Phase, the material layer of FILM should be constructed. Then, in Activity Design Phase, teacher and engineer cooperatively design individual activity and the activity layer of FILM will be done. After the first two phases, we have prepared the fundamental part of the model. Then, we go on the learning objective specific part of the model. In Learning Objective Analysis Phase, teacher analyzes and chooses suitable activities for the learning objective. In Process Design Phase, teacher uses the selected activities to author the learning process and sets necessary parameters, e.g. socket node. In Activity Instantiation Phase, according to the real situation, teacher customizes the selected activity by setting or modifying parameter and context condition. After that, we have finished designing the model for a specific learning objective and the model can be executed in Teaching Phase. Finally, after each teaching iteration, Evaluation Phase should be taken to evaluate the efficacy of learning and then make improvement to the instruction design, which involved process and activity modification.

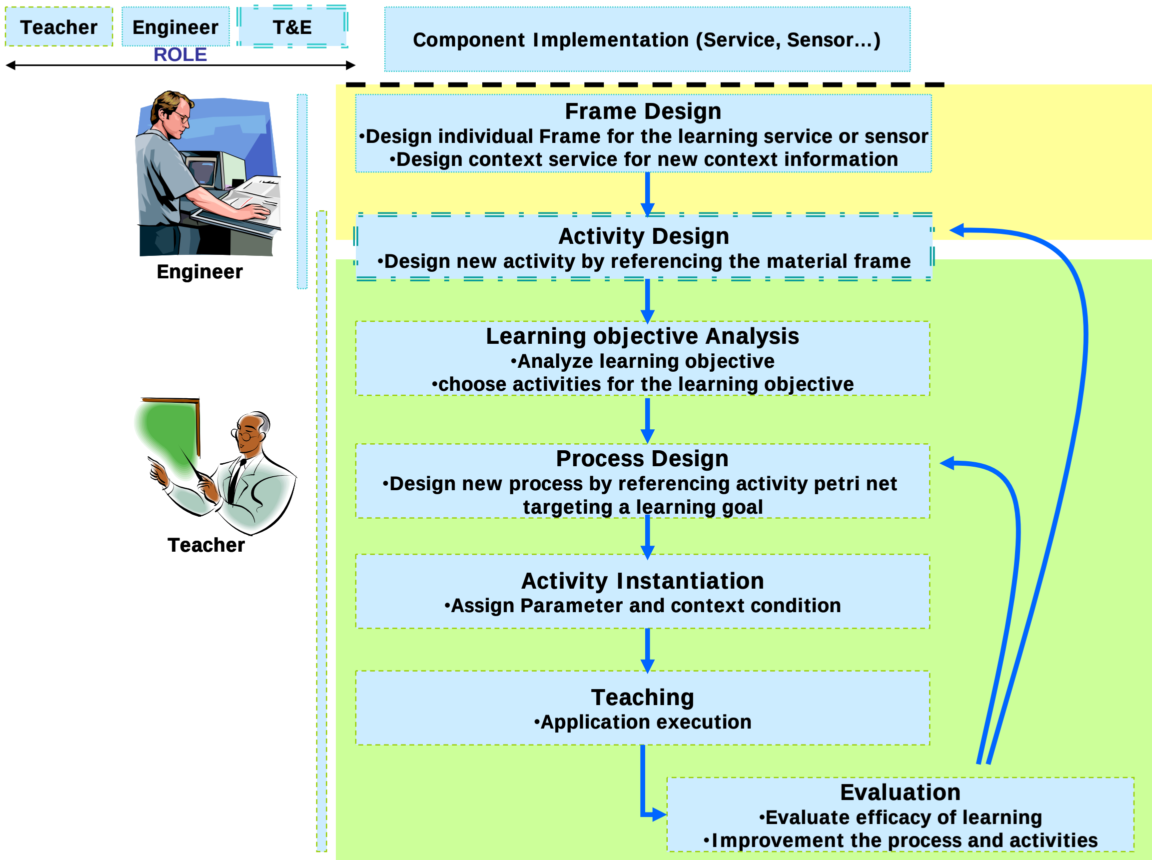

Figure 15: FILM Design Process

5.2 The FILM Model of Biology Lesson

In this section, FILM is used to apply to a scenario. The whole process model and the most representative activity model are described as follows.

Scenario

Instruction Design of Teacher Mary

Teacher Mary designed a two phase instruction design for a biology lesson and the objective of the course is to teach plants of Littoral Forest and Tropical Rain Forest Area. The two phases are preliminary reading and outdoor activity. The duration of preliminary reading phase is one week and should complete before the second phase started on 11th of May, Mary asked students to study three concepts about the learning topic and she thought that it is good for students to study the material at home and at night. The reading should be finished before the next phase, so Mary asks the system to remind students if they haven't finished the job. In the second phase, Mary wants the system to remind students to engage the activity at 8:00AM on the outdoor activity day and she has designed two guiding activities for students to choose.

One is passive spot guide which detects scenic spot students are interested in and provides corresponding assessment, then provides supplementary content according to the assessment result. The other is active showroom guide which directs students to a predefined learning path. With each scenic spot, corresponding content is provided. After finishing either guiding activities, students should have a collaborative discussion which begins by finding classmates nearby, and then invites them to have a discussion by message service. Besides having the major guiding activities, Mary also asks system to provide two supplementary services in the exhibition. The map service provides an overview map of the exhibition and could trace students and their participants' location. To ensure safety, if students move close to a poison plant, an alert is delivered and additional content about safety should be provided.

Learning Trace of Student Bob

In the daytime of 4th of May, Mary has announced to her students about the coming lesson. At that night, Bob was surfing the Internet with his computer at home, when it was 9:00PM, a message was delivered to his computer to ask him that concept C_1 was available. He accepted the invitation and content suitable for his computer was delivered. At the same time next day, he was playing game with his PDA and a message was delivered to ask him that concept C_2 was available. He accepted the invitation and an adapted content for his PDA was delivered. On the next few days, he had to do another projects and rejected to learn any more. At the night 10:00PM, 10th of May, he received a reminder message that he should finish the study at that night.

In the morning of 11th of May, he had been reminded that there was an outdoor activity. After he arrived in the activity destination – exhibition, system asked him to choose one of the two guiding schemas through his PDA. He had chosen the passive spot guide. When he approached a plant, a corresponding assessment was displayed to ask him about the plant. If he failed to answer the question, supplementary content was provided. Sometimes, he moved close to plants and wanted to touch the plant with curiosity. An alert was delivered and additional content about safety was provided when the plant was poison. During the activity, a map service was available in his PDA and he could have an overview to the exhibition and observed where he was instantly. After that, he was asked to have a collaborative discussion with other classmate. System automatically found a classmate nearby, Michelle, and then invited her to have a discussion by message service.

FILM of the biology lesson

The ULPPN of the scenario is shown in Figure 16. In order to show the model more clearly, we have added two blocks in the figure to divide the two phase. In the Preliminary Reading Phase, four transitions are involved in, three of them marked with T_1 are identical and they are instances of Content Delivery Activity which provides adaptive content to the device learner using. The place P_β connected to the transition T_1 is the parameter to the activity and it specifies the concept to be delivered in the activity. The transition T_2 is a Reminder Activity constructed by a reminder service. It is triggered with the conditions that the time is "5/10 20:00" and either one of the content delivery activities hasn't started. In the Outdoor Activity Phase, five transitions are involved in. Reminder Activity is used again with a time trigger condition to remind learner about the outdoor activity. Then, a context constraint P_γ is added to ensure the following three activities should be taken after learner arrive in the exhibition. Only one of the SpotGuide Activity and ShowRoomGuide Activity could start depends on the learner's choice. After SpotGuide Activity or ShowRoomGuide Activity finished, Discussion Activity follows.

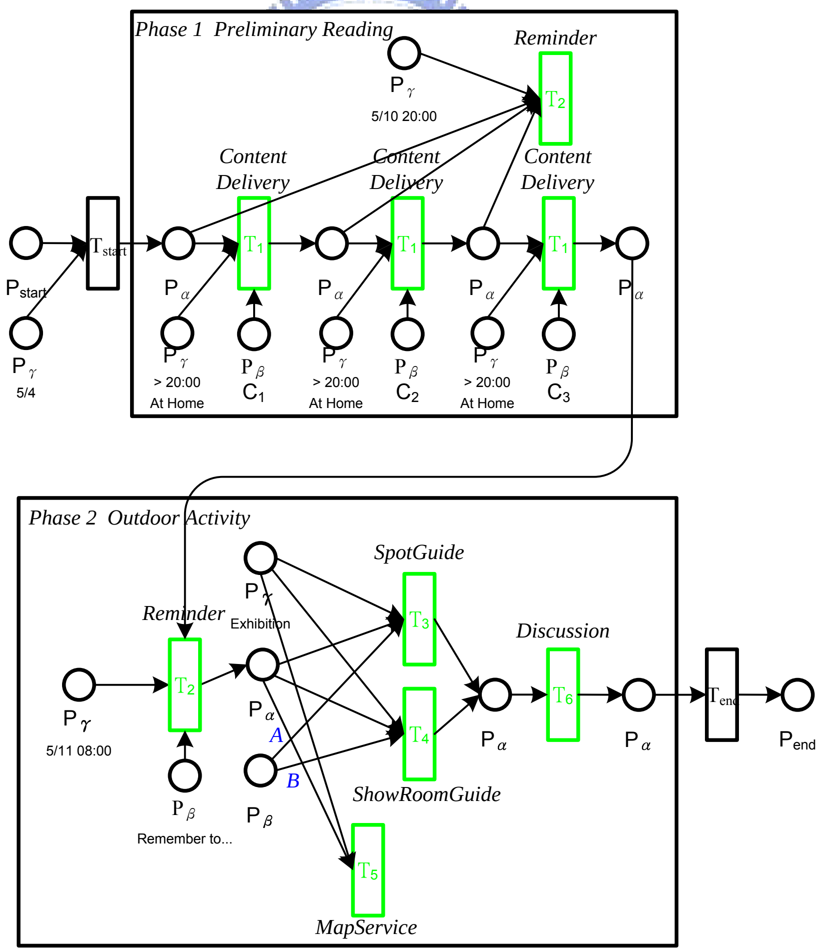

Figure 16: The ULPPN of the biology lesson

Context Service Modeling

We describe how context service used in the scenario could be modeled with ULMF.

Context service is the frame that providing context interpretation. Context including location, interaction and situation has been commonly used in ubiquitous application. We describe LocationService which uses in our scenario for location reasoning and we assume that it is also useful to other case. To simplify our discussion, we embedded the procedural attachment into the frame directly, so that the interface is not shown in the following example.

Location Service

Location-aware is very popular in the context-aware application; we define LocationService shown in Table 2 and provide the following functionality to this kind of application.

- Acquire the location of given sensible Id, e.g. RFID

  By setting CallerFrame and RFID, LocationService will call RegisterSensorsToReport procedural call which register sensors' discovery service with the sensible ID. When sensor senses the ID, it reports LocationService it's location by setting Location slot value. Whenever new location is reported, LocationService also reports it to CallerFrame.

- Obtain distance relative location to a given location.

  It is a common usage that we want to obtain a distance relative location to a given location, e.g. to obtain a location list which is near the given location.

  By setting DistanceType and OriginLocation, LocationService inferences to get the list of related location. The inference process queries the location ontology and use the node's distance as physical type, i.e., near means the node's distance equal to 1. According to the instance of location hierarchy shown in Figure 18, given an OriginLocation EC447 would obtain the relative location EC426.

Table 2: The ULMF of the LocationService

LocationService	
Slot	Slot Value
CallerFrame	
RFID	If added:[RegisterSensorsToReport()]
Location	If changed: [R_1: IF Exists CallerFrame THEN CallerFrame.Location = Location]
DistanceType	{Near, Far}
OriginLocation	If changed: [IF DistanceType = Near THEN N_o = OntologyFindNode(LocationOntology, OriginLocation); N_p = OntologyParentNode(LocationOntology, N_o); N_c = OntologyChildNodes(LocationOntology, N_p); RelatedLocation = N_c – {OriginLocation};]
RelatedLocation	

Figure 17 shows an example of using the location acquisition function. In step 1, PDA uses the LocationService by setting CallerFrame and RFID with its information and then LocationService call RegisterSensorsToReport() to register sensors' discovery service in step 2. Sensor frame is updated when the physical sensor sensed an object. Suppose in step 3, sensor#003 has sensed the RFID#001, RFID#001 is added to the member slot of sensor#003. Every time a member of sensor is added, sensor finds LocationService which is serving the member and report to the LocationService in step 4. Finally, in step 5, LocationService reports the location to PDA.

The rules in current context service are very simple. Reasonable inference could be obtained by considering more information and acquiring more advanced rules from domain expert. Interaction and situation services can be constructed in the similar way.

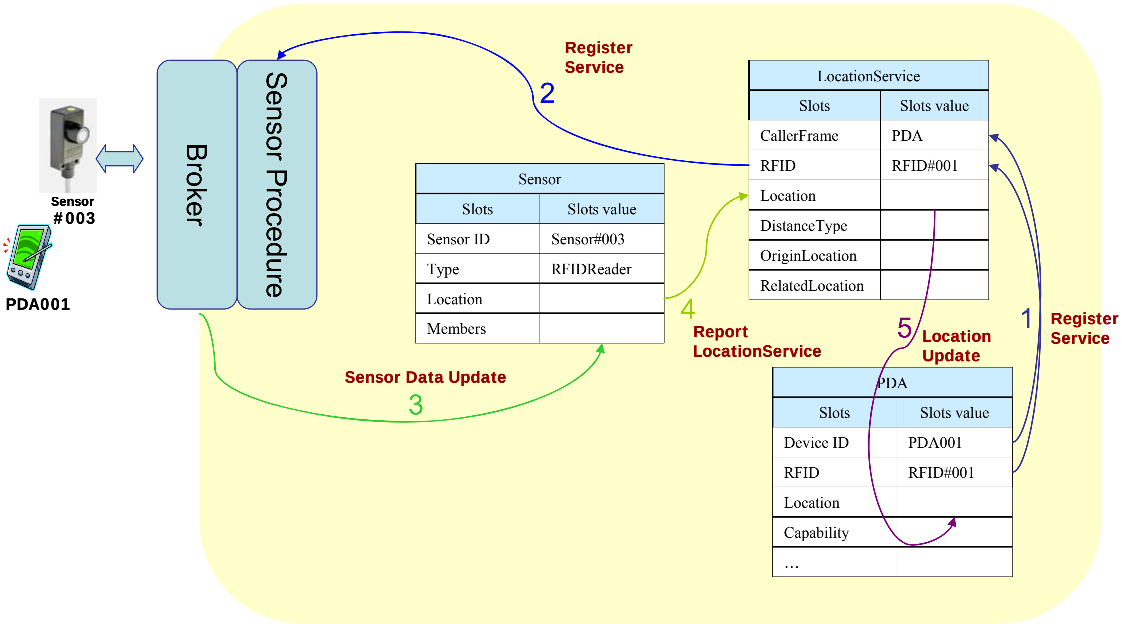

Figure 17: A LocationService reasoning example

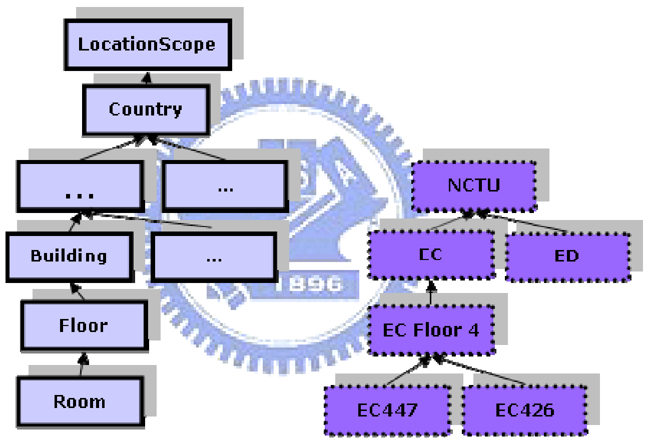

Figure 18: An instance of location hierarchy

Chapter 6 Conclusion and Future work

In this thesis, we applied knowledge based approach to the problem of development of U-Lesson. To author a U-Lesson, we first acquire the necessary knowledge from pedagogical expert and engineer using the proposed model as the knowledge representation. At run time, the lesson can be provided by executing the elicited knowledge with a specific engine. We proposed a knowledge representation **Frame-based Interaction and Learning Model (FILM)** to model a U-Lesson. FILM is a systematic and integrated method and consists of three layer, **U-Lesson Process Layer (ULP)**, **U-Lesson Activity Layer (ULA)** and **U-Lesson Material Layer (ULM)**, and two sub-models, **U-Lesson Behavior Petri Net Model (ULBPN)** which is designed for ULP and ULA, and **U-Lesson Material Frame Model (ULMF)** which is designed for ULM. The proposed model provides a new approach to easily construct U-Lesson and does have improvement in maintainability and extensibility of U-Lesson. Based on FILM, a generic reference design process, **FILM Design Process (FILM DP)** is also proposed to have a guideline of using FILM to design a U-Lesson. Finally, a U-Lesson is modeled by FILM. It shows that the FILM is a model that could take the knowledge of domain expert into consideration and a more flexible way to develop U-Lesson.

In the near future, an engine supporting the HHLPN process and frame reasoning should be constructed. In order to construct the FILM more easily and efficiently for teacher and engineer, a powerful knowledge acquisition process and the corresponding authoring tool should be investigated. In addition, activity patterns could be constructed for the activities widely used in U-Learning scenario. Furthermore, we will have our idea work in practice, propose system architecture and use it in real ubiquitous environment. The FILM system will be further implemented and experimented for more practical and performance evaluations.

References

[1] G. D. Abowd, et al., "CyberGuide: A Mobile Context-Aware Tour Guide". ACM Wireless Networks, 1997.97.

[2] M. Baldauf, S. Dustdar, "A Survey on Context-aware Systems", TUV-1841-2004-24, Technical University of Vienna, Nov 2004

[3] G. Chen; D. Kotz, "A Survey of Context-Aware Mobile Computing Research", Technical Report TR2000-381, Dartmouth College, Computer Science, 2000

[4] H. Chen, et al., "SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications", Proceedings of International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004

[5] H. Chen, "An Intelligent Broker Architecture for Pervasive Context-Aware Systems", , UMBC, PHD, 2004

[6] Z. Cheng, et al., "A personalized ubiquitous education support environment by comparing learning instructional requirement with learner's behavior", Proceedings of International Conference on Advanced Information Networking and Applications, 2005

[7] R. David; H. Alla, Petri Nets and Grafcet, Prentice Hall, 1992

[8] A. K. Dey, D. Salber, and G.D. Abowd, "A Context-Based Infrastructure for Smart Environments", Proceedings of the 1st International Workshop on Managing Interaction in Smart Environments, 1999

[9] A. K. Dey, et al., "The Conference Assistant: Combining context-awareness with wearable computing". Proceeding of the 3rd International Symposium on Wearable Computers, 1999

[10] W. Dick, L. Carey and J. Carey, Systematic Design of Instruction, 6th, Allyn & Bacon, 2004

[11] R. Fikes, T. Kehler, "The role of frame-based representation in reasoning", Communications of the ACM, vol. 28, no. 9, pp. 904-920, 1985.

[12] K. Jensen, Coloured Petri Nets, Basic Concepts, Analysis Methods and Practical Use, Springer-Verlag, 1992

[13] Y. T. Lin, S. S. Tseng and C. F. Tsai, "Design and implementation of new object-oriented rule base management system", Expert Systems with Applications, vol. 25, no. 3, pp. 369-385. Oct. 2003

[14] K. Lindquist, New Applications for Ubiquitous Learning, Open Consultation Process

Report, 2001

[15] O. Marjanovic, "Towards A Web-Based Handbook of Generic, Process-Oriented Learning Designs", Educational Technology & Society, vol. 8, no. 2, pp. 66-82, 2005

[16] D. L. McGuinness, F. V. Harmelen, "OWL Web Ontology Language Overiew", http://www.w3.org/TR/owl-features/, 2004

[17] M. Minsky. A framework for representing knowledge, The Psychology of Computer Vision, McGraw-Hill, New York, 1975

[18] M. Nelson, et al., "Hive: Distributed agents for networking things". IEEE Concurrency, vol. 8, no. 2, pp. 24-33, 2000

[19] H. Ogata and Y. Yano, "Context-Aware Support for Computer-Supported Ubiquitous Learning", Proceedings of the 2nd IEEE International Workshop on Wireless And Mobile technologies in Education, 2004

[20] J. Pascoe, "Adding generic Contextual Capabilities to Wearable Computers", Proceedings of the 2nd International Symposium on Wearable Computers, 1998

[21] G. Riley, Expert Systems – Principles and Programming, PWS, 1994

[22] K. Sakamura, N. Koshizuka, "Ubiquitous Computing Technologies for Ubiquitous Learning", Proccedings of the IEEE International Workshop on Wireless and Mobile Technologies in Education, 2005

[23] R. C. Schank, R. P. Abelson, Scripts, plans, goals, and understanding, Lawrence Erlbaum, 1977

[24] M. Weiser, The Computer for the Twenty-First Century. Scientific American, Scientific American, 1991

[25] S. J. H. Yang, "Context Aware Ubiquitous Learning Environments for Peer-to-Peer Collaborative Learning", Educational Technology & Society, vol. 9, no. 1, pp. 188-201, 2006

[26] G. Zhang, Q. Jin and M. Lin, "A Framework of Social Interaction Support for Ubiquitous learning", Advanced Information Networking and Applications, vol. 2, pp. 639-643, 2005