Tracing Traffic Information Identity in Distribution Systems

P. Bureš
Czech Technical University in Prague, Prague, Czech Republic

J. Vlčinský*
CAD programy, Karviná, Czech Republic
Corresponding authors: jan.vlcinsky@cad-programs.com

ABSTRACT: Tracing messages and information passing through a distribution system is necessary for resolving problems and improving performance. This paper summarizes the requirements, analyzes the aspects, proposes a conceptual approach, gives an example of possible implementation, and summarizes the findings. A number of challenges are uncovered, for example prescribed identification schemes in the system; versioning; repetition of incoming and outgoing messages; multiple output channels; recycling of IDs; invalid messages, etc. The key requirement is to trace messages with information on the input and output of the distribution system by assigning the proper identifier information. Unfortunately, identification schemes are different as to the information provider, in the central data store and in each output channel; the solution must keep track of these differences, and therefore events affecting the assigned information identity must be documented. The same applies to transactions, affecting information existence, as is the case with information deletion due to expiration. A sample of possible implementation shows that linking information between different types of log records might be greatly simplified if processed information keeps track of the transaction which has imported the information into the system.

KEY WORDS: identity, messaging, traffic and travel information.

1 INTRODUCTION

In the distribution of traffic information many parties and systems are involved. When designing, developing, testing and running such a system, tracing the identity of the information through the whole distribution process is essential to help things run, to improve performance and to resolve complaints and problems.

1.1 AIM, METHODOLOGY AND RESULTS

The aim of this paper is to analyze the identity of information in traffic information distribution systems, and propose a design for logging transactions which would allow the tracing of the identity of information in messages from the original information provider up to the consumer and from the consumer back to the provider. The solution must cope with the fact that with given identification schemes of the data provider and of the distribution channel, the same message can be broadcasted repeatedly and *the ID* in distribution channel might be recycled. The presented analysis and design is a byproduct of designing a system

for testing the distribution of traffic information in TPEG format using distribution over the Internet and DAB - Digital Audio Broadcast (Vlcinsky, 2009). TPEG stands for a specific format of traffic messages which allow a precise description of traffic related events and their location. The design was created using a software development methodology, called the Unified process (Arlow, 2007), which introduces 5 main activities (Requirements, Analysis, Design, Implementation and Testing). The first two activities were used (Requirements and Analysis) and their corresponding models were covered by a team review. The core results are analytical documents consisting of an analytical class model and use a case model. In this paper, requirements and assumptions' are summarized and recommendations for tracing the identity are stated. These results shall generally be usable in many information distribution systems, not only being limited to TPEG, or even only to traffic information.

1.2 WHAT ARE TPEG AND DAB?

TPEG is a new standardized technology for the encoding and distribution of Traffic and Travel Information providing modular a toolkit solving various applications, transmission methods, location referencing methods and devices. *DAB* is a digital radio technology for broadcasting radio stations providing a primary service (radio stations) and also a data service, usable for the distribution of traffic information.

1.3 EVENT, INFORMATION, MESSAGE AND OTHER MESSAGING TERMS

When an *event* happens, (primary data information) a *provider* should create *information* about this event. The information is then transmitted in the form of different *messages* to other systems. In this paper we assume that the message comes into our *distribution system*, is stored internally there and might be distributed via distribution *channels* to *consumers*. If existing information is updated, a new *version* of the information for a given *ID* is created. As a result, the information about the same event is stored in multiple places. To update the information everywhere, information identifiers are used for updating messages.

1.4 IDENTITY, IDENTIFICATION SCHEMES AND *ID* RECYCLING

Information, describing one event, has an assigned *ID*, which uniquely identifies the information and together with a version identifies the unique status of the information in time. The *ID* is used as method for referring to one unique object and is considered as unique only within the so called *identification scheme*. If a passport number is an ID, and the passport was issued by the Czech Republic, then the identification scheme might be called the "Czech Passport". If the range of possible *ID* values is relatively small compared to the number of identified items, then the uniqueness cannot be guaranteed forever and the time for so-called *ID recycling* comes. Even with recycling, uniqueness must be respected at least at a given moment, so that at any one moment only one logical object can be referred to by one ID. When recycling, the *ID* must be supplemented by a timestamp or at least a time period to refer to a proper object.

1.5 SCOPE

The scope of the solution starts at the moment that the distribution system receives a message from the information provider and ends with sending the message to the consumer. Only the creation and collection of log data is discussed here. The evaluation of gathered data is outside of the scope of this paper.

2 TYPICAL TRAFFIC INFORMATION DISTRIBUTION SYSTEM

2.1 PRIMARY INFORMATION PROVIDER, DISTRIBUTOR AND CONSUMER

The primary information provider's main responsibility is to provide "good quality information". From an information identity point of view quality can be defined as the fact that the same event is described by a piece of information only once, so that no merging of duplicated information is necessary. The distributor receives messages with information from the provider and manages the distribution via one or more channels. From an information identity point of view, the distributor must internally keep track of each unique piece of information without creating unnecessary duplicates. The distribution channel is fed by data from the distribution system and allows broadcasting, or another method of distribution, to consumers. The consumer receives the transmitted information in the form of a message and uses it. For auditing purposes, the consumer shall be able to provide the received message in an unchanged form together with a timestamp denoting when the message with the information was received.

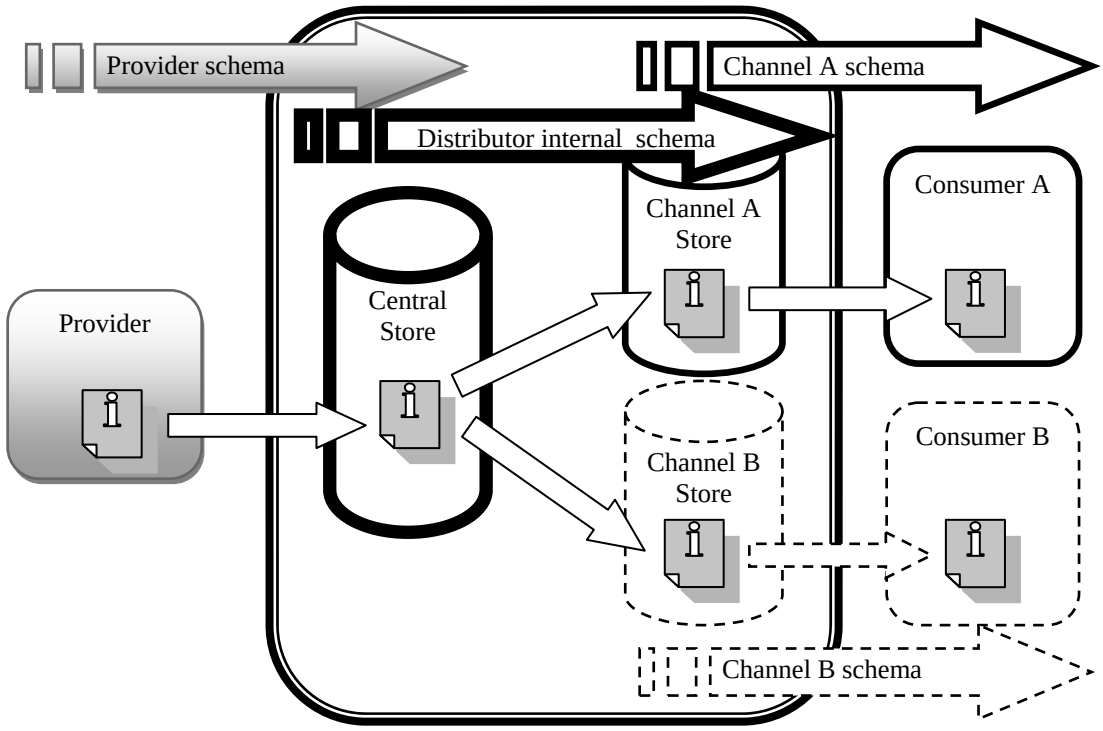

Figure 1: Information, messages and identification schemas through the distribution chain

2.2 CHALLENGES

Real life traffic information distribution (sending TPEG over DAB channel, TMC over RDS channel, etc.) brings the following challenges:

- *Fixed identification scheme of information provider* – the provider uses its own identification scheme and the distributor has no real chance of forcing the distributor to modify it.
- *Versioning information* - as time goes on, the same event might change and the information describing it needs to be updated. To allow information updating, versioning must be used.

- *Repeatedly received information* – the provider might send a message with the same information more than once. Assuming the *ID* of the information is the same in both cases the distributor must not create duplicates of the information.
- *Channel specific identification schemes* - each channel might use different identification scheme. *IDs* from the provider or internal *IDs* are often not usable due to the different types of *ID*.
- *Recycling IDs in channel* - some channels (e.g. TPEG over DAB) might use *ID* with a limited range of possible values, thus requiring the recycling of *IDs* over time. It is assumed that recycling is possible only for the *IDs* whose information has already expired.
- *Sending messages to the channel repeatedly* - some distribution channels (typically DAB for TPEG or RDS for TMC) use message repetition. The message is broadcast repeatedly to ensure at least one reception by the consumer and to allow error checking.
- *Information expiration* - information has a limited validity in time and might expire before it is distributed.
- *Message rejection* - some messages might be at some point in the distribution rejected as invalid and therefore not distributed further on.
- *Some messages in the channel might get lost* - not all messages sent to the channel are necessarily received by a consumer.

3 REQUIREMENTS

The following requirements for a logging system were identified:
- Traceability of messages and information.
- Logging must not overload the distribution system.
- Allow asynchronous processing.

3.1 TRACEABILITY OF MESSAGES AND INFORMATION

The system shall create log data, which allows the tracing of information traveling through the system by means of messages. Tracing starts at the moment a message is received by the distribution system and continues to the moment a message with the same information is sent to the channel.

Tracing of messages with information from the provider to the consumer must be possible. For each message with information from the provider, all messages sent to the consumer must be identifiable. Tracing of messages from the consumer back to the provider must be possible. For any message received by the consumer, the originating message with the information from the provider must be identifiable.

3.2 LOGGING MUST NOT OVERLOAD THE DISTRIBUTION SYSTEM

Because it is more important to distribute the information then to trace it, the system must not endanger the distribution itself. For this purpose the following requirements are set.
- Logging must not block the distribution and shall not consume many resources.
- Offline evaluation of logged data is acceptable.
- Allow the archiving and purging of older information and log records. An evaluation of log records must be possible even with an incomplete set of records.

3.3 ALLOW ASYNCHRONOUS PROCESSING

When a message is received, a number of different steps shall be done, like receiving a complete message, validating the message structure, reading the external information *ID* and comparing it to *IDs* already in the system, and assigning an internal *ID*. Despite the fact,

that all the steps can be completed in one (synchronous) step, the logging system shall not put this constraint on the distribution system and must also allow asynchronous processing (when for example the reception and validation might happen in different processes).

4 TYPICAL SCENARIOS IN INFORMATION DISTRIBUTION

Typical scenarios in information distribution are as follows:

- Scenario: Receive message from provider.
- Scenario: Assign information to output channels.
- Scenario: Send message with information to output channel.
- Scenario: Archive and purge expired information.

The scenarios mentioned here do not prescribe the internal logic of the distribution system (as logging shall fit into different distribution designs with different internal logic). Their purpose is to depict the general logic and serve as a basis for the description of a proposed logging solution.

4.1 SCENARIO: RECEIVE MESSAGE FROM PROVIDER

This scenario is triggered by the reception of a message containing information from the provider.

1. Receive data from the provider, store it as an incoming message, respond to the provider that the data have been received and create a log record of the type *DataReceived*.
2. Validate the structure of the incoming message if invalid; create a log record of the type *ProblemReport*.
3. Read the incoming message and try to find an external information *ID*, compare this with the internal records of the external *IDs,* assign an internal *ID* if necessary, and create log records of the type *ProviderIdFound* and *IntIdAssigned*.

5 STORE THE INFORMATION AND MARK IT READY FOR DISTRIBUTION

5.1 SCENARIO: ASSIGN OUTPUT CHANNELS TO INFORMATION

This scenario is triggered by the creation of new or updated information in the internal data store of the distribution system.

1. Evaluate the information and find which output channels shall be distributing it.
2. For each assigned channel:
    a. Store or update the information in the internal channel data store and create a log record of the type *AssignedToChannel.*
    b. If necessary, assign an internal channel information *ID* and create a log record of the type *ChannelIdAssigned*.

5.2 SCENARIO: SEND A MESSAGE WITH THE INFORMATION TO THE OUTPUT CHANNEL

Depending on the type of channel and client, this scenario can be initiated by a client request, or by the channel itself, which might run in a continuous loop. The following steps are for a channel which sends messages in a continuous loop:

1. Find the information to send in the internal channel data store
2. Send a message with the information to the channel and create a log record of the type *DataSentToChannel*
3. If applicable, wait for a response and record the result in a log record of the type *ChannelResponse*.
4. Continue with the next information to send.

5.3 SCENARIO: ARCHIVE AND PURGE THE EXPIRED INFORMATION

This scenario is typically triggered by an internal scheduler:

1. For each data store (central distribution store and internal channel stores) search for information which has expired.
2. Archive expired information.
3. Purge archived information from data stores
4. Create a log record of the proper type for each piece of purged information

6 SOLUTION

6.1 LOG ATOMIC TRANSACTION FOR ASYNCHRONOUS PROCESSING

Receiving a message is an important transaction and we might think of a log record telling us the time of reception, data validity, external *ID* and internal *ID*. However, this design would be forced to receive the message in one single synchronous step, or it would make asynchronous processing with logging quite difficult. Because of this, transactions like reception shall be split into small atomic transactions, which are then easy to record regardless of synchronous or asynchronous processing.

6.2 RELEVANT TRANSACTIONS

Any transaction which affects information existence or handles any type of *ID* for the information must be recorded by means of a log record. The following transactions are considered:

- *Transactions related to message reception*: Receiving a message from the provider, recognizing external information *ID*, assigning internal information *ID* and rejecting a message as invalid.
- *Internal transactions*: Purging expired information from the central or channel data store.
- *Transactions related to sending a message to a channel*: assigning information to the output channel, assigning channel specific *ID* to information, sending the message to the output channel, receiving a response from the output channel.

6.3 STRUCTURE OF LOG RECORDS

Structure of log reports must follow a few simple requirements:

- *Loose coupling*: to allow easy purging and archiving of log records without breaking data integrity, loose coupling must be used. Thus to limit the number of references, the structure must be as flat as possible, even if some data would have to be repeated. Another rule is that any referencing to other records or objects must be by value, which can be found in a (preferably archived) object.

- *Common attributes of a log record*: each log record shall contain information about when the transaction happened (timestamp), which systems participated in the transaction (can be given by context, but for multiple providers and multiple output channels it might

require using the system or channel name), identity and version of the information in the participating systems, a simple description of the result and, if necessary, the data of the exchanged message. To allow a standard and simplified description of possible problems, the general object *ProblemReport* might be used and referred to from any log record. The following Figure shows the type of log records proposed.

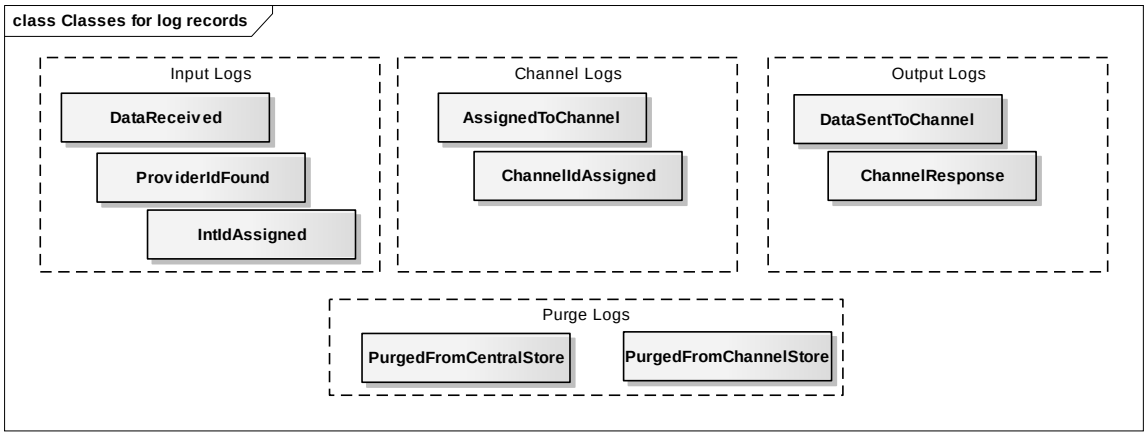

Figure 2: Classes for log records.

7 AN EXAMPLE OF POSSIBLE IMPLEMENTATION

To make the understanding of the proposed solution easier, an example of possible implementation is given. The distribution system shall have a single input channel, process TPEG RTM data and distribute it via two DAB channels, one for the eastern and another for the western part of a country. The information for the middle part of the country is distributed by both channels.

All log records will be written asynchronously to a database. The system will allow the export of log records in the XML Atom web syndication format with one entry per log record and using a specific namespace for the storing structure of the log record (each type of record is possible to store in such a structure). Offline processing of log records is expected in separate system, which will be able to load these XML files and do the processing and evaluation. To prevent data overrun in the log database, the scheduled task is used to archive and delete older records. Input log *DataReceived* also contains the data which were received to make the monitoring of the data exchange possible. Each import transaction gets a unique *uuid* string, which is stored with the received data, as well as with the log record. Input log *ProviderIdFound* contains the transaction *uuid* (to make linking to the *DataReceived* log record possible) and adds the *ID* and version found in the message. Input log *IntIdAssigned* contains the transaction *uuid*, together with the assigned internal information *ID* and the internal version. A field of type update, new, unchanged, ignored or updated is used to identify if the information was already in the system and what happened to it. Channel log *AssignedToChannel* uses the import transaction *uuid*, internal information *ID* and an internal version. Channel identification string “east” or “west” is used to denote the channel the information was assigned to. For the same import transaction more log records might appear as information, for the middle part of the country might fit into “east”, as well as “west”, channel region selection criteria. Channel log *ChannelIdAssigned* still uses the import transaction *uuid* (all the time related to the import to the distribution system, not to the channel). Channel identification, internal channel *ID* of the information and version is recorded. Output log *DataSentToChannel* uses the import transaction *uuid*. As information is sent to DAB repeatedly in cycles, the log also records the sequence number of the cycle. Channel identification, internal channel *ID* of the information and version is also recorded.

Output log *ChannelResponse* is not used as DAB does not return any return code. Purge log *PurgedFromCentralStore* is used when the scheduler runs a script archiving and purging the expired information and is written after the information is deleted from the central store. It contains the import transaction *uuid*, internal *ID* of the information and version. Purge log *PurgedFromChannelStore* has the same structure as the output log *DataSentToChannel*. Output log *DataSentToChannel* might optionally contain the data which are really sent. However, as sending data to channel is frequent activity, this option is used only when testing the system and not in production use, as it might slow down the system.

8 CONCLUSIONS

Tracing the identity of traffic information in real distribution systems might get complex and is a complicated topic. This paper tries to help by uncovering the many hidden aspects and by proposing general solutions. The distribution system must take into account that identification schemas at the provider, central distribution store and each distribution channel might be different.

The most important transaction in the system is the import of data from the provider. Using *uuid* for this transaction and keeping track of it through the whole distribution system simplifies the tracking path of information through all the processing steps. Each transaction affecting any type of *id* or version must be recorded. Recording the deletion of information due to expiration should not be omitted, as we would lose the information where the information disappeared.

9 AKNOWLEDGEMENT

The study was supported by the Czech Ministry of Transport, within research project R&D Nr. CG741-139-120 called “Use of new protocols and distribution channels for distribution of traffic information”.

10 REFERENCES

Arlow, J.; Neustadt, I.: 2007. *UML2 a unifikovaný proces vývoje aplikací Průvodce analýzou a návrhem objektově orientovaného softwaru*. Vydání první Brno: Computer Press, 2007. ISBN 978-80-251-1503-9.

Vlcinsky, J.: 2009. *Architecturing future-proof distribution of traffic and travel information using TPEG.* Transactions on Transport Sciences, Praha.