Proc. 2nd International Conference on Web Information Systems Engineering (WISE2001), Kyoto, Japan, December 2001.

Integration of Multiple Dissemination-Based Information Sources Using Source Data Arrival Properties

Yousuke Watanabe[†] Hiroyuki Kitagawa[††] Yoshiharu Ishikawa[††]

[†] Graduate School of Systems and Information Engineering, University of Tsukuba
[††] Institute of Information Sciences and Electronics, University of Tsukuba
watanabe@kde.is.tsukuba.ac.jp
{kitagawa, ishikawa}@is.tsukuba.ac.jp

Abstract

The integration of heterogeneous information sources is an important data engineering research issue. Various types of information sources are available today. They include dissemination-based information sources, which actively and autonomously deliver information from servers to users. We are developing a mediator/wrapper-based information integration system in which we employ ECA rules to define new information delivery channels, integrating multiple existing dissemination-based information sources. ECA rules in this system are derived from integration requirement specifications based on relational algebra provided by users. Dissemination-based information sources usually have data arrival properties, such as an information delivery schedule. Using the data arrival properties of underlying information sources, the system can derive more appropriate ECA rules and check the consistency of requirements more accurately. This paper proposes an extended scheme to process information integration requirements using source data arrival properties of dissemination-based information sources.

1. Introduction

The recent development of Internet technology has enabled us to access a huge number of information sources, making the integration of heterogeneous information sources an important research issue [1, 2, 3, 4, 5]. Various types of information sources are available. They include dissemination-based information sources, which actively and autonomously deliver information from servers to users. Examples are push-based information delivery services [6, 7, 8], data broadcasting services [9, 10], and messages delivered from mailing-lists.

We are developing a mediator/wrapper-based information integration system, which can deliver information by integrating multiple existing dissemination-based information sources and other conventional information sources [11]. Users define new information delivery channels; integrated information is then sent through the channels according to user-specified schedules. We have employed ECA rules [12] to achieve active features. Triggered by events such as arrival of new data and time progress, the system actively performs actions related to data storage, integration, and delivery. Our study also provides a framework in which ECA rules are automatically derived from relational algebra-based user requirement specifications [13].

This paper proposes an extended scheme to process information integration requirements using data arrival properties of information sources. Many dissemination-based information sources, such as broadcasting stations, have their own information delivery schedules. In such cases, data arrival time is predetermined or predictable. If we use the data arrival properties of such underlying information sources, the system can derive more appropriate ECA rules and check the consistency of user requirements more accurately. We define two important properties of requirement specifications: consistency and satisfiability. We then show ECA rule derivation taking data arrival properties into account.

The remaining part of this paper is organized as follows: Section 2 shows a simple example of a new information delivery channel that integrates multiple existing dissemination-based information sources. Section 3 overviews our information integration system architecture. Section 4 explains the specification of source data arrival properties. Section 5 discusses consistency and satisfiability of requirement specifications. Section 6 explains ECA rule derivation, taking into account data arrival properties. Section 7 mentions related works. Finally, Section 8 presents

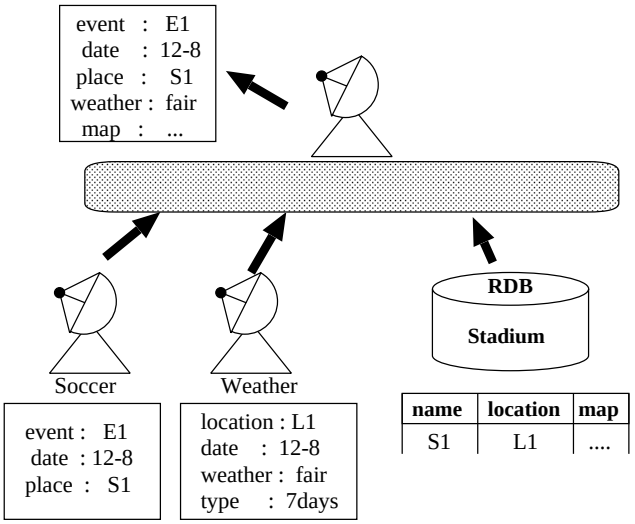

Figure 1. Integration Example

the conclusion.

2. Motivating Example

This section shows a simple integration scenario. It defines a new information delivery channel formed by integrating two dissemination-based information sources and a relational database (Figure 1). Here, we assume the existence of the following three information sources:

- **Soccer match information delivery channel (Soccer):**
  This source sends information on soccer matches to be held over the weekend. Each data unit from this channel contains such data items as date of the match, place, teams, and other useful information. Data units from this channel are assumed to arrive every Wednesday.

- **Weather information delivery channel (Weather):**
  This source sends weather forecast information. Data units from this channel are assumed to arrive at 6 o'clock every day. They are classified into '7 days' forecast data units and '4 weeks' forecast data units. Each '7 days' forecast data unit contains weather information for a specific region for one of the next seven days, and each '4 weeks' forecast data unit contains information for one of the next four weeks.

- **Stadium information database (Stadium):**
  This source contains information on stadiums. It is a relational database, with stadium name, location, and map attributes.

In the remaining part, we use the following scenario as a running example of information integration: "When soccer match data has arrived, compose a message containing the soccer match information, the most recent '7 days' weather forecast for the match place and date, and the stadium information, then deliver it at midnight of the day." This requirement can be met by defining a new information delivery channel, which integrates the three information sources and sends messages periodically.

Subsection 3.3 shows how this requirement can be specified in our framework. Section 4 gives data arrival property specifications for this example. Finally, Section 6 shows how ECA rules to achieve this requirement are derived from the specifications.

3. Overview of Integration Framework

This section overviews our integration framework [11, 13], which forms the basis of this work.

3.1. Architecture

Our framework is based on mediator/wrapper-based information integration architecture (Figure 2). The relational model is used as a common data model at the mediator level. Wrappers are associated with underlying information sources. To cope with data incoming from dissemination-based information sources, we allocate wrappers, named DIS wrappers, to dissemination-based information sources. When a DIS wrapper receives a data unit from the server, it raises an arrival event to notify arrival of the data unit. The timer module also raises events; it raises alarm events to notify time progress.

The rule processing module maintains ECA rules, which specify actions to be triggered by the events raised by the DIS wrappers and the timer module. When events are raised, the rule processing module selects relevant ECA rules, which invoke actions related to data storage, integration, and delivery[2].

Users can define new information delivery channels on this system by giving relational algebra-based requirement specifications. From the specifications, the rule generation module derives the ECA rules required to meet the requirements. More details of requirement specifications are explained in Subsection 3.3.

In our framework, the integration requirement given in Section 2 is processed as follows:

1. When the DIS wrapper receives a data unit from the weather information delivery channel, the wrapper

[2]Events raised while processing an ECA rule are queued and processed sequentially.

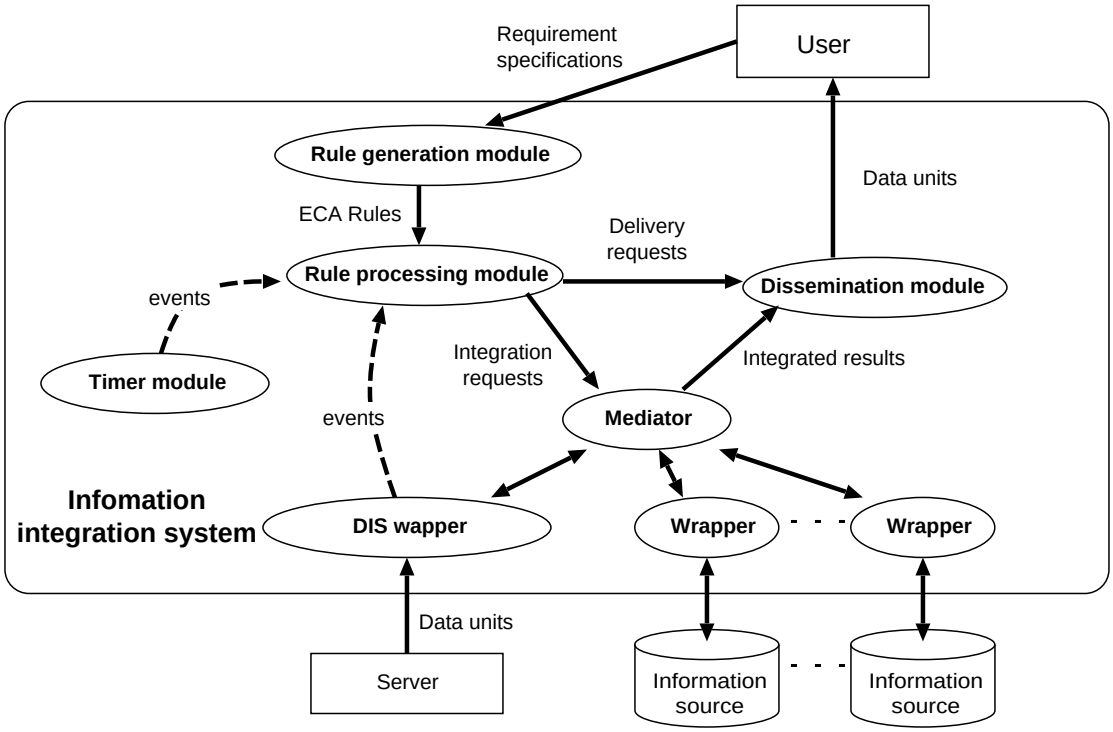

Figure 2. Integration System Architecture

translates it into a tuple in the relational model, and notifies the rule processing system of an arrival event.

2. The rule processing module invokes the ECA rule corresponding to the notified event, and sends a request to the mediator to store the new tuple provided by the wrapper in a temporary relation.

3. When a data unit from the soccer information delivery channel has arrived, the DIS wrapper also raises an arrival event.

4. The rule processing module requests the mediator to store the data in a temporary relation, and sets a timer alarm for midnight of the day.

5. At midnight, the timer module raises an event. The rule processing module then requests the mediator to compose a message by integrating the soccer information, the most recent weather forecast information, and the stadium information.

6. Finally, the rule processing system orders the dissemination module to deliver the composed message to the user.

3.2. ECA Rules

This subsection explains the ECA rules employed in our framework. The ECA rules in this paper are basically the same as those in active databases [12]. An ECA rule consists of three parts: the event (on), condition (if), and action (do) clauses.

The event clause specifies an event that triggers the rule. There are two kinds of events: arrival and alarm. An arrival event is raised from a DIS wrapper that notifies the arrival of a new data unit. An alarm event is raised from the timer module when the set time has arrived.

The condition clause specifies a precondition that must be met for the action clause to be executed. In this context, we allow condition clauses to contain selection conditions in the relational algebra expression.

The action clause specifies data storage, integration, and/or delivery actions. These actions are specified in expressions involving the relational algebra, assignments to temporary relations, and information delivery operators. Details of the operators are given in [11].

As explained in Subsection 3.1, operations such as data storage, integration, and delivery are involved in data integration. We divide this process into the following three phases:

1. *Storage of data units*:
   In this phase, triggered by arrival events, the mediator confirms that data is needed to fulfill user requirements. If data is needed, it is stored in temporary relations. ECA rules employed to implement this phase are called *storage rules*.

2. *Generation of new data units*:
   In this phase, triggered by alarm events, the mediator integrates data in temporary relations and other underlying information sources, and generates the requested data unit. The delivery module is then invoked for the data delivery. ECA rules used for this phase are called *generation rules*.

3. *Disposal of unnecessary data units*:
   Unnecessary data in temporary relations must be periodically thrown away. ECA rules used for this purpose are called *garbage disposal rules*. They are triggered by alarm events.

The following is a storage rule to store new data units from the weather information channel. It confirms that the new data unit is '7 days' information and, if it is, stores it in a temporary relation.

Rule	$Storage_{Weather}$
on:	$arrival_{N_{Weather}}$
if:	$N_{Weather}.type =$ '7 days'
do:	$Temp_{Weather} += N_{Weather}$

Here, $N_{Weather}$ represents the new data unit from the Weather channel, and $Temp_{Weather}$ += $N_{Weather}$ appends it to the temporary relation $Temp_{Weather}$ in the mediator.

3.3. Requirement Specification

As explained in Subsection 3.1, we use the relational model as a common model at the mediator level. Thus, we also model dissemination-based information sources as relations. A relation modeling data incoming from a dissemination-based information source is called an *I-sequence relation*. One data unit corresponds to a relational tuple. Each I-sequence relation has an ITS time stamp attribute. It designates the arrival time of the corresponding data unit from the information source. A relation modeling data outgoing through a newly defined information delivery channel is called an *O-sequence relation*. Each O-sequence relation has an OTS time stamp attribute. It designates the scheduled delivery time of the corresponding data unit.

Since we have modeled dissemination-based information sources as relations, we can employ relational algebra to define a new information delivery channel. The main part is to specify how to create an O-sequence relation from I-sequence relations (and other relations representing conventional information sources). Definition of a new delivery channel is given by the following expression.

$$O_{new} = \Omega_{f(I_k.ITS)}(E(I_1, \cdots, I_n)),$$

where O_{new} is an O-sequence relation, E is a relational algebra expression to specify how to derive it from I-sequence relations $I_1, \cdots, I_n$ and other conventional relations. The operator $\Omega_{f(I_k.ITS)}$ is new. It adds the OTS attribute to the relation obtained by $E(I_1, \cdots, I_n)$, sets $f(I_k.ITS)$ as its values, and removes all ITS attributes included in E. The function $f(I_k.ITS)$ is a time stamp function, and derives a new time stamp from the ITS attribute value in I-sequence relation I_k. I_k is called the *master I-sequence relation (or master source)*.

Time stamp functions considered in the paper are as follows:

- **immediate**(t) : Returns the given time stamp t itself.
- **next**$_p(t)$: Returns the time stamp, matching the temporal pattern p, chronologically next to the time stamp t.
- **previous**$_p(t)$: Returns the time stamp, matching the temporal pattern p, chronologically previous to the time stamp t.
- **after**$_{\delta t}(t)$: Returns the time stamp after the time interval δt from the time t.
- **before**$_{\delta t}(t)$: Returns the time stamp before the time interval δt from the time t.

The temporal pattern p is given in one of the following formats:

1. day:hour:minute:second
2. dayofweek:hour:minute:second

The fields 'day', 'hour', 'minute', and 'second' can be non-negative integers or the wild card symbol '*'. The field 'dayofweek' can contain one of the strings { Sun, Mon, Tue, Wed, Thu, Fri, Sat } or '*'. The time interval δt is specified using format 1, but '*' is not allowed.

For example,

$$next_{*:0:0:0}(Soccer.ITS)$$

gives the time stamp corresponding to the next midnight from the arrival time *Soccer.ITS*.

The new information delivery channel required in Section 2 can be specified as follows within this framework:

$$
\begin{aligned}
O_{new} = \Omega_{next_{*:0:0:0}(Soccer.ITS)}(& \\
& Soccer \\
& \bowtie_{Soccer.date=Weather.date} \\
& \quad \wedge previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS)) \leq Weather.ITS \\
& \quad \wedge Weather.ITS < next_{*:0:0:0}(Soccer.ITS) \\
& \sigma_{Weather.type=\text{`7 days'}}(Weather) \\
& \bowtie_{Weather.location=Stadium.location} \\
& \quad \wedge Soccer.place=Stadium.name \\
& Stadium \\
), &
\end{aligned}
$$

where *Soccer* and *Weather* are the I-sequence relations, and σ and $\bowtie$ represent selection and join, respectively. The following join condition requires that the most recent '7 days' weather forecast information be joined with the arrived soccer information:

$$
\begin{aligned}
& previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS)) \leq Weather.ITS \\
& \wedge Weather.ITS < next_{*:0:0:0}(Soccer.ITS)
\end{aligned}
$$

Fig 3 illustrates how data for the new channel is specified in this expression.

4. Data Arrival Property Specification

This paper extends our previous framework by taking into account data arrival properties of dissemination-based information sources. This section explains the specification of source data arrival properties of dissemination-based information sources.

The explanation utilizes expressions used for constraint specification in [14]. The data arrival property specification (shortly, property specification) takes the following form:

$$I_i \equiv \sigma_{PROP_i}(I_i)$$

It states that the left side is guaranteed to be the same as the right side. Therefore, when an expression refers to I-sequence relation I_i, we can replace it with the selection on the right side.

The example in Section 2 involves two dissemination-based information sources. Data units from the soccer information channel arrive every Wednesday. We can specify this property as follows:

$$
\begin{aligned}
& Soccer \equiv \\
& \sigma_{previous_{Wed:0:0:0}(Soccer.ITS) \leq Soccer.ITS \\
& \quad \wedge Soccer.ITS < after_{1:0:0:0}(previous_{Wed:0:0:0}(Soccer.ITS))}(Soccer)
\end{aligned}
$$

This expression assures that all the ITS values in *Soccer* are included in the range from 0 o'clock on Wednesday to 0 o'clock on Thursday.

Data units from the weather information channel arrive at 6 o'clock everyday. We can specify this property as follows:

$$
\begin{aligned}
& Weather \equiv \\
& \quad \sigma_{Weather.ITS=after_{0:6:0:0}(previous_{*:0:0:0}(Weather.ITS))}(Weather)
\end{aligned}
$$

It assures that all the ITS values in *Weather* imply 6 o'clock.

5. Basic Properties of Requirement Specification

Before we derive ECA rules from given requirement specifications, we check their consistency and satisfiability. In our previous work [13], we outlined the idea of consistency. We elaborate on that idea here and take data delivery properties into account in checking consistency and satisfiability.

5.1. Consistency

Assume that the user gives the following requirement specification.

$$O_{new} = \Omega_{immediate(Soccer.ITS)}(Soccer \times Weather)$$

This specification can be interpreted as "When soccer information has arrived, integrate it with all the weather information (which is delivered from the beginning to the end of the service from the Weather channel), and deliver the integration result immediately". It is, of course, impossible to obtain weather information to be delivered in the future when the soccer information has arrived. Therefore, the system cannot compose the integration result correctly, so this requirement is not feasible. We define notion of *consistency* to detect such requirements that are not feasible.

Before explaining the notion of consistency, we introduce the inverse $f^{-1}(t)$ of the time stamp function $f(t)$ as follows:

$$f^{-1}(t) = \{u | f(u) = t\}$$

For each time stamp function introduced in Subsection 3.3, the inverse can be defined as follows.

1. $\mathbf{immediate}^{-1}(t) = \{u | u = t\}$
2. $\mathbf{next}_p^{-1}(t) = \{u | previous_p(t) \leq u < t\}$
3. $\mathbf{previous}_p^{-1}(t) = \{u | t < u \leq next_p(t)\}$
4. $\mathbf{after}_{\delta t}^{-1}(t) = \{u | u = before_\delta(t)\}$
5. $\mathbf{before}_{\delta t}^{-1}(t) = \{u | u = after_\delta(t)\}$

Definition 1 *Suppose a requirement specification* $O_{new} = \Omega_{f(I_k.ITS)}\ (E(I_1, \cdots, I_n))$ *and property specifications* $I_1 \equiv \sigma_{PROP_1}(I_1), \cdots, I_n \equiv \sigma_{PROP_n}(I_n)$ [2] *are given. Then, let* E' *be the following expression.*

[2] When no data arrival property specification is given for I_i, $PROP_i = true$.

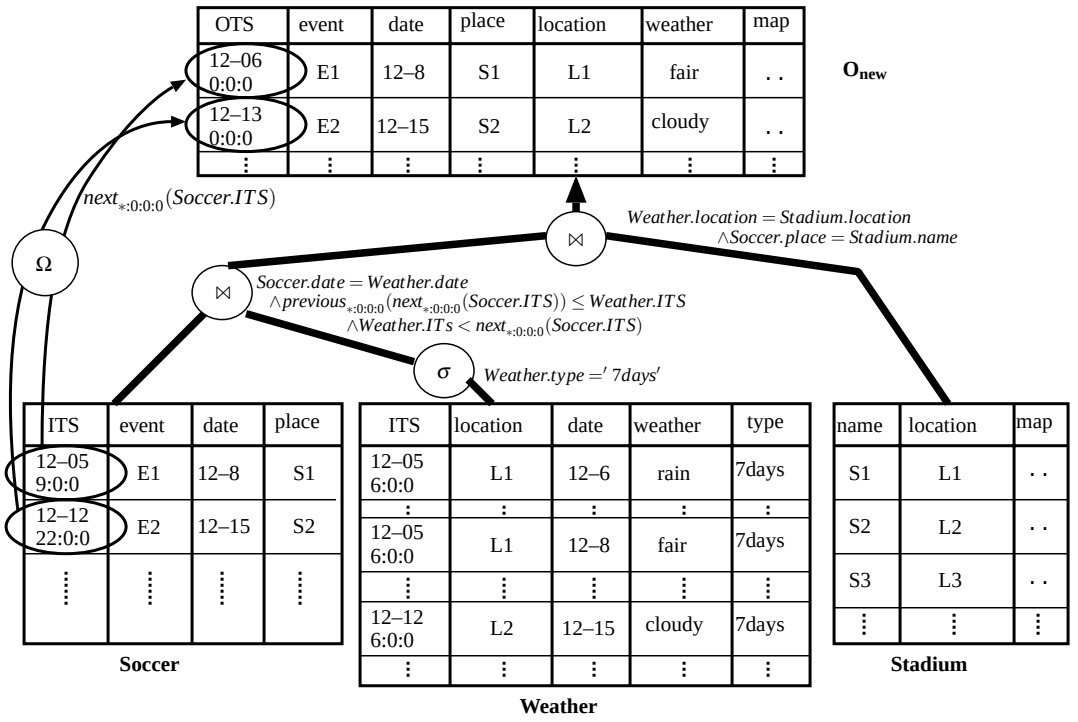

Figure 3. Requirement Specification

$$E'(I_1,\cdots,I_n,t) = \sigma_{I_k.ITS\in f^{-1}(t)}(E(\sigma_{PROP_1}(I_1),\cdots,\sigma_{PROP_n}(I_n)))$$

If the following equation holds for any time stamp u, the requirement specification is said to be consistent.

$$E'(I_1,\cdots,I_n,f(u)) = E'(\sigma_{ITS\leq f(u)}(I_1),\cdots,\sigma_{ITS\leq f(u)}(I_n),f(u))$$

□

E' selects tuples to be delivered at given time t under the constraint that the data arrival properties are assured. Definition 1 says that the integration result delivered at time $f(u)$ should be composed only from the information that arrived before or at $f(u)$.

We check consistency of the requirement specification in Section 2. From the relational algebra-based channel definition in Subsection 3.3 and property specifications in Section 4, we get the following expression as E' (Figure 4). Note that $\sigma_{Soccer.ITS\in next^{-1}_{*:0:0:0}(t)}(E) = \sigma_{previous_{*:0:0:0}(t)\leq Soccer.ITS\wedge Soccer.ITS<t}(E)$ holds from basic properties of the time stamp function *next*.

$$
\begin{aligned}
&\sigma_{previous_{*:0:0:0}(t)\leq Soccer.ITS\wedge Soccer.ITS<t}(&& \Big\}C_1\\
&\quad(\\
&\quad\sigma_{previous_{Wed:0:0:0}(Soccer.ITS)\leq Soccer.ITS \wedge Soccer.ITS<after_{1:0:0:0}(previous_{Wed:0:0:0}(Soccer.ITS))}(Soccer) && \Big\}C_2\\
&\quad)\\
&\quad\bowtie_{Soccer.date=Weather.date \wedge previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS))\leq Weather.ITS \wedge Weather.ITS<next_{*:0:0:0}(Soccer.ITS)} && \Big\}C_3\\
&\quad(\\
&\quad\sigma_{Weather.type=\text{`7 days'} \wedge Weather.ITS=after_{0:6:0:0}(previous_{*:0:0:0}(Weather.ITS))}(Weather) && \Big\}C_4\\
&\quad)\\
&\quad\bowtie_{Weather.location=Stadium.location \wedge Soccer.place=Stadium.name} && \Big\}C_5\\
&\quad Stadium\\
&)
\end{aligned}
$$

This expression references the tuples in Soccer that meet the condition $Soccer.ITS < t$. Also, tuples referenced in Weather satisfy the condition $Weather.ITS < f(u) = next_{*:0:0:0}(u)$, since $next_{*:0:0:0}(u) = next_{*:0:0:0}(Soccer.ITS)$. The latter equation can be obtained by substituting

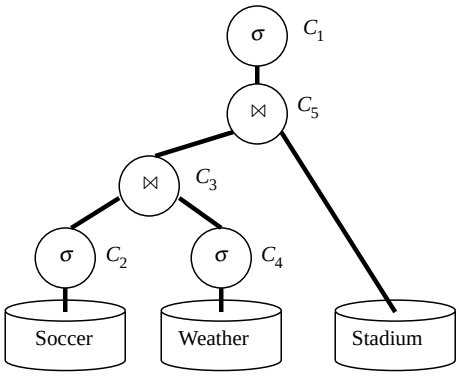

Figure 4. Expression E'

$next_{*:0:0:0}(u)$ for t in $previous_{*:0:0:0}(t) \leq Soccer.ITS \wedge Soccer.ITS < t$. Therefore, this requirement specification is consistent.

5.2. Satisfiability

Let us consider another requirement specification:

$$O_{new} = \Omega_{next_{Tue:0:0:0}(Soccer.ITS)}(\sigma_{Soccer.ITS \geq previous_{Sun:0:0:0}(Soccer.ITS)}(Soccer))$$

This requirement can be interpreted as "At 0 o'clock every Tuesday, deliver all the soccer information that has arrived from the previous Sunday up to the time." However, there is never any information to be delivered at 0 o'clock on Tuesday, since soccer information arrives on Wednesday. Therefore, the user will never get the specified information.

Deriving ECA rules for such requirements is meaningless. Beyond that, the specification could be an error, because the user would not intentionally specify a meaningless requirement. We can detect such error-prone requirements by taking into account the data arrival properties. We define the concept of *satisfiability* for this purpose.

Definition 2 *Suppose a requirement specification* $O_{new} = \Omega_{f(I_k.ITS)}(E(I_1,\cdots,I_n))$ *and property specifications* $I_1 \equiv \sigma_{PROP_1}(I_1),\cdots,I_n \equiv \sigma_{PROP_n}(I_n)$ *are given. Let* E' *be the following expression.*

$$E'(I_1,\cdots,I_n,t) = \sigma_{I_k.ITS \in f^{-1}(t)}(E(\sigma_{PROP_1}(I_1),\cdots,\sigma_{PROP_n}(I_n)))$$

If the following equation holds for some time stamp $f(u)$*, which is greater than "now"*[8]*, the requirement specification is said to be satisfiable.*

$$E'(I_1,\cdots,I_n,f(u)) \neq \phi$$

[8]The time stamp "now" represents the current time.

□

Definition 2 says that satisfiable requirements define new channels that will actually deliver information in the future.

We check satisfiability of the requirement specification in Section 2. We have already obtained E' in Subsection 5.1. Assume that a '7 days' weather information data unit arrives at 6 o'clock next Wednesday, and a soccer information data unit arrives on the same day. We can then compose the integration result by using E' at the midnight of the day. Therefore, this requirement specification is satisfiable.

5.3. Verification of Consistency and Satisfiability

When a requirement specification is given, we must confirm that it is consistent and satisfiable. To do this, we map time stamp values into integers, and time stamp functions into integer expressions (e.g. $after_{\delta t}(t) \Rightarrow t + \delta t$). Thus, conditions including time stamp functions can be translated into conditions on integers and integer variables. In this way, we can derive inequalities on integer variables. Verification of consistency and satisfiability can be reduced to the problem of deciding whether the inequalities have any solutions. This check can be done using integer programming algorithms. Further details are outside the scope of this paper and omitted here.

6. Derivation of ECA Rules

This section explains how to derive ECA rules by taking into account data arrival properties.

6.1. Overview

Given a requirement specification $O_{new} = \Omega_{f(I_k.ITS)}(E(I_1,\cdots,I_n))$ and property specifications $I_1 \equiv \sigma_{PROP_1}(I_1),\cdots,I_n \equiv \sigma_{PROP_n}(I_n)$, ECA rules can be derived as follows:

1. From the given specification, derive the following expression E':
   $$E'(I_1,\cdots,I_n,t) = \sigma_{I_k.ITS \in f^{-1}(t)}(E(\sigma_{PROP_1}(I_1),\cdots,\sigma_{PROP_n}(I_n)))$$
2. Confirm that the specification is consistent. If it is not, exit from the procedure.
3. Confirm that the specification is satisfiable. If it is not, exit from the procedure.
4. Push down selection conditions as far as possible using the conventional query optimization scheme [15], and rewrite the above expression into the following form:
   $$E''(\sigma_{P_1(t) \wedge Q_1}(I_1),\cdots,\sigma_{P_n(t) \wedge Q_n}(I_n),t),$$

where $P_i(t)$ is the selection condition on ITS attribute of information source I_i, and Q_i expresses other selection conditions.

5. Derive a storage rule for each I-sequence relation $I_i (i = 1, \cdots, n)$.

6. Derive a generation rule.

7. Derive a garbage disposal rule for each I-sequence relation $I_i (i = 1, \cdots, n)$.

6.2. ECA Rule Derivation

From the requirement specification in Subsection 3.3, we derive ECA rules while taking into account the property specifications in Section 4. In Section 5 we already explained steps 1 to 3 in the derivation process above. Here, we describe the remaining steps.

At step 4, we push down the selection conditions as far as possible. After eliminating redundant conditions, we get the following expression as E'' from the expression E' shown in Subsection 5.1.

$$
\begin{array}{l}
\Big(\\
\quad \Big\{ \sigma_{previous_{*:0:0:0}(t) \le Soccer.ITS \wedge Soccer.ITS < t}(Soccer) \Big\} C_s \\
\Big) \\
\bowtie_{Soccer.date = Weather.date} \\
\quad \wedge previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS)) \le Weather.ITS \\
\quad \wedge Weather.ITS \le next_{*:0:0:0}(Soccer.ITS) \\
\Big(\\
\quad \Big\{ \sigma_{previous_{*:0:0:0}(t) \le Weather.ITS} \\
\qquad \wedge Weather.ITS < t \\
\qquad \wedge previous_{Wed:0:0:0}(t) \le Weather.ITS \\
\qquad \wedge Weather.ITS < after_{1:0:0:0}(previous_{Wed:0:0:0}(t)) \\
\qquad \wedge Weather.type = \text{`7 days'} (Weather) \Big\} C_w \\
\Big) \\
\bowtie_{Weather.location = Stadium.location} \\
\quad \wedge Soccer.place = Stadium.name \\
Stadium
\end{array}
\Big\} E''
$$

Here, selection condition C_s for Soccer says that the new data unit to be delivered at time t on some day is composed by Soccer data units that have arrived up to the time on the same day. Selection condition C_w for Weather says that only data units that arrived on Wednesday and have the type '7 days' are needed in addition to the same condition for Soccer.

At steps 5, 6, and 7, we derive the three kinds of ECA rules explained in Subsection 3.2. The basic derivation scheme is the same as in [13] except that property specifications are taken into account.

The storage rule for Soccer is derived based on the selection condition C_s as follows.

Rule Storage$_{Soccer}$
on: arrival(N_{Soccer})
if: $N_{Soccer}.ITS \in \bigcup_{t \ge now} \{u | previous_{*:0:0:0}(t) \le u \wedge u < t\}$
do: $Temp_{Soccer}$ += N_{Soccer} ;
$setTimer(next_{*:0:0:0}(N_{Soccer}.ITS), new)$;

As mentioned in Subsection 3.2, N_{Soccer} represents the new Soccer data unit. The setTimer operator in the action clause is used to set the timer alarm for midnight of the day. This rule is triggered by the arrival of a soccer information data unit, and it stores the data unit into the temporary relation $Temp_{Soccer}$ and sets the timer alarm.

The storage rule for Weather is derived based on the selection condition C_w as follows.

Rule Storage$_{Weather}$
on: arrival($N_{Weather}$)
if: $N_{Weather}.ITS \in$
$\bigcup_{t \ge now} \{u | previous_{*:0:0:0}(t) \le u \wedge u < t$
$\wedge previous_{Wed:0:0:0}(t) \le u$
$\wedge u < after_{1:0:0:0}(previous_{Wed:0:0:0}(t))\}$
$\wedge N_{Weather}.type =$ '7 days'
do: $Temp_{Weather}$ += $N_{Weather}$;

This rule is triggered by the arrival of a weather information data unit. If it arrives on Wednesday and has the type '7 days,' it stores the data unit into the temporary relation $Temp_{Weahter}$.

For this example scenario, the following generation rule is derived.

Rule Generation$_{new}$
on: alarm(*new*)
if: true
do: $O_{new} =$
$\Big($
$\sigma_{previous_{*:0:0:0}(t) \le Soccer.ITS \wedge Soccer.ITS < t}(Temp_{Soccer})$
$\bowtie_{Soccer.date = Weather.date}$
$\wedge previous_{*:0:0:0}(next_{*:0:0:0}(Soccer.ITS)) \le Weather.ITS$
$\wedge Weather.ITS \le next_{*:0:0:0}(Soccer.ITS)$
$\sigma_{previous_{*:0:0:0}(t) \le Weather.ITS \wedge Weather.ITS < t}$
$(Temp_{Weather})$
$\bowtie_{Weather.location = Stadium.location}$
$\wedge Soccer.place = Stadium.name$
$Stadium$
$\Big)$;
$Deliver(O_{new})$;

The action clause in the generation rule basically corresponds to E''. However, the selection condition that Weather information data units should have arrived on Wednesday and have the type '7 days' is omitted, since this condition is guaranteed by the storage rule for Weather. *Deliver* in the action clause is the operator to deliver the integrating result. This rule means "When the timer alarm *new* is raised, compose a new message by integrat-

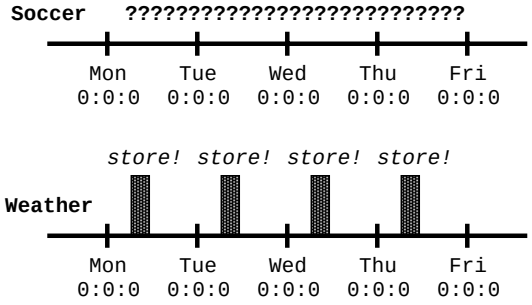

Figure 5. Without Property Specification

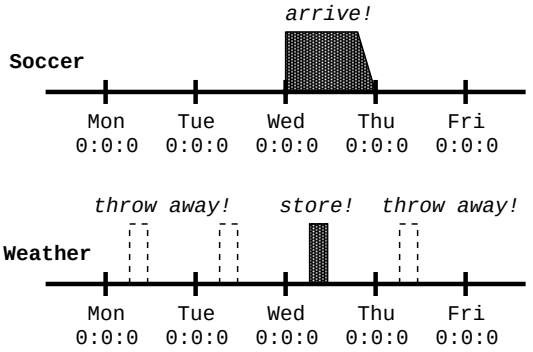

Figure 6. With Property Specification

ing the soccer match information in the temporary relation $Temp_{Soccer}$, the most recent weather information in the temporary relation *TempWeather*, and the corresponding stadium information. Then, deliver the integration result."

We can derive the garbage disposal rules in a similar manner. We can cope with situations in which multiple user requirements are given as discussed in [13].

6.3. Discussion

When no data arrival properties are given, we get the following storage rule for Weather.

Rule Storage$_{Weather}$
on: arrival($N_{Weather}$)
if: $N_{Weather}.ITS \in \bigcup_{t \geq now}\{u | previous_{*:0:0:0}(t) \leq u \wedge u < t\}$
$\wedge N_{Weather}.type =$ '7 days'
do: $Temp_{Weather}$ += $N_{Weather}$;

In contrast to the storage rule given in Subsection 6.2, this rule stores all weather information data units that have the type '7 days.'

In this example, the user requests that the most recent weather information be joined with the soccer information. Without the property specification for Soccer, of course, the system cannot predict when the soccer information arrives. Thus, '7 days' weather forecast data units must always be stored (Figure 5).

On the other hand, given the soccer property specification, the system can derive more strict conditions. Since soccer information arrives only on Wednesday, '7 days' weather information data units need to be stored only on Wednesday (Figure 6). The storage rule for Weather presented in Subsection 6.2 takes this property into account.

7. Related Works

There are several works related to the integration of dissemination-based information sources.

DBIS [16, 17] and *Muffin* [18] are examples of systems that can extract information from dissemination-based information sources. DBIS implements accesses to multiple dissemination-based information sources and supports information selection based on user profiles. Muffin can create a new virtual channel based on the user profile. To select information from multiple dissemination-base information sources, it uses frequency of delivery, freshness, and popularity as well as similarity. These researches concentrate on information selection from dissemination-based information sources.

OpenCQ [19] is an information integration system for the distributed information mediation system. This system is based on the event-driven approach and supports continual queries. A continual query consists of three components: normal query, trigger condition, and termination condition. Whenever a new update event occurs and a trigger condition becomes true, queries are executed. Their work is related to our approach in the context of event-driven information integration. In OpenCQ, the user must write the continual queries directly.

Some other works are also related to the integration of dissemination-based information sources. To the best of our knowledge, however, derivation of rules to realize active information integration from algebraic requirement specifications is original with our approach. Specifically, in this paper, we have extended our framework by incorporating data arrival properties. Works related to periodic temporal data sequences are as follows:

- [20] proposed *generalized relations*, and an extended relational algebra. It can handle *linear repeating points* (LRPs): sequences of integers in the form of $kn + c$, where k and c are constants and n is an arbitrary integer value. A generalized relation represents periodic temporal data sequences as generalized tuples using LRPs and constraints. However, generalized relations are not appropriate to express dissemination-based information sources, because it is usually impossible to completely decide the incoming data sequences in advance.

- [21] proposed methods to select appropriate data combination based on the concept of freshness and synchronousness from multiple periodic data sequences with different intervals. They also showed how to precompute the optimum data combination for consecutive queries. With our approach, data intervals are not necessarily fixed, and integration requirements are explicitly specified in relational algebra.

8. Conclusion

We have proposed a scheme to integrate dissemination-based information sources, using source data arrival properties. In our previous work, based on mediator/wrapper-based information integration architecture and ECA rule processing, we proposed the derivation of ECA rules from relational algebra-based user requirement specifications. In this paper, we have extended our scheme to take into account data arrival properties of dissemination-based information sources. We defined consistency and satisfiability of requirement specifications under the given data arrival properties. We have shown how we can derive ECA rules incorporating the data arrival properties, once the requirements are confirmed to be consistent and satisfiable. Implementation of the extended scheme presented in this paper is ongoing based on our information integration system *InfoWeaver*[3, 11].

Acknowledgments

This research is supported in part by the Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science and the Ministry of Education, Culture, Sports, Science and Technology.

References

[1] R. Domenig and K. R. Dittrich. An Overview and Classification of Mediated Query Systems. *ACM SIGMOD Record*, 28(3), pp.63–72, 1999.

[2] A. Elmagarmid, M. Rusinkiewicz, and A. Sheth (Eds.). Management of Heterogeneous and Autonomous Database Systems. Morgan Kaufmann, 1999.

[3] H. Kitagawa, A. Morishima, and H. Mizuguchi. Integration of Heterogeneous Information Sources in InfoWeaver. *Advances in Database and Multimedia for the New Century – A Swiss/Japanese Perspective*, World Scientific Publishing, pp. 124–137, 2000.

[4] A. Morishima and H. Kitagawa. InfoWeaver: Dynamic and Tailor-Made Integration of Structured Documents, Web, and Databases. *Proc. ACM DL '99*, pp. 235–236, 1999.

[5] G. Wiederhold. Mediators in the Architecture of Future Information Systems. *IEEE Computer*, Vol 25, No 3, pp.38–49, 1992.

[6] M. Franklin and S. Zdonik. Data in Your Face: Push Technology in Perspective. *Proc. ACM SIGMOD*, pp.516–519, Jun, 1998.

[7] S. Ramakrishnan and V. Dayal. The PointCast Network. *ACM SIGMOD Record*, 27(2), p. 520, 1998.

[8] Infogate Inc. Infogate. http://www.infogate.com/.

[9] TV-Asahi Data Inc. ADAMS. http://www.tv-asahidata.com/.

[10] INFOCITY Inc. bitcast. http://www.bitcast.ne.jp/.

[11] H. Mizuguchi, H. Kitagawa, Y. Ishikawa, and A. Morishima. A Rule-oriented Architecture to Incorporate Dissemination-based Information Delivery into Information Integration Environments. *Proc. 2000 ADBIS-DASFAA*, pp. 185–199, 2000.

[12] N. W. Paton and O. Diaz. Active Database Systems. *ACM Comp. Serv.*, 31(1), 1999.

[13] H. Kitagawa, T. Kajino, and Y. Ishikawa. Algebraic Service Specification and Rule Generation for Integrating Multiple Dissemination-Based Information Sources. *Proc. 7th International Conference on Database Systems for Advanced Applications*, pp. 344–351, 2001.

[14] H. Garcia-Molina, W. Labio, and J. Yang. Expiring Data in a Warehouse. *Proc. 24th VLDB Conference*, pp. 500–511, 1998.

[15] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price. Access Path Selection in a Relational Database Management System. *Proc. ACM SIGMOD Conference*, pp. 23–34, 1979.

[16] D. Aksoy, M. Altinel, R. Bose, U. Cetintemel, M. Franklin, J. Wang, and S. Zdonik. Research in Data Broadcast and Dissemination. *Proc. AMCP '98*, pp. 194–207, 1998.

[17] M. Altinel, D. Aksoy, T. Baby, M. Franklin, W. Shapiro, and S. Zdonik. DBIS Toolkit – Adaptable Middleware for Large-Scale Data Delivery. *Proc. ACM SIGMOD '99*, 1999.

[18] M. Qiang, H. Kondo, K. Sumiya, and K. Tanaka. Virtual TV Channel: Filtering Merging and Presenting Internet Broadcasting Channels. *ACM DL Workshop on WOWS*, 1999.

[19] L. Liu, C. Pu, and W. Tang. Continual Queries for Internet Scale Event-Driven Information Delivery. *IEEE TKDE*, 11(4), pp. 610–628, 1999.

[20] F. Kabanza, J. M. Stevenne, and P. Wolper. Handling Infinite Temporal Data. *Proc. 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 392–403, 1990.

[21] K. Munakata, M. Yoshikawa, and S. Uemura. Integrating Periodic Data Sequences Based on Freshness and Synchronousness. *Proc. 10th IEEE Workshop on Research Issues in Data Engineering*, pp. 47–54, 2000.