Original Article

Healthc Inform Res. 2011 December;17(4):214-223.
http://dx.doi.org/10.4258/hir.2011.17.4.214
pISSN 2093-3681 • eISSN 2093-369X

HIR
Healthcare Informatics Research

The Development of a Graphical User Interface Engine for the Convenient Use of the HL7 Version 2.x Interface Engine

Hwa Sun Kim, PhD[1], **Hune Cho, PhD**[2], **In Keun Lee, PhD**[2]
[1]Department of Medical Information Technology, Daegu Haany University; [2]Department of Medical Informatics, Kyungpook National University School of Medicine, Daegu, Korea

Objectives: The Health Level Seven Interface Engine (HL7 IE), developed by Kyungpook National University, has been employed in health information systems, however users without a background in programming have reported difficulties in using it. Therefore, we developed a graphical user interface (GUI) engine to make the use of the HL7 IE more convenient. **Methods:** The GUI engine was directly connected with the HL7 IE to handle the HL7 version 2.x messages. Furthermore, the information exchange rules (called the mapping data), represented by a conceptual graph in the GUI engine, were transformed into program objects that were made available to the HL7 IE; the mapping data were stored as binary files for reuse. The usefulness of the GUI engine was examined through information exchange tests between an HL7 version 2.x message and a health information database system. **Results:** Users could easily create HL7 version 2.x messages by creating a conceptual graph through the GUI engine without requiring assistance from programmers. In addition, time could be saved when creating new information exchange rules by reusing the stored mapping data. **Conclusions:** The GUI engine was not able to incorporate information types (e.g., extensible markup language, XML) other than the HL7 version 2.x messages and the database, because it was designed exclusively for the HL7 IE protocol. However, in future work, by including additional parsers to manage XML-based information such as Continuity of Care Documents (CCD) and Continuity of Care Records (CCR), we plan to ensure that the GUI engine will be more widely accessible for the health field.

Keywords: Health Level Seven, Medical Informatics, Computer Graphics, Software Design

Submitted: August 31, 2011
Revised: September 16, 2011
Accepted: October 11, 2011

Corresponding Author
In Keun Lee, PhD
Department of Medical Informatics, Kyungpook National University, Samdeok-dong 2-ga, Jung-gu, Daegu 700-721, Korea. Tel: +82-53-420-4896, Fax: +82-53-423-1242, E-mail: inkeunlee@gmail.com

I. Introduction

Efforts to computerize health information have led to the development of various health information systems that are used in many facilities including hospitals [1-5]. However, the information structures (e.g., database design and data structure) used by multiple health information systems are not necessarily exchangeable. For guaranteeing interoperability between health information systems, various standards for interoperability have been developed, such as Health Level Seven (HL7) [6], Digital Imaging and Communications in Medicine (DICOM) [7], Clinical Document Architecture (CDA) [8], Continuity of Care Document (CCD) [9], and Continuity of Care Record (CCR) [10]. Despite that the structure of health information is different, health information systems can exchange information by transforming the information according to common standards [5,11].

HL7 is the leading international standard for exchanging health information, and many facilities are using HL7 in their health information systems through HL7 interface engines [12-16] developed by various research facilities and companies. While HL7 interface engines are useful for parsing and creation of HL7 messages, computer programmers are required for practical operation of the HL7 interface engine and for connecting it with health information systems. However, it is unreasonable to request assistance from a programmer whenever it is necessary to create or modify HL7 messages. The connection module between an HL7 interface engine and a health information system may be modified by a small alteration of a rule or system in the facility and a new connection module would accordingly have to be developed according to the healthcare provider's needs. However, the speed of developing a connection module may be subject to the availability of a computer programmer. If there is a lack of understanding between a healthcare provider and a computer programmer, more time is needed.

Some commercial solutions have been developed to address these problems, including Symphonia from Orion [14], Iguana from Interfaceware [15], and Mapforce from Altova [16]. These solutions minimize programmer intervention when healthcare providers try to process HL7 messages by providing a graphical user interface (GUI) that is easy to use for parsing and creation of HL7 messages. However, commercial GUI engines connected with commercial HL7 interface engines prevent users from utilizing another parser, because these engines use self-developed parsers for HL7 and extensible markup language (XML). This makes it very difficult to connect a specific HL7 parser with a commercial GUI engine without analyzing the commercial GUI engine or negotiating with companies supplying the software. Thus, even though some non-commercial HL7 interface engines such as HAPI [12] have been released as open source technology with powerful parsers for HL7 messages, HL7 interface engines without GUIs are not suitable for users without programming knowledge.

Unlike commercial programs, the HL7 version 2.x Message Interface Engine developed in Kyungpook National University [13] (we refer to it as “HL7 IE” to distinguish it from general HL7 Interface Engines) did not provide a GUI. Therefore, it is desirable to realize convenient use of HL7 IE by connecting a GUI engine that is accessible to non-programmer users. However, it should be noted that it is difficult to develop a GUI engine that is compatible with all parsers developed by other parties.

The purpose of this study was to develop a GUI engine that provides a convenient interface between users and HL7 IE (we hereafter refer to the HL7 IE combined with the GUI engine as “GUI-based HL7 IE”). The usefulness of the GUI engine was confirmed through an evaluation that demonstrated the simplicity of producing a health information exchange model based on HL7 messages.

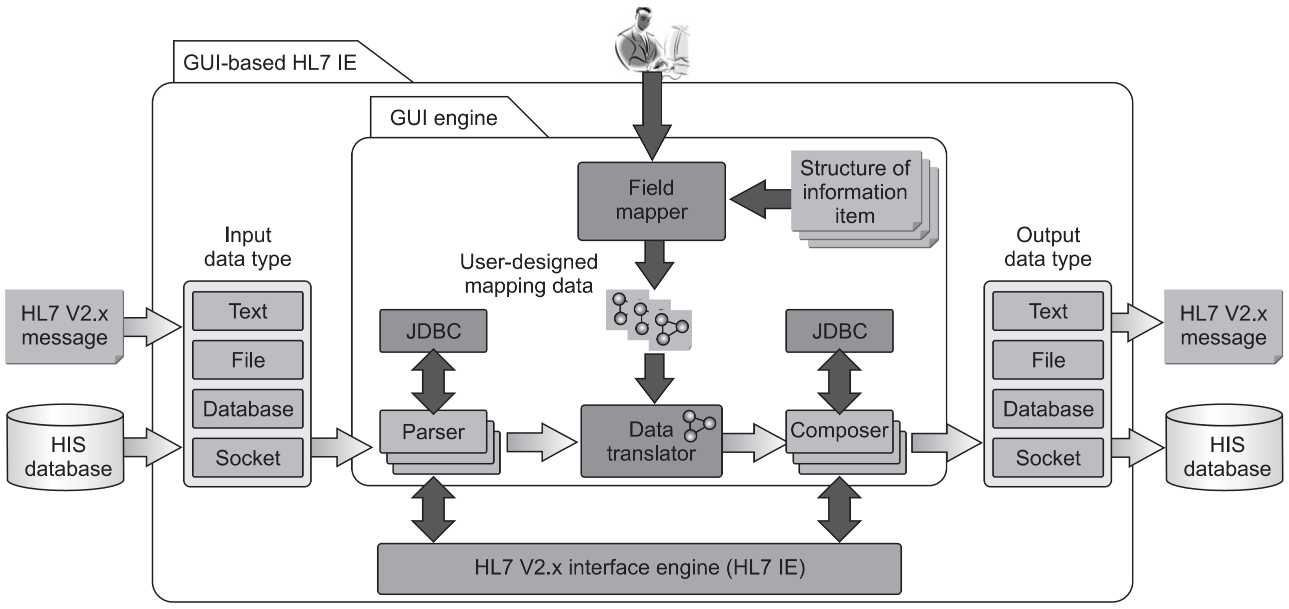


Figure 1. Graphical user interface (GUI)-based HL7 version 2.x interface engine architecture. HL7 IE: Health Level Seven Interface Engine, HIS: Health Information System, JDBC: Java database connectivity.

II. Methods

1. Development of the GUI Engine

1) Architecture of the GUI engine

The primary role of the GUI engine is to provide an easy way to handle HL7 messages based on HL7 IE and health information database systems. Therefore, as shown in Figure 1, the GUI engine is connected with HL7 IE and the database of a health information system to handle HL7 messages and database information, respectively. By using the GUI engine, the user requires no understanding of the structure or program code of HL7 IE. As such, the user only needs to understand the structure of each information item (e.g., the HL7 message structure or the table design of the relational database) when making rules for health information exchange between fields, which is accomplished using a graph via the GUI of the "Field Mapper." Here, we refer to columns of the table in a relational database or the primitive data type of the HL7 message structure as "fields," and the rules of exchange between fields as "mapping data." The mapping data are transformed into program objects using the "Field Mapper" and the objects are processed by a "Data Translator," which translates the input data into the required output data. "Parsers" and "Composers" are used to parse and compose various types of data, such as Text, File, Network (Socket), and Database, for each information item. In the case of a HL7 message, in particular, HL7 IE performs parsing and creation of the HL7 message, and Java database connectivity (JDBC) is used to connect with database. The mapping data can also be stored and reused at a different location/time from that where the GUI-based HL7 IE is installed.

2) Design of the GUI

We designed a GUI to represent information items as graphs and trees that allow the creation of mapping data linking two fields using a mouse interface. Mapping data are created by designing graphs with lines and information items. They should then be stored as program objects that computer software can process. Thus, we designed the "Field Mapper" by dividing objects into the following two classes, as shown in Figure 2: objects for graphic representation and objects for mapping data. Figure 2 shows that "NodeBean" is used to represent the field and "TreeBean" is the structure of a tree containing "NodeBean" objects. "InstanceBean" indicates an object in the "TreeBean" and "ItemBean," while the object "ItemBean" indicates the type of information item. "LinkageBean" is used to represent a link between two objects in "NodeBean." All objects are part of an object called "MappingDataBean" that facilitates easy handling of mapping results. The "MappingDataBean" object is used to manage the task of mapping data creation. To support the graphical creation of mapping data using the GUI, all information structures, linkage information, and the entire mapping task can be represented graphically by using the "InstancePane," "LinkagePane," and "DesignPane," respectively. The "InstancePane" is used to draw the structure of information items including fields and the "LinkagePane" is used to draw the lines linking fields. Individual designs are gathered in the "DesignPane." All classes are implemented using a "Serializable" Java interface class allowing objects to be stored as a binary file.

3) Representation of information items

The GUI engine deals with the following four types of information items: items related to input and output (I/O) that are used to designate the type of I/O; items to represent the

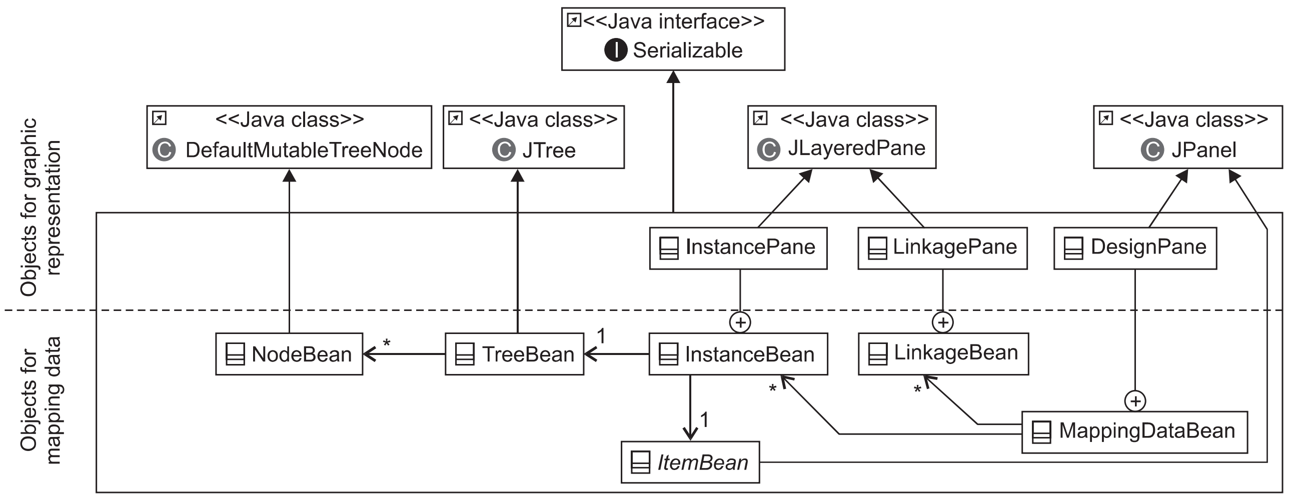


Figure 2. Class diagram of graphical user interface (GUI) engine showing the field mapper and its graphical representation.

HL7 message types; items that represent database tables connected to the GUI engine; and items related to functions for transforming and creating data during field mapping. In Figure 3, the important role of "ItemBean" is to provide operations (Java methods) such as "encode" and "decode" that are used to parse and compose health information. The details of operations are defined for each item class inherited from the "ItemBean" class by overriding the operations. For example, the "decode" operation of item classes in the "ItemMaker_HL7_2_5" group performs parsing of an inputted HL7 message, while the "encode" operation of item classes performs HL7 message creation using pre-defined data for "NodeBean" objects. The operations of item classes in the "ItemMaker_Database" group execute the defined SQL queries, i.e., the "decode" operation of the item classes conducts the search of a specific database, while the "encode" operation performs tasks related to the database, such as inserting, updating, and removing records. Groups and items can be represented graphically via the "ItemPanel" and new item groups or items can be added to the GUI engine using the "ItemPanel."

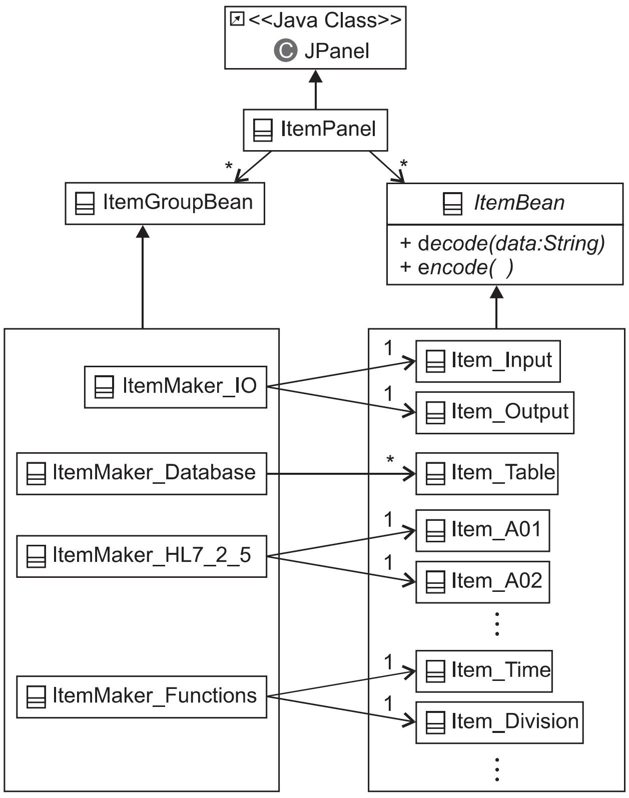


Figure 3. Class diagram of information items.

4) Connection with the HL7 Version 2.x Interface Engine

Figure 4 shows the partial class structure of the HL7 IE for the ADT_A01 message represented in Figure 5. Although the properties of the MSH segment are "required" and "unrepeatable" in the HL7 standard, HL7 IE designed the "ADT_A01" object to have plural objects in "MSH," providing a consistent program structure. However, cardinality can be limited by the properties of the MSH segment. "ADT_A01_INSURANCE" class indicates a group containing segments defined as "[{IN1 [IN2] [{IN3}] [{ROL}]}]." It is convenient to manage these segments as a group, because the properties of all segments in the group are "optional" and "repeatable." Here, the symbols "[]" and "{}" indicate the "optional" property and the "repeatable" property, respectively.

The structure of classes related to HL7 messages in the GUI engine also replicates that found in HL7 IE, as shown in Figure 6. Classes in the GUI engine are directly connected with those in HL7 IE, allowing effective parsing and creation of HL7 messages. In the case of HL7 IE, "ADT_A01" class was classified in the "Message" group, as shown in Figure 4. However, in the case of the GUI engine, as shown in Figures 3 and 6, the "ADT_A01" class was classified in the "ItemBean" group, because various information items such as "Database", "I/O", "Functions", and "HL7 2.x (messages)" are classified in the same manner in order to maintain the consistency and

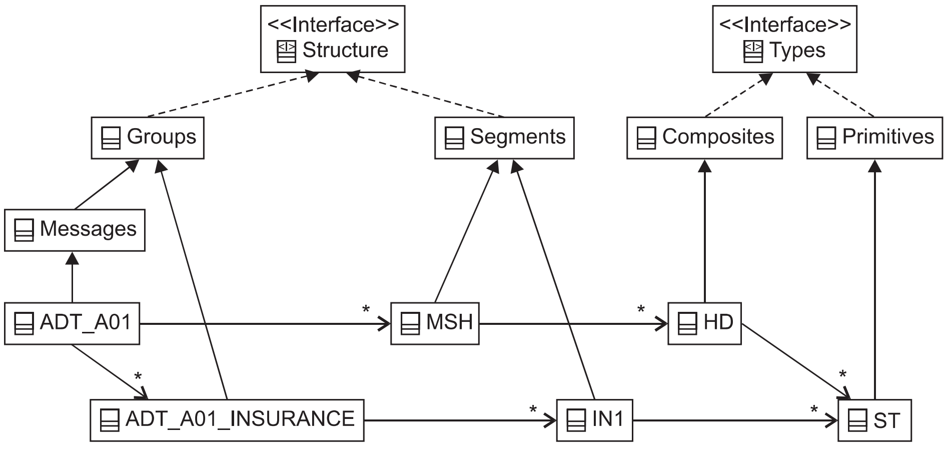
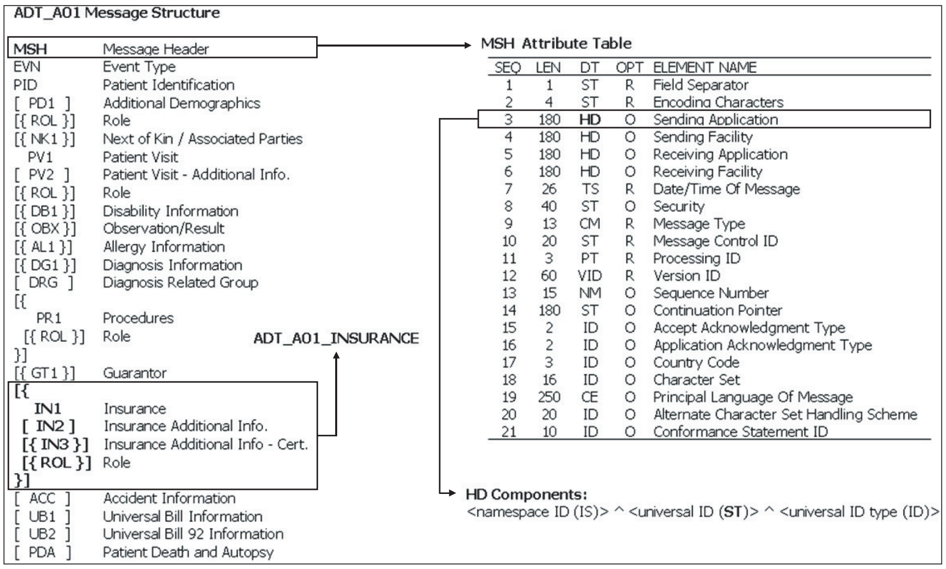


Figure 4. Partial class diagram of the Health Level Seven (HL7) interface engine.

the extensibility of the program.

2. Examination of the GUI Engine

We simulated information exchange between health information systems (SeniCare, as designed by Kim et al. [1]) using the GUI engine, as shown in Figure 7. In the simulation, a GUI-based HL7 IE transmits a message through the network after creating an HL7 message based on the SeniCare database on the transmitter side. Another GUI-based HL7 IE updates the SeniCare database on the receiver side after parsing the transmitted HL7 message, and it transmits a response message to the transmitter side of the SeniCare system. Although the health information systems (SeniCare) used in the simulation were identical, we regarded them as different systems because they were operated independently of each other. We assumed the following scenario when producing an HL7 message in the simulation: "There was a new addition to the SeniCare new allergy information relating to 'Penicillin' for an outpatient named 'Alice Smith' born in 1953, and we need to send the information to the Kyung-



ADT_A01 Message Structure

MSH	Message Header
EVN	Event Type
PID	Patient Identification
[PD1]	Additional Demographics
[{ ROL }]	Role
[{ NK1 }]	Next of Kin / Associated Parties
PV1	Patient Visit
[PV2]	Patient Visit - Additional Info.
[{ ROL }]	Role
[{ DB1 }]	Disability Information
[{ OBX }]	Observation/Result
[{ AL1 }]	Allergy Information
[{ DG1 }]	Diagnosis Information
[DRG]	Diagnosis Related Group
[{	
PR1	Procedures
[{ ROL }]	Role
}]	
[{ GT1 }]	Guarantor
[{	
IN1	Insurance
[IN2]	Insurance Additional Info.
[{ IN3 }]	Insurance Additional Info - Cert.
[{ ROL }]	Role
}]	
[ACC]	Accident Information
[UB1]	Universal Bill Information
[UB2]	Universal Bill 92 Information
[PDA]	Patient Death and Autopsy

ADT_A01_INSURANCE

MSH Attribute Table

SEQ	LEN	DT	OPT	ELEMENT NAME
1	1	ST	R	Field Separator
2	4	ST	R	Encoding Characters
3	180	HD	O	Sending Application
4	180	HD	O	Sending Facility
5	180	HD	O	Receiving Application
6	180	HD	O	Receiving Facility
7	26	TS	R	Date/Time Of Message
8	40	ST	O	Security
9	13	CM	R	Message Type
10	20	ST	R	Message Control ID
11	3	PT	R	Processing ID
12	60	VID	R	Version ID
13	15	NM	O	Sequence Number
14	180	ST	O	Continuation Pointer
15	2	ID	O	Accept Acknowledgment Type
16	2	ID	O	Application Acknowledgment Type
17	3	ID	O	Country Code
18	16	ID	O	Character Set
19	250	CE	O	Principal Language Of Message
20	20	ID	O	Alternate Character Set Handling Scheme
21	10	ID	O	Conformance Statement ID

HD Components:
<namespace ID (IS)> ^ <universal ID (**ST**)> ^ <universal ID type (ID)>

Figure 5. Example of Health Level Seven (HL7) message structure.

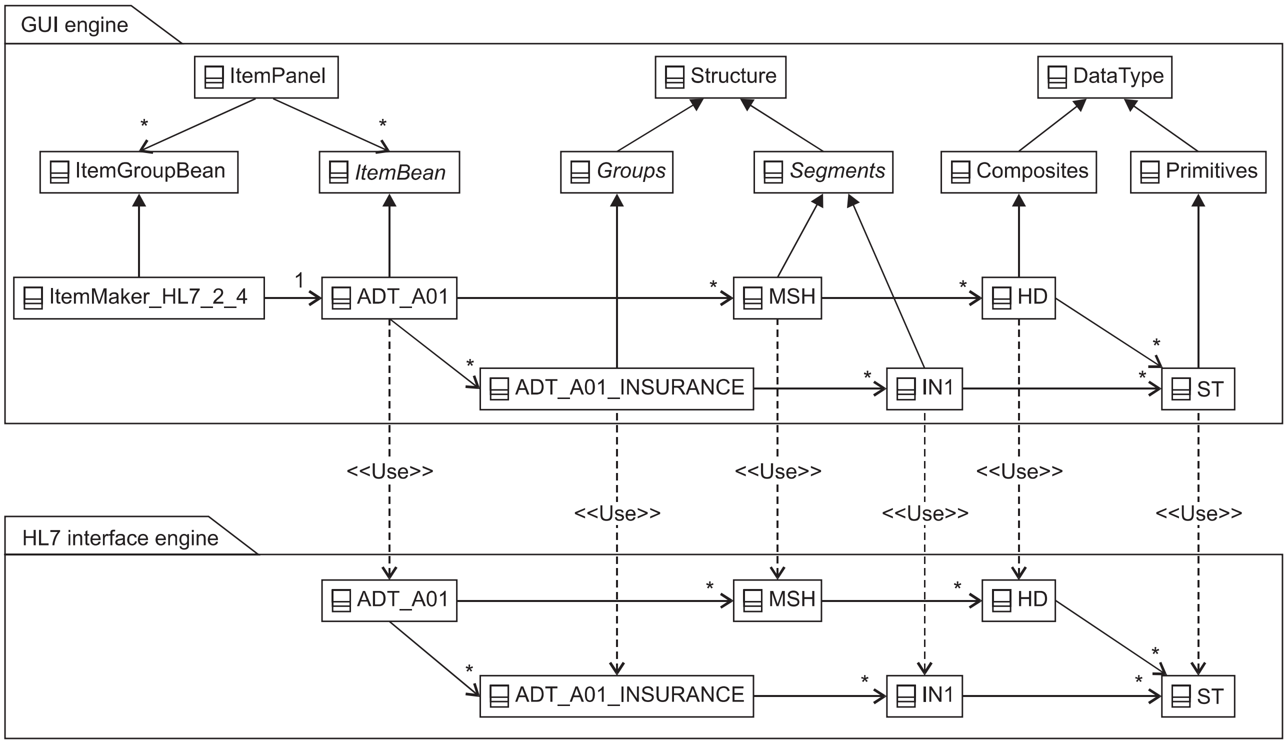


Figure 6. Connection of the graphical user interface (GUI) engine with the Health Level Seven (HL7) interface engine.

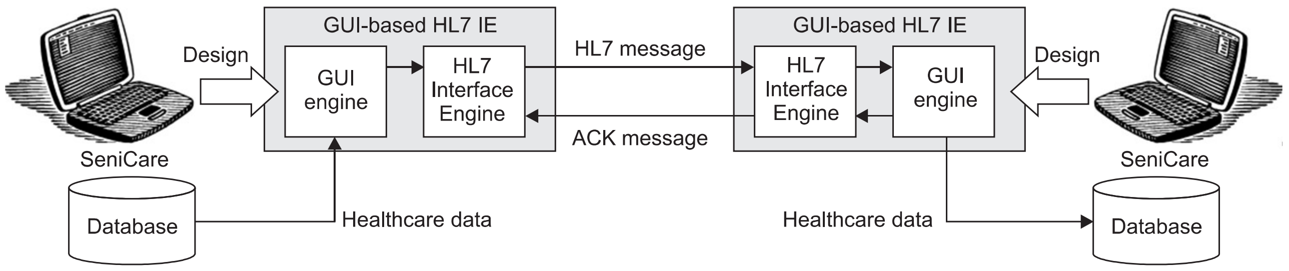
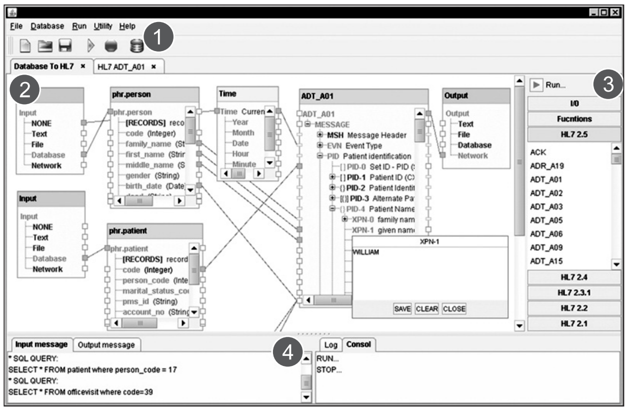


Figure 7. Model of health information exchange using the graphical user interface (GUI) engines. HL7: Health Level Seven, IE: interface engine.



```
* SQL QUERY:
SELECT * FROM patient where person_code = 17
* SQL QUERY:
SELECT * FROM officevisit where code=39
```

```
RUN...
STOP...
```

Figure 8. Snapshot of the graphical user interface (GUI) engine.

pook National University Hospital (KNUH) facility." To represent this sequence, the GUI engine on the transmitter side creates an ADT_A60 message representing "update adverse reaction information" and it requests an update of the SeniCare database in KNUH by transmitting a message to the receiver side of SeniCare.

Furthermore, we conducted some tests creating four HL7 messages (ADT_A60, ADT_A01, ADT_A49, RQI_I01) by creating mapping data using the GUI engine and by developing Java application programs based on the HL7 IE with the help of two participants, a computer programmer and a non-programmer. The non-programmer who created mapping data had considerable experience in use of the GUI engine, and the programmer who developed the application programs was familiar with HL7 IE. And we confirmed the usefulness of the GUI engine by comparing records measuring the time spent creating the four HL7 messages in both ways.

III. Results

1. Development of the GUI Engine

Figure 8 shows a snapshot of the developed GUI engine. The GUI engine consists of four parts based on their function, as follows: 1) a control panel containing shortcut buttons to perform basic functions such as saving and loading mapping data, connecting to a database, and starting and finishing the exchange process; 2) a design panel to create mapping data via graphical inputs. Linkage information between fields is represented using lines, while the structure of information items is represented as a tree in this panel. Users can create linkage information with a mouse interface by linking two small squares (representing fields) on different large squares (representing information items). Moreover, users can confirm in each field of the tree more detailed information such as parsed data of HL7 messages and records of a table in the connected database. For example, a patient's given name, "WILLIAM," is shown below the field "XPN-1" belonging to the "ADT_A01" item tree in Figure 8; 3) an information item panel. For example, if the input data of the HL7 message is ADT_A01, the ADT_A01 item can be added to panel ② by choosing it in panel ③; and ④ an information panel showing input and output messages, the progress of the exchange process, and the status of the GUI engine. The GUI-based HL7 IE was designed as a standalone system, as shown in Figure 8, but it could also be used as a module in a system developed in Java programming language.

2. Examination of the GUI Engine

1) Creation and treatment of HL7 messages using GUI engine

We used the GUI-based HL7 IE to handle HL7 messages and to update the SeniCare database according to the scenario. The details are as follows.

Figure 9A shows that HL7 messages are created by the "HL7 Encoder" based on data in the SeniCare database and user-inserted data. When the format of the data from the database is different from that of the HL7 message, the data are transformed to the correct format for the HL7 message by "Function" items provided in the GUI engine. The created HL7 message is transmitted through the TCP/IP network. Figure 9B shows an example of mapping data that implement the model in Figure 9A using the "Field Mapper" of the GUI

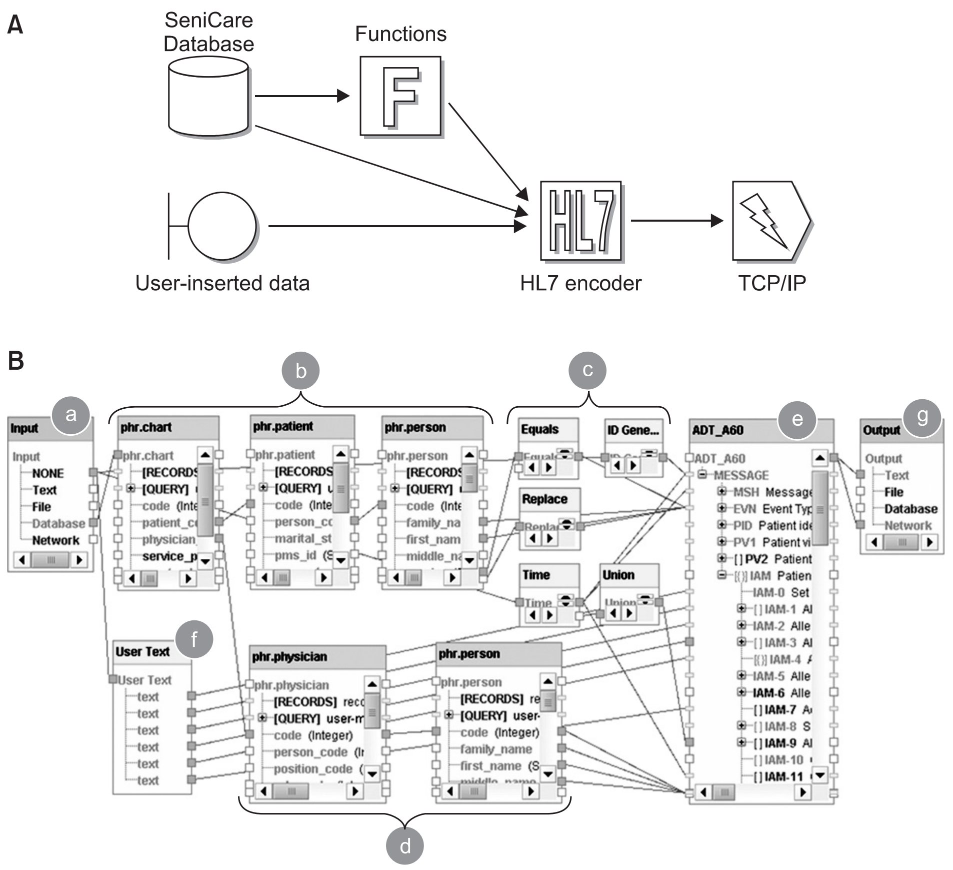


Figure 9. (A) Conceptual model of transmitter-side activities. (B) Exchange model designed using the graphical user interface (GUI) engine to create an ADT_A60 HL7 message. HL7: Health Level Seven.

engine. Figure 9B shows the following: ⓐ input data type to connect with the SeniCare database; ⓑ and ⓓ show groups for extracting information on patients and physicians to be used to create ADT_A60 message; ⓒ shows the use of functions provided by the GUI engine for transforming the data format of the database to the HL7 message format. For example, the representation of the gender of a patient is “1” for a male and “0” for a female in the database, whereas “M” and “F” are used for male and female in HL7 messages. Therefore, the gender value should be transformed from “1” to “M” and from “0” to “F” so that the user can define the role of the function “Equals” to return the value “M” if the input data equals “1”; and otherwise “F.” Additional data absent from the database can be inserted directly using the function “User Text” indicated in ⓕ. Additional data can also be placed in ⓔ directly. However, it may be better to use ⓕ for more convenient reuse of the mapping data, because handling a complex HL7 message structure is difficult whenever mapping data are reused. These data are used as sources for creating ADT_A60 messages by linking with the ADT_A60 fields indicated in ⓔ. ⓖ shows the output data types, where the created HL7 message is transmitted through the network and printed as text. The created HL7 message for ADT_A60 shown as follows:

```
MSH|^~\&|SeniCare|MIPTH|SeniCare|KNUH|20110818121005||
 ADT^A60^ADT_A60|316900860228771072|P|2.4
EVN||20110818121005
PID||17|||Smith^Alice||19530101|M
PV1||O
IAM|1|DA|^Penicillin|SV|rush on back|A|||AL||20110818||201108
 18121005|||||1^Martinson^Frederic^T^^^MD
```

The HL7 message is transmitted through the network and parsed by the “HL7 Decoder,” as shown in Figure 10A. The parsing data are used to update the SeniCare database on the receiver side. A response message is created and transmitted to the transmitter side after updating the database. Figure 10B shows a detailed example of the conceptual design of Figure 10A. ⓗ shows that the GUI engine receives a message through the network. The role of ⓘ is to parse the transmitted HL7 message and to display the parsed data as a tree structure. The role of group ⓙ is to find the record in the database that needs to be updated based on the input information related to a patient named “Alice Smith”, by sequentially connecting the fields of the tables in the database. In ⓚ, the selected record is updated in ⓝ by integrating the transmitted information related to the allergy. Functions such as “Union” and “Equals” are also used to transform the input data to a specific symbol (e.g., “A” denotes “addition” and “D” denotes “deletion”), according to the “Allergy Action Code” in the structure of the IAM segment of the HL7 message. For example, if the “Allergy Action Code” is “A”, then the record in the “chartcontent” table of the database should be updated by integrating the transmitted allergy information with the previously stored allergy data in the database. The role of ⓜ is to create a response message informing the transmitter side of SeniCare that the receiver side of SeniCare received the message. The response message created by ⓜ is as follows:

```
MSH|^~\&|SeniCare|KNUH|SeniCare|MIPTH|20110818121006||
 ACK|316900860228771072|P|2.4
MSA|AA|316900860228771072
```

2) Evaluation of the Usefulness of the GUI engine

We created four mapping data using the GUI engine and developed four Java application programs to create the four HL7 messages. Table 1 represents the time recorded when the mapping data were created by a non-programmer and when the programs were developed by a programmer. The recorded time includes the verification time of the created

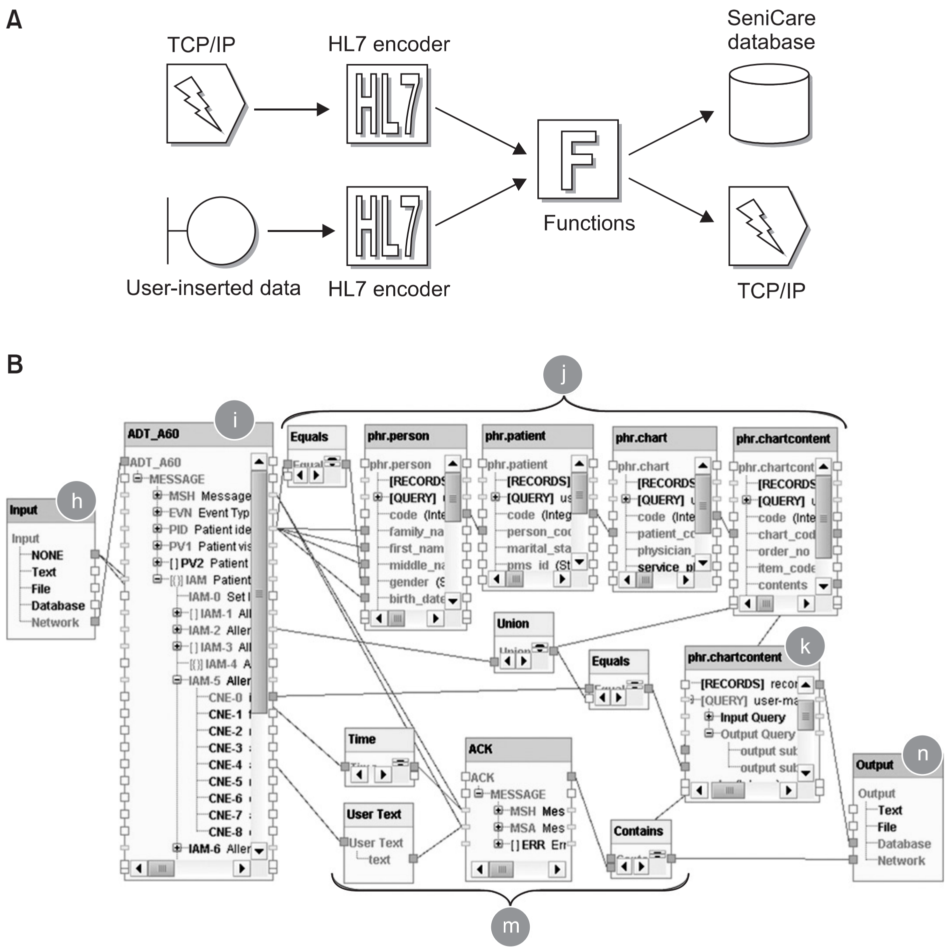


Figure 10. (A) Conceptual model of receiver-side activities. (B) An exchange model designed by the graphical user interface (GUI) engine to update the SeniCare database and create a response message.

Table 1. Time recorded upon creation of mapping data using the GUI engine and development of Java application program using HL7 IE

Order	HL7 message	[A] Creation of mapping data (min)	[B] Development of Java application program (min)	Difference [B]-[A] (min)	Ratio [A]/[B]
1	ADT_A60	24	86	62	3.58
2	ADT_A01	18	42	24	2.33
3	ADT_A41	15	28	13	1.86
4	RQI_I01	12	36	24	3
	Average	17.25	48	30.75	2.78

HL7: Health Level Seven, GUI: graphical user interface.

HL7 messages.

In terms of creation time of HL7 messages, as shown in Table 1, we found that the creation of HL7 messages by using the GUI engine was faster than that by developing Java application programs. Comparing the results of the first test and those of the other tests, the time was shortened as the tests continued, because the creation time could be saved by reusing the mapping data and the program that was made in the first test. However, we found that creating HL7 messages by developing java application programs involved almost 3 times longer time than that by using the GUI engine. From the test results, we could conclude that the GUI engine provides a convenient and useful way to create HL7 messages rather than direct use of HL7 IE.

IV. Discussion

We developed a GUI engine for convenient use of HL7 IE, developed in Kyungpook National University [13], as HL7 IE was released as a Java library that can only be used by Java programmers. The GUI engine that was developed in this study has the following characteristics: 1) various input and output data types are accessible, such as Text, File, Network, and Database; 2) HL7 version 2.x messages can be created by simply using the GUI without any help from programmers; 3) mapping data designed using the GUI engine can be stored and reused; and 4) some tools for data transformation are provided. These features are possible even when the user has no knowledge of the program used to run HL7 IE.

We examined the GUI engine through a simulation that demonstrated the simplicity of producing a health information exchange model. We also confirmed the usefulness of the GUI engine by comparing the time to create HL7 version 2.x messages by using the GUI engine and by developing Java application programs. From the tests, we found that even a non-programmer can conveniently control and manage the process of health information exchange with the GUI engine.

This work has several limitations. We considered only one exchange model in the simulation, but numerous exchange models that perform the same objective can be designed depending on the user's preferred concepts of health information systems. Moreover, we used two identical systems (i.e., SeniCare) for data exchange to test the GUI engine. We plan to carry out a simulation test to examine whether the GUI engine works seamlessly between two or more different systems.

We designed the GUI engine after benchmarking commercial programs. The quality of the commercial programs might currently be better than that of the GUI engine, but we will update the functions of the GUI engine and confirm its usability based on continuous user feedback. The HL7 message parser in HL7 IE covers HL7 versions 2.1 to 2.5, and hence the GUI engine can also deal with those versions of items. We will update the items after developing newer versions of the parsers, i.e., HL7 versions 2.6 and 2.7. In future research, we will include additional parsers to manage XML-based information such as CCR and CCD.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgements

This work was supported by the R&D Program of MKE/KEIT (KI10033576).

References

1. Kim HS, Cho H, Lee IK. Development of an electronic claim system based on an integrated electronic health record platform to guarantee interoperability. Healthc Inform Res 2011; 17: 101-110.
2. Walker JM, Bieber EJ, Richards F. Implementing an electronic health record system. London, UK: Springer; 2005.
3. Henricks WH. "Meaningful use" of electronic health records and its relevance to laboratories and pathologists. J Pathol Inform 2011; 2: 7.
4. Kim J, Jung H, Bates DW. History and trends of "personal health record" research in PubMed. Healthc Inform Res 2011; 17: 3-17.
5. Lopez DM, Blobel BG. A development framework for semantically interoperable health information systems. Int J Med Inform 2009; 78: 83-103.
6. Health Level Seven International. HL7 Standards - master grid [Internet]. Ann Arbor, MI: Health Level Seven International; c2011 [cited at 2011 Aug 16]. Available from: http://www.hl7.org/implement/standards/v2messages.cfm.
7. Digital Imaging and Communications in Medicine (DICOM). DICOM homepage [Internet]. [cited at 2011 Aug 16]. Available from: http://medical.nema.org.
8. Health Level Seven International. Clinical document architecture [Internet]. Ann Arbor, MI: Health Level Seven International; c2011 [cited at 2011 Aug 16]. Avail-

able from: http://www.hl7.org/implement/standards/cda.cfm.

9. Health Level Seven International. Continuity of care document [Internet]. Ann Arbor, MI: Health Level Seven International; c2011 [cited at 2011 Aug 16]. Available from: http://www.hl7.org/documentcenter/private/standards/cda/igs/HL7_CCD_final.zip.
10. American Society for Testing and Materials (ASTM). ASTM E2369 - 05e2 Standard specification for continuity of care record (CCR) [Internet]. West Conshohocken, PA: ASTM [cited at 2011 Aug 16]. Available from: http://www.astm.org/Standards/E2369.htm.
11. Li JS, Zhou TS, Chu J, Araki K, Yoshihara H. Design and development of an international clinical data exchange system: the international layer function of the Dolphin Project. J Am Med Inform Assoc 2011; 18: 683-689.
12. University Health Network. Hapi: the free, open, and best HL7 parser and library for Java [Internet]. University Health Network; c2001-2011 [cited at 2011 Aug 16]. Available from: http://hl7api.sourceforge.net.
13. Tran T. A development of integrated personal health record system to support continuity of care in Vietnam [dissertation]. Daegu: Kyungpook National University; 2008.
14. Orion Health. Symphonia messaging & mapping tools [Internet]. Santa Monica, CA: Orion Health; c2002-2011 [cited at 2011 Aug 16]. Available from: http://www.orion-health.com/products/symphonia.
15. Interfaceware. Products: Iguana [Internet]. Ontario, Canada: Interfaceware; c2011 [cited at 2011 Aug 16]. Available from: http://www.interfaceware.com/iguana.html.
16. Altova. MapForce: graphical data mapping, conversion, and integration tool [Internet]. Beverly, MA: Altova; c2011 [cited at 2011 Aug 16]. Available from: http://www.altova.com/mapforce.html.