International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:4, No:9, 2010

An Automatic Tool for Checking Consistency between Data Flow Diagrams (DFDs)

Rosziati Ibrahim, Siow Yen Yen

***Abstract*—**System development life cycle (SDLC) is a process uses during the development of any system. SDLC consists of four main phases: analysis, design, implement and testing. During analysis phase, context diagram and data flow diagrams are used to produce the process model of a system. A consistency of the context diagram to lower-level data flow diagrams is very important in smoothing up developing process of a system. However, manual consistency check from context diagram to lower-level data flow diagrams by using a checklist is time-consuming process. At the same time, the limitation of human ability to validate the errors is one of the factors that influence the correctness and balancing of the diagrams. This paper presents a tool that automates the consistency check between Data Flow Diagrams (DFDs) based on the rules of DFDs. The tool serves two purposes: as an editor to draw the diagrams and as a checker to check the correctness of the diagrams drawn. The consistency check from context diagram to lower-level data flow diagrams is embedded inside the tool to overcome the manual checking problem.

***Keywords*—**Data Flow Diagram, Context Diagram, Consistency Check, Syntax and Semantic Rules

I. Introduction

MODELING of a system is an essential process in system development lifecycle. It will produce a system artifact called a system model. A system model can provide structure for problem solving. Models make it easier to express complex ideas, for example, an architect builds a model to communicate ideas more easily to clients. Models reduce complexity by separating those aspects that are unimportant from those that are important. The models should have traceability links or in other words they must be consistent.

In system development life cycle (SDLC), a system model can be developed by using Data Flow Diagram (DFD). DFD is graphical diagrams for specifying, constructing and visualizing the system. DFD is used in defining the requirements in a graphical view. In this paper, we will focus on DFD and its rules for drawing and defining the diagrams. We then develop the tool based on the DFD rules. The rules for consistency check between the diagrams are embedded inside the tool. This is to ensure the syntax for drawing the diagrams is correct and strictly followed. The tool automates the process of manual consistency check between data flow diagrams.

The rest of this paper is organized as follows. The review of DFD is in Section 2 and the discussion on the related works is in Section 3. Section 4 discusses the syntax and semantics rules of DFD and Section 5 presents the tool developed based on the rules. Finally, Section 6 concludes the paper.

Rosziati Ibrahim is with the Software Engineering Department, Faculty of Information Technology and Multimedia, Universiti Tun Hussein Onn Malaysia (UTHM), Parit Raja, Batu Pahat, 86400, Malaysia (phone: 607-453-8001; fax: 607-453-2199; e-mail: rosziati@ uthm.edu.my).

Siow Yen Yen is with the Software Engineering Department, Universiti Tun Hussein Onn Malaysia (UTHM), Malaysia (yenyen0831@hotmail.com).

II. Overview of DFD

SDLC is a process uses during the development of software system starting from planning until the implementation phase. Data flow diagramming, on the other hand, is used to produce the process model during the analysis phase. Process model is very important in defining the requirements in a graphical view. Therefore, the reliability of the process model is the key element to improve the performance of the following phases in SDLC.

According to Dennis et al. [1], SDLC is a process of understanding on how an information system can support business needs, designing the system, building the system and delivering the system to users. SDLC consists of four fundamental phases, which are analysis, design, implement and testing phases. In the analysis phase, requirements of a system are identified and refined into a process model. Process model can be used to represent the processes or activities that are performed in a system and show the way of data moves among the processes. In order to diagram a process model, data flow diagramming is needed. According to Dixit et al. [2], data flow diagram is a graphical tool that allows system analysts and users to depict the flow of data in an information system.

Normally, the system can be physical or logical, manual or computer based. Data flow diagram symbols consist of four symbols which are processes, data flows, data stores and external entities. The standard set of symbols that will be used in this paper is devised by Gane and Sarson symbols in [1]. In data flow diagram, the highest-level view of the system is known as context diagram. The next level of data flow diagram is called the level 0 data flow diagram which represents a system's major processes, data flows and data stores at a high level of detail. Every process in the level n-1 data flow diagram would be decomposed into its lower-level data flow diagram which is level n data flow diagram. The key principle in data flow diagram is to ensure balancing which means that the data flow diagram at one level is accurately represented in the next level data flow diagram when developing a project. The ideal level of decomposition is to

decompose the system until system analysts and users can provide a detailed description of the process whereby the process descriptions is not more than one page. The final set of data flow diagrams is validated for ensuring quality. In general, there are two types of problems that can occur in data flow diagrams which are syntax errors and semantics errors. Semantics errors are more complicated than syntax errors due to a set of rules that need to be followed in order to identify them. For example, every process has at least one input data flow and every process has at least one output data flow. Therefore, understanding the set of rules for data flow diagrams is important. Once the rules are understand, a system can be developed based on the rules so that the system can perform consistency check between context diagram to level 0 data flow diagram. The system is also able to perform grammatical errors checking within or across data flow diagrams in order to achieve consistency. This system therefore might increase the correctness and reliability of data flow diagrams.

III. RELATED WORKS

According to Lucas et al. [3], consistency problems have existed in Information System development since its beginning and are usually linked to the existence of multiple models or views which participate in the development process. Tao & Kung [4] state that a data flow diagram is visual and informal, hence, easy to learn and use. However, its informality makes it difficult to conduct formal verification of the consistency and completeness of a data flow diagram specification.

Dixit et al. [2], on the other hand, defined data flow diagram consistency is the extent to which information contained on one level of a set of nested data flow diagram is also included on other levels. According to Tao & Kong [4], the child data flow diagram that results from decomposition is consistent with the precedence relation for the parent process and does not introduce additional precedence relationships between the input and output flows of the parent process. Recently, many systems have been developed to provide automatic support for data flow diagrams specifications. However, all of these systems are lack the ability to manipulate the semantics of data flow diagram specification by Tong & Tang [5].

Various researches also stated that no formal language has been currently used for semantic specification of data flow diagram ([5], [6], and [7]). Tao & Kung [4], on the other hand, point out that there are few development environments or CASE tools provide automated verification facilities that can detect inconsistency and incompleteness in a data flow diagram specification. Dixit et al. [2] therefore describe that the concept of data flow diagram consistency is refers to whether or not the depiction of the system shown at one level of a nested set of data flow diagram is compatible with the depictions of the system shown at other levels. Consistency in process decomposition, on the other hand, means that the precedence relation is faithfully inherited by the child data flow diagram [4]. Ahmed Jilani et al. [7], on the other hand, state that notations used in the data flow diagram are usually graphical and different tools and practitioners interpret their notations differently. Therefore, a well defined semantics or data flow diagram formalism could help to reduce inconsistencies and confusion. Dixit et al [2] also state that a consistency check facility with a CASE tool will be helpful for the practitioners.

Our research therefore focuses on consistency check between data flow diagrams and develops a tool for consistency check between the data flow diagrams.

IV. SYNTAX AND SEMANTIC RULES OF DFD

According to Jeffrey et al. [8], data flow diagrams (DFDs) are graphically illustrate movement of data between external entities and the processes and data stores within a system. According to Donald [9], data flow diagrams are a tool that can reveal relationships among and between the various components in a program or system. Tao & Kung [4], on the other hand, describe the data flow diagram technique is effective for expressing functional requirements for large complex systems. There are four symbols in the data flow diagram which are processes, data flows, data stores and external entities (source/sink), each of which is represented by a different graphic symbol ([1] and [2]). In general, there are two commonly used styles of symbols in data flow diagram. For our research, we will use Gane and Sarson symbols as described in [1].

Definition 1: A Data Flow Diagram consists of:

- Process
- Data Flow
- Data Store
- External Entity

where

- A process is an activity or a function that is performed for some specific business reason.
- A data flow is a single piece of data or a logical collection of several pieces of information.
- A data store is a collection of data that is stored in some way.
- An external entity is a person, organization, or system that is external to the system but interact with it.

The highest-level of data flow diagram known as the context diagram. According to Jeffrey et al. [8], a context diagram is a data flow diagram of the 10 scope of an organizational system that shows the system boundaries, external entities that interact with the system and the major information flows between the entities and the system. Dennis et al. [1] state that the context diagram shows the overall business process as just one process and shows the data flows to and from external entities. Data stores are not usually included on the context diagram. The context diagram therefore is decomposed into the lower-level diagram which is level 0 data flow diagram. In fact, each process on the level 0 data flow diagram can be decomposed into more explicit data flow diagram, called level 1 diagram and can be further decomposed into next lower-level diagram when it is needed.

In general, there are two fundamentally different types of problems that can occur in data flow diagrams which are syntax errors and semantics errors. Tao & Kung [4] defined the syntax of the data flow diagram is how components are interconnected through data flows and what components constitute the subsystem being modeled. The semantics of the data flow diagram, on the other hand, is how data flows are interrelated in terms of data transformations. According to Dennis et al. [1], syntax errors are easier to find and fix than are semantics errors because there are clear rules that can be used to identify them. There is a set of rules that must be followed by analysts when drawing data flow diagrams in order to evaluate data flow diagrams for correctness.

Definition 2: Rules of data flow diagrams:

- A unique name (verb phase), a number and a description for a process
- At least one input data flow and at least one output data flow for a process
- Output data flows usually have different names than input data flows for a process
- A unique name that is a noun and a description for a data flow
- Every data flow connects to at least one process.
- Data flows only in one direction
- A minimum number of data flow line cross.
- A unique name that is a noun and a description for data store
- At least one input data flow on some page of the DFD.
- At least one output data flow on some page of the DFD.
- A unique name that is a noun and a description for external entity
- At least one input or output data flow for external entity

Definition 3: Consistency:

- Every set of data flow diagrams must have one context diagram.

Definition 4: Consistency Viewpoint:

- There is a consistency viewpoint for the entire set of DFDs.

Definition 5: Decomposition:

- Every process is wholly and completely described by the processes on its children DFDs.

Definition 6: Balancing:

- Every data flow, data store and external entity on a higher level DFD is shown on the lower-level DFD that decomposes it.

Definition 7: Data Store:

- For every data store, data cannot move directly from one data store to another data store. Data must be moved by a process.
- Data cannot move directly from a source to a sink. It can only be moved by a process.

Syntax rules are used to verify syntax errors within the data flow diagram. There is few syntax rules that have been defined to be used in a proposed system. The syntax rules are defined in Definition 8.

Definition 8: Syntax rules of data flow diagram:

- At least one input data flow for a process
- At least one output data flow for a process
- Process from external entity cannot move directly to another external entity
- At least one input data flow for a data store
- At least one output data flow for a data store
- Data from one data store cannot move directly to another data store.

Based on Definition 8, six syntax rules are used in order to verify the correctness of the context diagram and level 0 data flow diagram. However, the syntax rules of data store only applied in level 0 data flow diagram.

Semantics rules are used to verify semantics errors from context diagram to level 0 data flow diagram. Same as syntax rules, there are four semantics rules that have been defined to be used in a proposed system. The semantics rules are defined in Definition 9.

Definition 9: Semantic rules of data flow diagram:

- The total number and name of external entities in context diagram are the same as in level 0 data flow diagram.
- The total number and name of data flows between process and external entity in context diagram are same as level 0 data flow diagram.
- The total number and name of external entities in level 0 DFD are same as context diagram.
- The total number and name of data flows between process and external entity are the same as in context diagram.

The semantics rules defined in Definition 9 are used to perform consistency check from context diagram to level 0 data flow diagram. The rules are embedded inside the tool in order to perform an automatic consistency check between the data flow diagrams.

V. THE TOOL

The tool is developed based on the set of rules imposed by data flow diagrams as described from Definitions 1 until 9. A graphical layout is used in order to use the tool as an editor for drawing the diagrams and as a checker as well to check the correctness of the diagrams. Figure 1 shows the main interface of the tool.

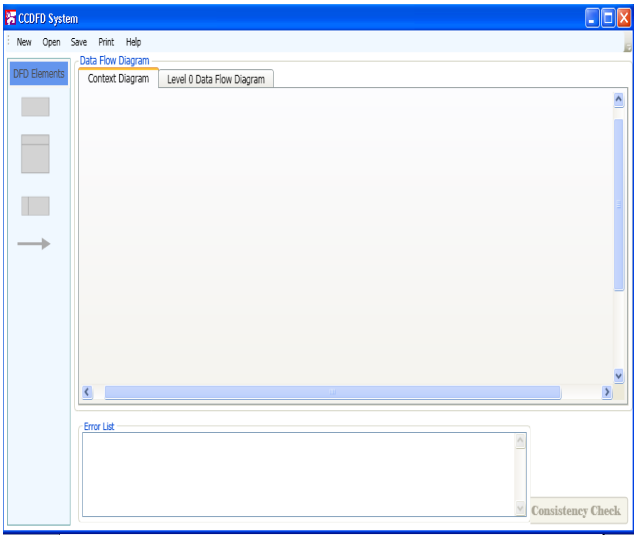

Fig. 1 The main interface of the tool

From Figure 1, the main interface provides a platform that allows the user to input both diagrams by using the data flow diagram elements provided. The main interface includes four main parts where the top of the interface is the menu bar consists of five menu functions, the toolbar of the data flow diagram elements is in the left-side of the interface, the bottom-right is an error list text box and a "Consistency Check" button. The rest of interface is the drawing panel for user to draw the particular diagram. The five functions in menu bar are open a new file, open a saved file, save the data flow diagrams, print the data flow diagrams and open a help menu. In the toolbar, there are four data flow diagram elements which are process, external entity, data flow and data store. User is allowed to drag and drop the data flow diagram elements on the drawing panel. "Consistency Check" button, on the other hand, is used to perform the consistency check after both diagrams are created. Therefore, the tool serves two purposes. The first purpose is as an editor for drawing the context diagram and level 0 data flow diagram and the second purpose is as a checker for checking the consistency between context diagram and level 0 data flow diagram.

In this paper, we give one simple example of an academic information system and use the tool to represent the context diagram and its level 0 of data flow diagram. Figure 2 shows the example of the context diagram for a lecturer who is going to use the Academic Information System. A lecturer can send his or her academic information to the system and can get a list of academicians from the system.

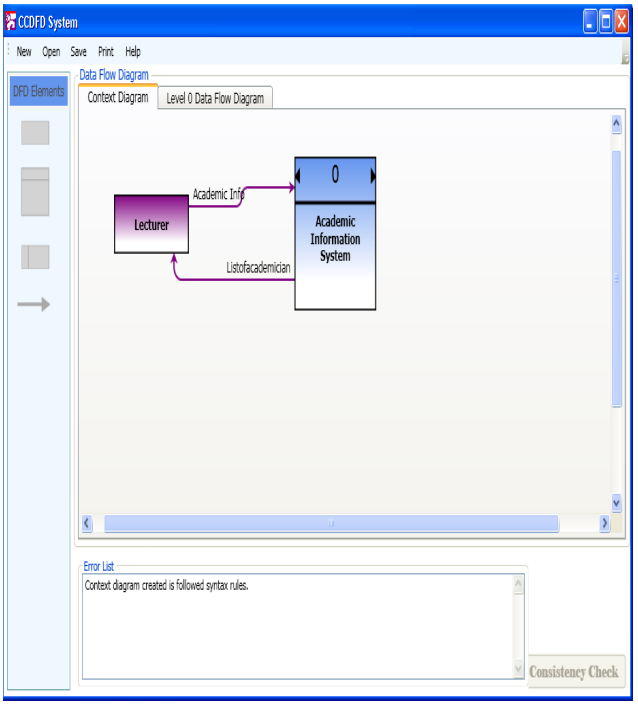

Fig. 2 Example of Context Diagram Drawn

From context diagram, a level 0 data flow diagram can be drawn. Figure 3 shows the example.

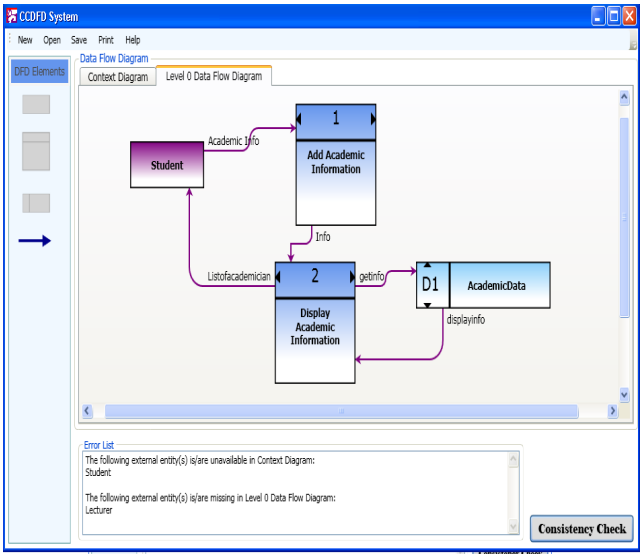

Fig. 3 Example of Level 0 Data Flow Diagram

The tool will verify the syntax errors for all the data flow diagram elements used. When there is any syntax errors exist in the data flow diagram elements, system will display an error message to user. Figure 3 shows an existence of syntax error from the data flow diagram. Since the entity from context diagram is Lecturer, the tool will inform the inconsistency between context diagram and data flow diagram. The user can then use the editor of the tool to correct

the syntax error. If the entity is correct, the tool will validate the consistency check as shown in Figure 4.

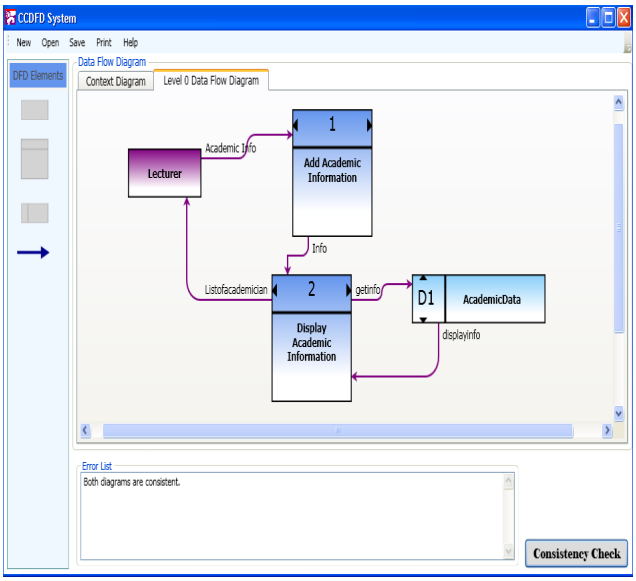

Fig. 4 The Consistency Check

Once the diagrams are drawn, they can be saved to a new or existing folder. The user can open the folder again for viewing or editing of the diagrams. The user can also print the diagrams. The Help menu can be used for getting more information regarding the tool.

VI. Conclusion

This paper has discussed a set of syntax and semantics rules of data flow diagrams. The rules are then used to automate the process of checking the consistency between the context diagram and level 0 data flow diagrams. The automatic checking of consistency overcomes the time-consuming process of manually checking the correctness of the diagrams.

For future enhancement of the tool, we would like to add the next level of consistency check. That is, the tool would be able to check the consistency between level 0 to level 1 of the data flow diagrams.

Acknowledgment

This research is supported by the Science Fund under Ministry of Science, Technology and Innovation (MOSTI), Malaysia.

References

[1] Dennis, A., Wixom, B.H. and Roth, R.M., *Systems Analysis and Design*. 3rd ed. Hoboken: John Wiley & Sons, Inc., 2006.

[2] Dixit, J. B. & Kumar, R., *Structured System Analysis and Design*. Paperback ed. New Delhi, India: Laxmi Publisher, 2008.

[3] Lucas, F.J., Molina, F. and Toval, A., A Systematic Review of UML Model Consistency Management. *Information and Software Technology*, 51(12), pp. 1 – 15, 2009.

[4] Tao, Y.L. and Kung, C.H., Formal Definition and Verification of Data Flow Diagrams. *Journal of Systems and Software*, 16(1), pp. 29-36, 1991.

[5] Tong, L. and Tang, C.S., Semantic Specification and Verification of Data Flow Diagrams. *Journal of Computer Science and Technology*, 6(1), pp. 21-31, 1991.

[6] Leavens, G.T., Wahls, T. and Bakar, A.L., Formal Semantics for SA Style Data Flow Diagram Specification Languages. *Proceedings of the 1999 ACM Symposium on Applied Computing.* Oregon, US: IEEE Computer Society. pp. 526–532, 1999.

[7] Ahmed Jilani, A. A., Nadeem, A., Kim, T. H. & Cho, E. S., Formal Representations of the Data Flow Diagram: A Survey. *Proc. of the 2008 Advanced Software Engineering and Its Applications.* Washington, USA: IEEE Computer Society. pp. 153-158, 2008.

[8] Jeffrey, A. H., George, J.F. and Valacich, J.S., *Modern Systems Analysis and Design*. 3rd ed. US: Prentice-Hall, 2002.

[9] Donald, S. Le Vie, Jr., Understanding Data Flow Diagram. *Proceedings of the 47th annual conference on Society for Technical Communication*. Texas: Integrated Concepts, Inc. 2000.