DIF Knowledge Management System: Bridging Viewpoints for Interactive System Design

Eui-Chul Jung
Institute of Design
Illinois Institute of Technology
350 N LaSalle 4th Floor
Chicago, IL, 60610, USA
jung@id.iit.edu

Keiichi Sato
Institute of Design
Illinois Institute of Technology
350 N LaSalle 4th Floor
Chicago, IL, 60610, USA
sato@id.iit.edu

Yong Chen
Dept. of Computer Science
Illinois Institute of Technology
10 W 31ST Street
Chicago, IL, 60616, USA
Chenyon1@iit.edu

Xiaoshan He
Dept. of Computer Science
Illinois Institute of Technology
10 W 31ST Street
Chicago, IL, 60616, USA
hexiaos@iit.edu

Tom MacTavish
Human Interface Labs.
Motorola
1303 E Algonquin Road
Schaumburg, IL, 60196, USA
tom.mactavish@motorola.com

Deborah Cracchiolo
Human Interface Labs.
Motorola
1303 E Algonquin Road
Schaumburg, IL, 60196, USA
Deborah.Cracchiolo@motorola.com

Abstract

Interactive systems have begun to adopt embedded and ubiquitous computing technologies in an attempt to effectively distribute information and functions in our activity space. Because of the pervasive nature of those technologies, it becomes critical to pursue deeper understanding of user needs and consistent incorporation of those needs into the system from a wide range of viewpoints. In order to enhance integration of multiple viewpoints throughout system development, the concept of the Design Information Framework (DIF) was previously introduced as a common information representation platform for interactive system development. This paper reports the development of the DIF Knowledge Management System (DIF-KMS) that provide a common information and tool platform for interdisciplinary collaboration for human-centered interactive system development. The DIF-KMS provides the following functions: 1) defines a set of DILs to establish an evolutionary but consistent ontology of the project, Project DIF (P-DIF), by incorporating different disciplinary viewpoints involved in the development, 2) manages multi-media information resources such as video, photo, sound, graphic, and text documents, 3) encodes data into different representation formats for different viewpoints, 4) translates the encoded data to an aspect model representation by selecting a viewpoint, and 5) composes composite aspect models and other types of representations such as scenarios to integrate information from different views. The DIF-KMS has been implemented as an Internet-based collaboration environment to support development resource sharing among remotely distributed users during interactive system development.

1   Introduction

Information technology is becoming more pervasive and ubiquitous in our work and daily life environments and is shaping our new interactive experience by integrating the information space and the physical space. In order to understand diverse aspects of the user experience and to bridge different disciplinary viewpoints involved in system development, it is critical to introduce a mechanism for sharing and integrating design information across different disciplinary viewpoints.

The Unified Modeling Language (UML) has been developed to provide modeling methods for describing software systems from different viewpoints using standardized representation methods [1, 2]. The diagrammatic representation in the UML can be used for sharing and integrating design information to manage the software development process. In the case of interactive systems, computing and networking functions are embedded in physical subsystems. The development of such systems involves a much wider range of disciplinary viewpoints. In addition, user interaction with those systems needs to be understood in relation to much more diverse dimensions than that of software systems. Because of the diversity of the types of systems from small information devices such as cell phones to large scale public or industrial systems such as transportation systems, types of information, methods, and processes involved in the development widely vary from project to project. The development environment for such systems requires flexibility to handle a wide range of information formats from qualitative to quantitative and descriptive to symbolic. It must also handle information formats from diverse viewpoints such as cognitive, social, and hardware engineering.

The concept of Design Information Framework (DIF) was previously introduced as a mechanism to provide a common information platform for developing the type of interactive system described above. The DIF enables design information to be consistently organized, translated, integrated, and communicated [3, 4, 5, 6]. The goal of this research is to implement a DIF-based Knowledge Management System (DIF-KMS) applicable to the practice of interactive systems research and development. The benefit of using DIF as the foundation of interactive system development is to implement an explicit mechanism for information sharing, for interdisciplinary collaboration, and for archiving project resources for reuse. It also works as a unified platform for developing methods and tools for interactive system development.

2   Design Information Framework (DIF)

The DIF provides a mechanism for setting a basic project ontology which frames design information for the domain of concern. In order to make the ontological structure flexible and interpretable for each project, the DIF uses a small number of primitive information types called Design Information Primitives (DIPs) that cannot be further decomposed into other types of information. These basic types of information, Entities, Attributes, Acts, and Time are used as basic units to compose all other types of information. "Entities" are tangible and intangible objects, "attributes" are the properties for describing the nature of objects, and "acts" are performed by objects [1, 2, 3, 7]. They are used as building blocks for defining higher level concepts of design information used in the design process such as "goals," "functions," and "profiles" [3, 4].

2.1   The Structure of DIF

As shown in Figure 1, the DIF consists of two layers of information types, Design Information Primitives (DIPs) and Design Information Elements (DILs). The concept of DIP as basic
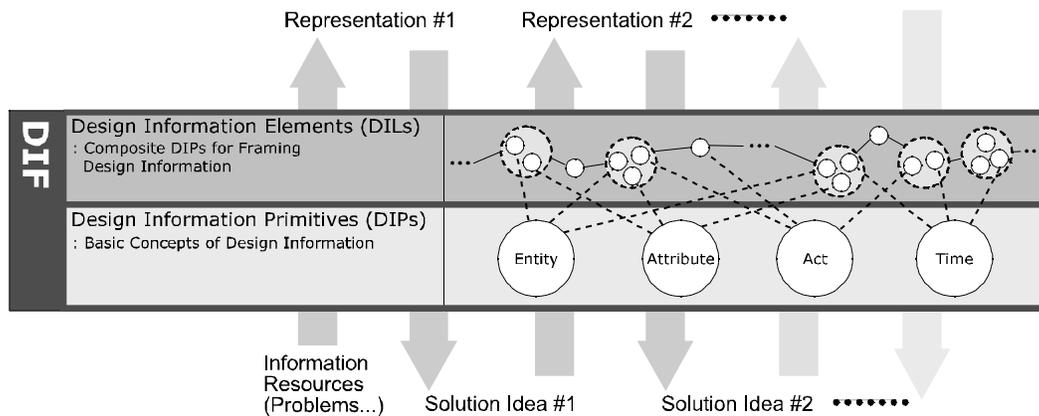
Figure 1. Basic Structure of DIF for Interactive System Design

vocabularies consists of four types in the DIF: entities (PE), attributes (PR), acts (PA), and time (PT) [3, 4]. Each DIP is categorized into classes as follows:

- Entities: general (PE), agent (PEa), object (PEo), and space (PEs)
- Attributes: constant (PR), state (PRs), and Location (PRloc)
- Acts: active (PA), stative (PAs), and relational (PAr)
- Time: time point (PT), time duration (PTd), temporal relations (PTr), and local time (PTl)

In addition, the categorization scheme was introduced to allow detailed and unique description of DIP instances in the DIF-KMS implementation. For example, "run" was an instance of "acts" by the original DIF concepts. However, it is still a category not an instance of "acts" for describing such as "run fast" and "run with 10 miles per hour." In this case, the example is the instances of "run." The structure of DIPs, particularly "acts" and "attributes," was modified from the original to accommodate such diversity of descriptions. The definition and structure of DIPs remain constant so that DIPs function as a consistent reference for flexible definitions of DILs.

DILs represent higher level concepts of design information such as "activities," "goals," and "functions" that are composed of DIPs and other DILs. For example, an "activity" is a series of actions that can be described as the sentence "agents (entity) do something (act) on an object (entity) at time (time)." Therefore, DIL representing an activity can be denoted as LAct: [Pea, PA PEo, PT]. DILs have an open structure to allow definitions of different types of DILs depending on the nature, goals, and interests of the project [3, 4].

2.2 P-DIF

Every system development project has unique concerns reflecting its position in the surrounding situations, and stakeholders and project participants with different perspectives. The concept of Project-DIF (P-DIF) is defined as an ontology representing concerns of a specific project [3, 4]. P-DIF is composed of multiple sets of DIF elements representing the different interests of project groups. Table 1 shows the examples of P-DIF for an automotive information system design. Group 1 is a team responsible for understanding user experience regarding such aspects as actions,

Table 1: Example of P-DIF for Automotive Information Systems Development

Group	Viewpoints	Set of P-DIF	
		DILs	Format (Sets of DIPs)
Group1: User Experience	Analyze user's experience and context	User Actions	PEa, PA, PEo, PRs, PEo, PT, PT
		User Environments	PEc, PRs, PT, PT
		User Profiles	PEa, PR
		Information Sources	PEo, PRs, PEa, PT, PT
		User Task	PEa, PA, PEo, PEo, PT, PT
Group 2: Intelligent System	Analyze device's properties, functions, and states	Device Properties	PEo, PR
		Device States	PEo, PRs
		Device Functions	PEo, PA, PEo, PEa, PT, PT

environments, profiles, tasks, and their relationships. Group 2 is an intelligent system team responsible for designing adaptive software subsystems.

3   DIF Knowledge Management System

The basic system architecture of the DIF Knowledge Management System (DIF-KMS) was developed to facilitate interdisciplinary collaboration for designing interactive systems. The database server and its application packages manage multiple user access in order to provide collaborative development environments for remotely distributed users. The database server can locally run to enable users to work in the standalone mode for mobility. The data structure is designed for compatibility with the standard formats of spreadsheet, documentation, and generic graphic applications.

3.1   System Overview

The DIF-KMS consists of three subsystems: database server, foundation package, and application package. The database server handles the meta-structure of projects and consists of seven key tables and their sub-tables. Its primary roles are to perform project management functions, to generate information frameworks on P-DIF templates, and to handle instances, resources, aspect models, scenario data, and users' profiles. The foundation package consists of several classes for implementing shared functions of the system. For example, modules of the application packages such as the project wizard, data encoder, aspect modeler, scenario composer, and data analyzer use the same data structure of DILs. The classes for dynamically creating, updating, and deleting the tables of DILs are designed into the foundation packages.
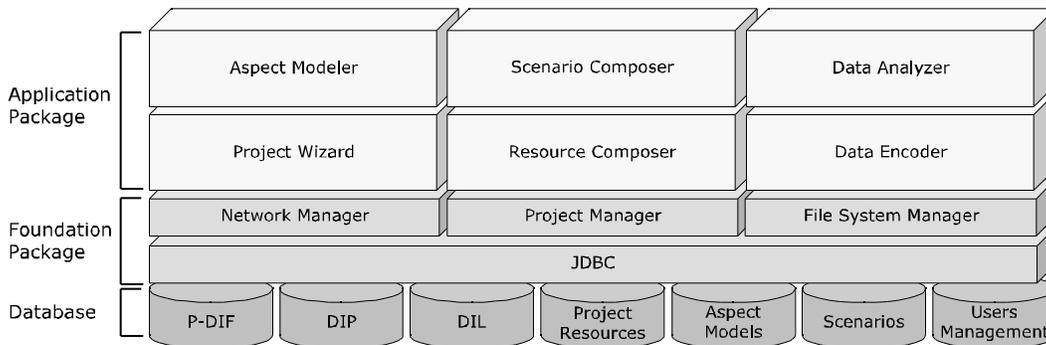
Figure 2. System Architecture of DIF-KMS

Figure 2 shows the system architecture of DIF-KMS. The mechanisms for implementing key functions of DIF-KMS are as follows.

1) Each DIP can have different categorization schemes depending on the nature of the project as shown in Table 1. Each DIP is also represented in its unique data structure.

2) The DIF meta-data structure enables the P-DIF definition to accommodate different theories and methodologies reflecting different disciplinary concerns and to manage data consistency among different representations.

3) Resource management, editing, and media control functions such as synchronization, annotation, and tagging support flexible ways of using the information resources from different viewpoints.

4) DILs are dynamically created by referring the meta-data structure of P-DIF. This allows users to introduce DIL's representing new information concepts in relation to existing DILs or DIPs. The project ontology therefore can evolve as the project develops.

5) The visualization mechanism using P-DIF structure allows user selection of variables to form aspect models and represents them in diagrammatic forms such as timeline, object-relations, state-transitions, sequences, and spatial layouts.

6) The modular scenario scripting structure effectively provides consistent representation, accommodation of multiple media, combinatorial exploration with alternative description modules, and alternative description formats. This structure also supports transportability of information between narrative and analytical formats.

3.2 DIF-Based Database Design

The P-DIF is defined as a set of DILs or DIPs based on the nature of a project as shown in Figure 3. The dynamically generated tables for DIL instances work for different project members who use their own language for the project. For consistency, every record in the DIL instance tables generated by "Formula" field is stored in the DIP instance tables that maintain its structure. This DIP structure makes design information transferable and reusable.

Figure 3 shows the basic scheme of the database design. The table for P-DIF is designed for archiving DIF-based projects. The "Project Owner" field is for setting privileges for projects in
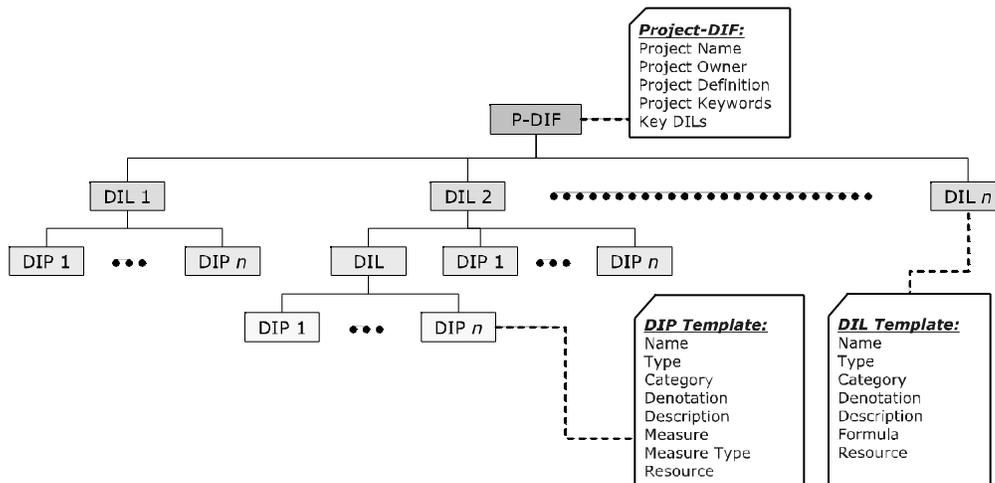
Figure 3. Database Scheme for DIF-KMS

multi-user mode. The table for the DIL template contains two child tables for "Formula" and "Resource." "Formula" field is used as references to generate DIL tables for storing instances of the DIL template. The field properties of "Resource" can contain multiple resources for each instance of DILs and DIPs to indicate origins of the data. The DIF-KMS also provides default DILs. For visual data encoding of spatial layouts, two DILs, "Entity Location" and "Location Change," are set as defaults. The DIP template in Figure 3 has a more complex structure defining instances of DIPs. If Conceptual Dependency Theory is the application case of the "Type" field, "give" (the value of "Category" in "Acts" DIP) belongs to ATRAN (the value of "Type" in "Acts" DIP). ATRAN includes several verbs such as give, take, and buy to describe abstract relationship such as possession, ownership or control [8]. By applying the ontology to DIPs, an object-relation diagram can be generated. "Measure" field is used to specify the level of DIPs with four "Measure Types," nominal, ordinal, interval, and ratio.

4 Application Modules of DIF-KMS

There are two layers of application modules implemented on the database. The first layer called "Foundation Package" consists of shared application classes such as file systems for importing/ exporting data from/to spreadsheet and image applications. The second layer, "Application Package," consists of applications implemented as needed in different projects by combining the shared application classes in the foundation package.

4.1 Setting Project Scopes and Viewpoints

Organizing a project is the process of building the basic information frame to best capture the nature of its domain and concerns of the sectors involved. The database scheme shown in Figure 3 provides meta-frame of the DIF for generating instances of P-DIF. Using the project wizard and/or the project properties windows, a project frame can be organized through the following four steps:

1) Set project definition: This is the step for creating instances on the "Project-DIF" table. The project name, definition, keywords, and owner are defined for DIF-Based project archiving system.
2) Set project resources: This is the step to set project resources such as interview files, documents, observation data, etc. These contain facts of the problems and concerns for the project.
3) Set DIP: This is the step to confirm DIPs as basic low-level frames of the project.
4) Set DIL: This is the step to set formulas with DIPs or other DILs. Depending on the project teams, different sets of DILs can be grouped as shown in Table 1.

4.2 Information Resource Management and Data Encoding

DIF-KMS has a function to structure resource files from various sources in the database. If some resources such as video clips have sequential relations, they are structured in the temporal dimension as shown in Figure 4. Questionnaires and reports are examples that have no sequential relations but Boolean relations might apply.

Data encoding creates instances of design information on templates of DIPs and formatted DILs. Figure 4 shows the relationships among resources and DIF-based data templates. When a DIL instance is encoded, the DIPs that compose the DIL instance are automatically encoded with the level of "Acts (PA)" and "Attributes (PR)."
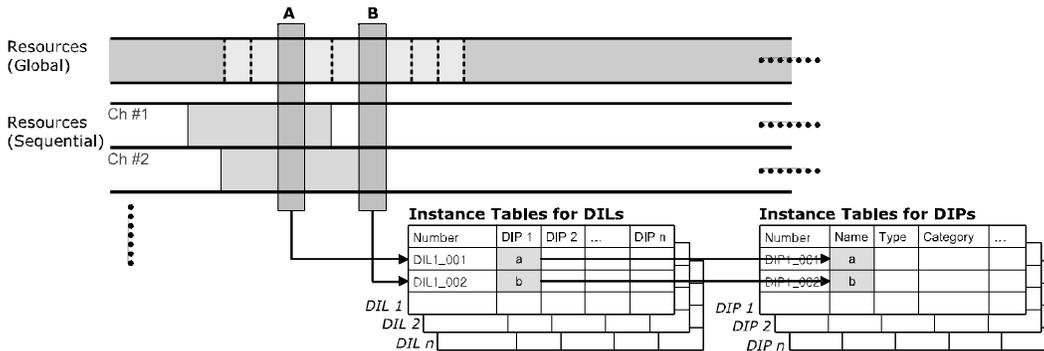
Figure 4. Relationships among Resources and DIF-based Templates

4.3 Aspect Models

P-DIF structure reflects viewpoints representing the concerns of the project through the DILs selected by participating members. This is a basic mechanism for creating diverse representation formats using DIF-based templates as shown in Figure 5. Aspect models can be diagrammatically represented by mapping the encoded data in the instance tables to visual objects in the display space. The data also can be mapped to the table format of the Modular Structured Scenario in text scripts.

4.4 Modular Structured Scenarios

Scenarios are used for many different purposes for a wide range of audiences because of their capability to carry rich qualitative information in plain text formats [9]. For example, they might be used for descriptions of existing situations, problem identification and analysis, alternative solutions, evaluations, and communication of proposals. DIF-KMS provides the mechanism for generating modular structured scenarios that enhance the ease of understanding, manipulation, and control. The scenario format consists of columns representing categories of information that can be selected depending on the purpose of the scenario. Example information categories in the columns are users, actions, objects, intention, environments, and contexts. Modular scenarios can be easily translated to other more analytical formats of aspect model representation.
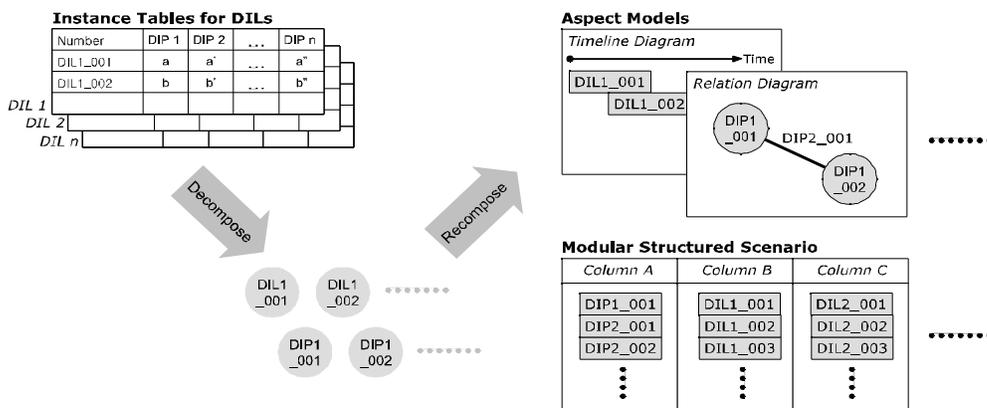
Figure 5. Information Representation Mechanism using DIF-based Templates

5 Conclusion

The goal of the DIF-KMS is to implement the concept of DIF for practical use in interactive system research and development. Through the implementation, the concept of DIF has been refined, the database structures for the DIF were developed, and the mechanisms of representing DIF structures were introduced. The following four features were demonstrated through applying DIF-KMS to different types of projects: 1) methods for framing design information reflecting multiple disciplinary viewpoints, 2) methods for managing and encoding design information with the meta-structure of P-DIF, 3) representation and visualization of design information with multiple aspect models, and 4) composition of modular scenarios to integrate multiple viewpoints.

This paper describes the first phase of the DIF-KMS development. In order to apply the DIF to real research and development practice to its fullest potential, further development of supporting functions is necessary. The incorporation of rapid prototyping functions is particularly desirable for effective use of DIF-KMS.

6 Acknowledgement

References

1. Quatrani, T. (2002). Visual Modeling with Rational Rose 2002 and UML (3rd ed.). Boston: Addison-Wesley.
2. Roberts, D., Berry, D., Isensee, S, and Mullaly, J. (1998). Designing for the User with OVID: Bridging User Interface Design and Software Engineering. Indianapolis: Macmillan Technical Publishing.
3. Lim, Y. and Sato, K. (2001). Development of Design Information Framework for Interactive Systems Design. Seoul, Korea: *Proceedings of the 5th Asian Design International Conference*.
4. Lim, Y. (2003). Design Information Framework for Integrating Multiple Aspects of Design. Chicago: PhD Dissertation in the Institute of Design at IIT.
5. Lim. Y. and Sato, K. (2003). Scenarios for Usability Evaluation: Using Design Information Framework and a Task Analysis Approach. Seoul, Korea: *Proceedings of the International Congress of Egonomics*.
6. Lim, Y. and Sato, K. (2003). Encoding Scenarios with Design Information Framework for the Generation of Multi-Aspect Models for System Implementation. Orlando: *International Conference on Systems, Cybernetics and Informatics.*
7. Yoshikawa, H. (1987). General Design Theory and Artificial Intelligence. *Artificial Intelligence in Manufacturing*, T. Bernold (Ed), Elsevier Science Publishers, 70-96.
8. Schank, R. and Ableson, R. (1977). Script, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures. New York: Lawrence Erlbaum Associates.
9. Carroll, J. M. (2000). Making Use: Scenario-Based Design of Human-Computer Interactions, MIT Press.