ABSTRACT
The Internet Open Trading Protocol (IOTP) is an electronic commerce protocol being developed by the Internet Engineering Task Force. The core of IOTP is a set of electronic transactions that reflect common trading activities such as purchasing goods or depositing funds. We use Coloured Petri Nets (CPNs) to specify IOTP. We enhance our previous specifications to include procedures for error handling and arbitrary transaction cancellation. This provides a complete specification of IOTP’s authentication and payment related transactions. Modularity and re-use are also improved. The new specification conforms to the narrative description of IOTP and allows us to analyse the protocol thoroughly.

Categories and Subject Descriptors
D.2.1 [Software Engineering]: Requirements / Specifications – analysis, specification, validation; J.8 [Internet Applications]: Electronic Commerce.

Keywords

1. INTRODUCTION
Electronic commerce (e-commerce) is a rapidly growing area with a multitude of underlying protocols developed to build the required information infrastructure. An important example of such a protocol supporting e-commerce applications is the Internet Open Trading Protocol (IOTP) [2]. It is developed by the Internet Engineering Task Force (IETF) [4] and published as Request For Comments (RFC) 2801 [2]. IOTP is a shopping protocol [1,6], which defines transactions for authentication, purchase, refund, deposit, withdrawal and value exchange. IOTP is designed to meet several design goals and is therefore complex.

For an e-commerce protocol such as IOTP, it is important to ensure that it works correctly and does not contain errors (e.g., deadlocks), which may cause disruption of the service or worse situations such as incorrect financial transactions. However, the RFC only provides a narrative description of IOTP, and some parts are ambiguous, difficult to understand and imprecise. Given the complexity of IOTP, implementations based on such an informal protocol specification [2] are likely to contain errors.

Formal methods [3] encompass mathematically based modelling languages, techniques and computer tools for specification, verification, implementation and testing of systems. They are necessary for construction of unambiguous and precise models which can be analysed by computer tools to identify errors and verify correctness before implementation. Application of formal techniques will lead to more reliable and trustworthy e-commerce protocols [11]. Our literature searches have not revealed any work on the application of formal methods to IOTP apart from our own [8-10], where we have used Coloured Petri Nets (CPNs) [5] for specification and analysis. The basic modelling approach and initial analysis of just the IOTP deposit transaction is given in [8]. In [9] we propose a simplified protocol architecture of IOTP based on our modelling experience. Our model of IOTP is extended in [10] to include authentication and all payment-related transactions, however, cancellation is restricted and error handling is not included. An analysis the purchase transaction is also reported.

Our previous work [8-10] on the modelling and analysis of IOTP represents an important step towards an improved specification of IOTP. In particular, the architecture is now more solidly defined. In this paper, we develop a refined specification that further exploits the modularity present in IOTP and enhances its readability by introducing a new hierarchical level associated with transactions, which re-uses modular exchange mechanisms in IOTP. The specification is also includes IOTP’s procedures for error handling and arbitrary transaction cancellation, which are essential for a complete specification.

The paper is organised as follows. Section 2 gives a short description of IOTP’s basic concepts. Section 3 introduces Coloured Petri Nets. We present our improved CPN model of IOTP in Section 4, and briefly discuss initial analysis results in Section 5. Finally, Section 6 concludes this paper.

2. INTERNET OPEN TRADING PROTOCOL
In this section, we briefly introduce the basic concepts and procedures of IOTP according to its definition in RFC 2801 [2]. Details will be given when the CPN model of IOTP is described.

Four interrelated concepts defined in IOTP are: trading roles, IOTP transactions, IOTP messages and document exchanges. A trading role is used to identify a certain role that an organisation can assume while trading. There are five IOTP trading roles: Consumer, Merchant, Payment Handler (a bank), Delivery Handler (a courier...
An IOTP transaction comprises a sequence of IOTP message exchanges between trading roles. IOTP defines an Authentication transaction and five payment-related transactions called Purchase, Refund, Deposit, Withdrawal and Value Exchange. A set of document exchanges are designed as building blocks for implementing IOTP transactions and consist of Authentication, Brand Dependent Offer, Brand Independent Offer, Payment, Delivery, and Payment-and-Delivery. Two important procedures related to IOTP transactions are: transaction cancellation and error handling. An IOTP transaction may be cancelled by any trading role engaged in that transaction. Rules are defined which determine how a transaction can be cancelled by a trading role. Errors that may occur during a transaction can be categorised into technical errors and business errors. IOTP error handling defined in the RFC is mainly concerned with how trading roles handle technical errors and exceptions in processing each IOTP message.

3. COLOURED PETRI NETS

CPNs [5] enhance Petri nets with data types. A CPN consists of two types of nodes, places (represented by ellipses) and transitions (rectangles), and directed edges known as arcs. A place is typed by a colour set and contains collections (multisets) of data items called tokens of the same type as the place. A transition represents an event and may have a guard associated with it. The guard is a boolean expression enclosed in square brackets. Arcs connect places to transitions and transitions to places, and are inscribed by expressions comprising variables, constants and functions. Variables are typed and can be assigned values known as binding.

A transition’s input places have arcs going to the transition, while its output places have arcs coming from the transition. A transition is enabled if sufficient tokens exist in each input place to match each respective input arc inscription when evaluated for a particular binding of its variables, and the transition guard evaluates to true for the same binding. The occurrence of an enabled transition removes tokens specified by the respective arc inscriptions from input places, and deposits tokens specified by inscriptions on the output arcs for the same binding into output places. The state of a CPN is called a marking and comprises the vector of tokens associated with each place of the CPN.

The hierarchical features of CPNs facilitate the construction of large models by using a number of CPN modules called pages. These pages are then related to each other in a well-defined way.

Design/CPN [7] is a software tool for the construction, simulation and analysis of CPN models. It comprises a graphical editor (with syntax checker), simulator and a state space tool. Design/CPN uses a variant of Standard ML [12] for net inscriptions and declarations of variables and types. It also supports hierarchical constructs. We use Design/CPN to create our model of IOTP in the next section.

4. CPN MODEL OF IOTP TRANSACTIONS

A CPN model has been constructed for all six IOTP transactions with cancellation occurring at some certain points [10]. We restructure our previous model to include a new hierarchical level associated with transactions, which then re-uses modular exchange mechanisms in IOTP. The CPN model is also extended to include procedures of error handling and arbitrary transaction cancellation.

4.1 Net Structure

Our revised CPN model has 31 pages (modules) organised into four hierarchical levels, as shown in Figure 1. An arc between two pages indicates that the destination page is a submodule (subpage) of the source page. The four-level hierarchy of the CPN model provides a logical structure that can be validated against RFC 2801.

![Figure 1: Hierarchy page for the CPN model.](image.png)

The first (top) level has one page named IOTP_Overview. The second level comprises four pages: Consumer, Merchant, PHandler and DHandler, corresponding to four trading roles. We refer to these pages as trading role pages. The Merchant Customer Care Provider, currently not used in any transactions, is not modelled.

Each trading role page has a set of subpages specifying the possible IOTP transactions for that trading role. All these subpages, which we call transaction pages, constitute the third level of the model. The initial letter of a trading role is used as a suffix of the name of transaction pages modelled for that trading role. For example, the page Consumer has four subpages modelling six transactions for the Consumer. The three transactions Deposit, Withdrawal and Refund use the same procedure and therefore are modelled on one page named Deposit_C/Withdrawal_C/Refund_C.

Each transaction page is further decomposed into a set of subpages modelling the document exchanges that are used to construct the transaction as well as error handling and cancellation procedures. These subpages, called exchange level pages, constitute the fourth level of the model. For example, the page Purchase_C has six subpages, modelling the six document exchanges used to implement the Purchase transaction for the Consumer, and two others for error handling and cancellation. Since a document exchange involves two trading roles, we have modelled each exchange as a pair of pages - one for each trading role. For example, the pages Authenticatee and Authenticator represent an Authentication exchange where the Consumer is authenticated by the Merchant.

4.2 Global Declarations

Global declarations comprise colour sets, variables, constants and functions used in CPNs. Unfortunately, space limitations preclude their inclusion in this paper.

4.3 The Top Level Page

Figure 2 depicts the top level page: IOTP_Overview.
There are four substitution transitions (transitions with a HS-tag) named Consumer, Merchant, Payment Handler and Delivery Handler, each of which represents IOTP’s procedures for each of the corresponding trading roles. A substitution transition is a macro defined by another CPN page (subpage), e.g., the substitution transition Consumer or Merchant is linked to the trading role page with the same name. There is only one place named Transport in Figure 2. It is typed by the product set TRxTRxMQ, modelling the transport medium over which the four trading roles communicate. TR is the set of 4 trading roles and MQ defines a first-in first-out message queue. Places associated with a substitution transition need to be assigned to places with the same function on the subpage. For example, the place Transport is assigned to the place with the same name on both the Consumer and Merchant pages.

4.4 Trading Role Pages

Both Consumer and Merchant are involved in all six transactions. Figure 3 depicts the two corresponding trading role pages. Each transaction is abstractly represented by a substitution transition which has the same name as the transaction. Each place whose name starts with C or M is typed by the product set StaxBfr, which models the state (Sta) of the Consumer or Merchant with its message retransmission buffer (Bfr) in each of the six transactions. For brevity, we refer to these places as C_places or M_places. Place Transport on each page in Figure 3 has been described above.

The page Consumer in Figure 3 (a) has only one ordinary transition, InitializeTr_C, used to initiate an appropriate transaction for the Consumer on receiving the first IOTP message from the Merchant. It has both input and output arcs associated with place Transport. The input arc has an inscription specifying the first IOTP message (represented by (TransRefBlk(trtype):m, id)) from the Merchant. The output arc indicates that processing a Transaction Reference Block, which conveys transaction type information (TransRefBlk(trtype)) in an IOTP message, is complete, and thus the block is removed from the input message buffer. Similarly, the token returned by function InitialState is added to one of the six C_places on occurrence of transition InitializeTr_C, modelling the corresponding transaction has been initiated at the Consumer side.

4.5 Transaction Pages

Each transaction page specifies a valid combination of document exchanges to construct a transaction for a trading role. All these pages have a similar structure. Hence, we illustrate our approach with the Purchase transaction, which is the most complex in that it involves four trading roles and uses all six document exchanges. It consists of an optional Authentication exchange, followed by either a Brand Dependent Offer or Brand Independent Offer, and then either a Payment exchange only, or a Payment followed by a Delivery exchange, or a Payment-and-Delivery exchange. Figure 4 illustrates the four transaction pages used to model a Purchase transaction for each of the four trading roles, respectively.

The Consumer is the only trading role involved in the whole transaction procedure. All six document exchanges are specified by the first six substitution transitions on page Purchase_C. The Merchant may be engaged in an Authentication exchange and either of the two Offer exchanges in a payment-related transaction, which are represented by the first three substitution transitions on the Purchase_M/Deposit_M/Withdrawal_M/Refund_M/ValExTr_M page. The Payment Handler may be involved with a Payment or a Payment-and-Delivery exchange, as specified by the first two substitution transitions on page Purchase_P. The Delivery Handler can only conduct a Delivery exchange, as represented by the top substitution transition on the Purchase_D page. Finally, two substitution transitions Error Handling and Cancellation on each of the four pages in Figure 4, correspond to the error handling and transaction cancellation procedures.
4.6 Exchange Level Pages

Each document exchange is modelled according to its message processing guidelines in Section 9.1.1 of RFC 2801, which do not involve error handling and transaction cancellation. Procedures for error handling and transaction cancellation are modelled based on their respective descriptions in Sections 4 and 3.10 of the RFC. The same modelling approach has been used for the six document exchanges and the error handling and cancellation procedures. We therefore only give a detailed account of how the Authentication document exchange has been modelled.

The Authentication document exchange can be used for a standalone Authentication transaction, or by the Merchant to authenticate the Consumer at the beginning of a payment-related transaction. Figure 5 shows a possible scenario of an Authentication exchange, where the Merchant acts as the Authenticator and the Consumer as the Authenticatee.

Two exchange level pages named Authenticatee and Authenticator, as shown in Figure 6, model the Authentication exchange between the Consumer and Merchant within an IOTP transaction. Each transition represents the event of sending, receiving or holding an IOTP message. Two places, C_Autee in Figure 6 (a) and M_Autor in Figure 6 (b) are linked to the C_place and M_place on the corresponding transition pages. For example, with a Purchase transaction, C_Autee is linked to place C_Purchase in Figure 6 (a), and M_Autor to place M_PayTr in (b). Below, we illustrate how Authenticatee and Authenticator pages in Figure 6 reflect the Authentication document exchange in Figure 5.

Initially, the place Mer_Autor on page Authenticator has a marking ((READY,trtype,0,0),[]). Transition SndAuthReq is enabled. On its occurrence, the Merchant initiates an Authentication exchange by sending an Authentication Request Message (represented by [TransRefBlk(trtype), AuthReqBlk] with message identifier 1) to request authentication information (e.g., user account and password) from the Consumer. This corresponds to event 1 in Figure 5. After sending the message, the Merchant enters the LISTEN state waiting for a response from the Consumer, updates the sent message identifier (from 0) to 1, retains the same message retransmission counter value (0), and stores the message content [AuthReqBlk] in the message retransmission buffer. The variable trtype can be bound to any transaction type, modelling the Authentication exchange can occur in any transaction.

When the Authentication Request Message reaches the Consumer, transition InitiateTr_C on page Consumer (Figure 3 (a)) occurs first. As a result, the place Con_Autee on page Authenticatee will have a marking ((READY,trtype,0,0),[]). Transition RcvAuthReq is enabled. On its occurrence, the Consumer starts the AUTH exchange, enters the WAIT state waiting for the message to be processed, and records the received message identifier (1). The occurrences of the two above transitions represent event 2 in Figure 5.

Next, transition SndAuthRsp on page Authenticatee can occur. It models event 3 in Figure 5, where the Consumer replies with an Authentication Response Message (e.g., containing the requested user account and password) to the Merchant. Note that the Consumer may detect errors when processing the Authentication Request message and send a message reporting the error to the Merchant before generating the Authentication Response Message. The error handling operations will increment the sent message identifier and overwrite the message retransmission buffer. We use the variable sid for the sent message identifier and sm for the message retransmission buffer. After sending the Authentication Response Message, the Consumer enters LISTEN, increments the sent message identifier to sid+1, and updates the message content to [AuthRespBlk] in the retransmission buffer.

Event 4 in Figure 5 is modelled by transition RcvAuthRsp on page Authenticator. The guard [id<>rid] on RcvAuthRsp indicates it can be enabled only on receiving a non-duplicate Authentication Response Message. Before receiving this message, the Merchant may resend the previous message upon timeout, or receive a message reporting an error from the Consumer. Again, this relates to error handling, and accordingly we use two variables rc for the message retransmission counter and rid for the received message identifier. After receiving the Authentication Response Message, the Merchant enters state WAIT and updates the received message identifier to id.

Two transitions SndAuthSts and HoldAuthSts on page Authenticator are both enabled, but only one of them can occur. SndAuthSts models the Merchant sending an Authentication Status Message to the Consumer in an Authentication transaction, corresponding to event 5 in Figure 5. After SndAuthSts occurs, the Merchant enters the COMPLETED state and the transaction terminates. HoldAuthSts, with guard [trtype<>Authentication], models the Merchant holding an Authentication Status Message that needs to be combined with the first message in the following Offer exchange in a payment-related transaction. After HoldAuthSts occurs, the Merchant enters the state FINISHED and moves onto the next Offer exchange.
Finally, transition RcvAuthSts on page Authorizatee can occur. Its occurrence with variable $m$ bound to the empty list $[\]$ models the receipt of a standalone Authentication Status Message, representing event 6 in Figure 5. Otherwise, if $m$ is not the empty list, RcvAuthSts models the receipt of an Authentication Status Message that is combined with the first message in the following Offer exchange.

5. INITIAL STATE SPACE ANALYSIS

We use state space analysis to validate the IOTP CPN model and to investigate the functional behaviour of IOTP transactions. In state space analysis we generate all execution paths and store them as a directed graph (called an occurrence graph or state space). Nodes in the state space represent states (markings), and arcs represent occurrences of transitions.

Each of the six IOTP transactions can be analysed by giving the appropriate initial marking to place Initialise on page Merchant (Figure 3 (b)). To analyse a particular transaction, Transaction is set to that transaction. For example, if Transaction is set to Purchase, the CPN model can be executed to capture all possible executions of a Purchase transaction and to generate the state space for the transaction. For each transaction, there are also different cases we can analyse by changing the maximum value of the retransmission counters (denoted by $RC_{\text{max}}$) for each of the four trading roles. Once the number of retransmissions reaches $RC_{\text{max}}$, the transaction will be cancelled. Thus we obtain analysis results for IOTP under different configurations. For example, the full state space of the Purchase transaction, with $RC_{\text{max}} = 1$ for each trading role, has 29856 nodes and 98389 arcs, and is generated in 841 seconds on a 700MHz Intel Pentium computer with 512 MB RAM.

The state space can be used to investigate properties of the system such as absence of deadlocks. A dead marking of a CPN model is a marking with no enabled transitions. The absence of deadlocks can be proved by demonstrating that each dead marking of the state space represents a desired terminal state of the transaction. Analysing the six IOTP transactions with $RC_{\text{max}} = 1$ has revealed two problems in the current design of all payment-related transactions in IOTP. Both are related to transaction termination and are due to lack of synchronisation between the trading roles. One could lead to an inventory problem for the Merchant caused by the Merchant not being informed that a transaction has been cancelled by the Consumer or the Payment or Delivery Handler. This was also detected when analysing the Purchase transaction in our previous specification of IOTP [10]. The other may result in the Consumer not being able to obtain a refund due to cancellation only occurring at the Consumer while the other trading roles believe that the transaction has been successfully completed. This is a new problem discovered in this specification of IOTP.

6. CONCLUSIONS

This paper has illustrated the first complete formal specification of the main transactions of the Internet Open Trading Protocol, a complex e-commerce protocol being developed by the Internet Engineering Task Force. These are the Authentication transaction and the five payment-related transactions. The specification is written in hierarchical Coloured Petri Nets.

This new specification is a significant improvement over our previous partial specifications [8-10]. Firstly, we have introduced a new hierarchical level associated with transactions that matches the structure of IOTP and allows re-use of modular message exchange mechanisms, known as document exchanges. Secondly, the previous model has been extended to include IOTP’s procedures for error handling and arbitrary transaction cancellation. The improved specification is at a sufficiently detailed level to allow its functional behaviour to be analysed thoroughly. The paper has summarised some initial findings that demonstrate a lack of synchronisation of trading roles when transactions are cancelled. We believe that this approach will allow errors to be removed from the IOTP definition, will support very rapid and less error prone implementations of IOTP, and will be amenable to automatic code generation.

7. ACKNOWLEDGMENTS

This work was carried out with financial support from a University of South Australia and Divisional Small Grant. We acknowledge the assistance of Dr Lars Michael Kristensen in earlier work in this project.

8. REFERENCES


