Formal Validation for Software Modeling

Baojun Tian[1], Yanlin Gu[2]

[1] College of Information Engineering, Inner Mongolia University of Technology, Huhhot, 010080, China

[2] Vocational College, Inner Mongolia University of Finance and Economics Huhhot, 010051, China

Abstract

Currently,modeling for software is mostly semiformal, such as UML(Unified Modeling Language).The main problem is difficult to analyze semantics and verify correctness for a vital system.CPN(Colored Petri Net)as modeling and verification method provides formal semantics and a number of analysis techniques and tools.This paper presents an approach of formal modeling and validation for software process, which transforms UML models based on RUP(Rational Unified Process) to CPN,and uses CPN tools to investigate the behaviour of modelled system.

Keywords: *Modeling*, *CPN*, *Formal, UML.*

1. Introduction

UML as a semiformal modeling language is widely applied to many large and complex software systems. Although UML's grammar is formal, natural language is used by its semantics,which makes it only to do static modeling not to formally analyze and verify the models. Formal methods(e.g.CPN)that have precise mathematic semantics and automated verification tools are used for both static structural analysis and dynamic behaviour analysis. CPN as a graphically oriented modeling language has functions of ordinary Petri Nets and high level programming languages providing the primitives for the definition of data types and describing data manipulation,which makes it is suitable for modeling for complex systems. The CPN language allows the model to be represented as a set of modules,which can make it to be organized as a set of hierarchically related modules, i.e. HCPN(Hierarchical Coloured Petri Net).CPN model is one of best ways for analysis and design of non-deterministic and concurrent systems. In the paper, I present a detailed statement of applying CPN to describe the semantics of UML diagram, and also do analysis and verification of object-oriented system[4].

2. Related Theory Knowledge

2.1 Formal Definition of HCPN

HCPN makes it possible to relate a amount of individual CPN to each other in a formal way.This is similar to the situation in which a programmer constructs a large-scale program from a set of modules and subroutines[8].

A Hierarchical Coloured Petri Net is a tuple HCPN=(S, SN, SA, PN, PT, PA, FS, PT, PP)satisfying the following requirements below[1][2]:

(i)S is a finite set of pages such that:

Each page s∈S is a non-hierarchical CPN:

$(\sum s, Ps, Ts, As, Ns, Cs, Gs, Es, Is)$.

The sets of net elements are pairwise disjoint:

$\forall$ s1,s2∈S:[s1≠s2 $(Ps1 \cup Ts1 \cup As1) \cap (Ps2 \cup Ts2 \cup As2) = \Phi$].

(ii)SN⊆T is a set of substitution nodes.

(iii)SA is a page assignment function. It is defined from SN into S such that:

No page is a subpage of itself: $\{s0s1 \ldots sn \in S^* \mid n \in N+ \wedge s0 = sn \wedge \forall\ k \in 1 \ldots n: sk \in SA(SNsk-1)\} = \Phi$

(iv)PN⊆P is a set of port nodes.

(v)PT is a port type function.It is defined from PN into{ in,out,i/o,general }.

(vi)PA is a port assignment function.It is defined from SN into binary relations such that:

Socket nodes are related to port nodes: $\forall$ t∈SN: $PA(t) \subseteq X(t) \times PNSA(t)$.

Socket nodes are of the correct type:

$\forall$ t ∈ SN $\forall$ (p1,p2) ∈ PA(t):[PT(p2 ≠ general ST(p1 ,t)=PT(p2)].

Project number: science research project of Inner Mongolia University of Technology(ZD201317).

Related nodes have identical colour sets and equivalent initialization expressions:

$\forall$ t$\in$SN$\forall$ (p1,p2)$\in$PA(t): [C(p1)=C(p2)$\wedge$I(p1)< > = I(p2)< >].

(vii)FS$\subseteq$Ps is a finite set of fusion sets such that:

Members of a fusion set have identical colour sets and equivalent initialization expressions:

$\forall$ fs$\in$FS: $\forall$ p1,p2$\in$fs:[C(p1)=C(p2)$\wedge$I(p1)< > = I(p2)< >].

(viii)FT is a fusion type function.It is defined from fusion sets into {global,page,instance} such that:
Page and instance fusion sets belong to a single page:

$\forall$ fs$\in$FS: [FT(fs)$\neq$global $\exists$ s$\in$S: fs$\subseteq$Ps].

(ix) PP$\in S_{MS}$ is a multi-set of prime pages.

2.2 Dynamic properties of coloured petri net

When it is represented by a CPN, UML models can be analyzed and verified by using dynamic properties.

(1) Boundedness properties[1]
Definition 1:Let a set of token elements X$\subseteq$TE (we use TE(p) to denote the set of token elements that correspond to a given place p)and a non-negative integer n$\in$N be given.
(i) n is an upper bound for X iff:
$\forall$M$\in$[M0>: | (M|X) |$\leq$n.
(ii) n is a lower bound for X iif:
$\forall$ M$\in$[M0>: | (M|X) |$\geq$n.
The set X is bounded iff it has an upper bound.
(2) Liveness Properties[1]
Definition 2:Let a marking M$\in$M and a set of binding elements X$\subseteq$BE(we use BE(t) to denote the set containing all those binding elements which correspond to a given transition t) be given.
(i) M is dead iff no binding element is enabled,i.e.,iff:
$\forall$x$\in$BE:$\neg$M[x>.
(ii) X is dead in M iff no element of X can become enabled,i.e.,iff : $\forall$M' $\in$[M> $\forall$x$\in$X: $\neg$M'[x>.
(iii) X is live iff there is no reachable marking in which it is dead,i.e.,iff:
$\forall$M' $\in$[M0> $\exists$M'' $\in$[M'> $\exists$x$\in$X: M''[x>.
(3) Fairness Properties[1]

We use ENx(σ)= $\sum_{i\in N+} ENx,i(\sigma)$, OCx($\sigma$)= $\sum_{i\in N+} OCx,i(\sigma)$

Definition 3:Let X$\subseteq$BE be a set of binding elements and $\sigma\in$OSI an infinite occurrence sequence[1].
(i) X is impartial for σ iff it has infinitely many occurrences,i.e.,iff: OCx(σ)=∞.
(ii) X is fair for σ iff an infinite number of enablings implies an infinite number of occurrences,
i.e.,iff:ENx(σ)=∞ OCx(σ)=∞.
(iii) X is just for σ iff a persistent enabling implies an occurrence,i.e.,iff:
$\forall$i$\in$N+: [ENx,i(σ)$\neq$0 $\exists$ k$\geq$i: [ENx,k(σ)=0 $\vee$OCx,k(σ)$\neq$0]].
(4) Home Properties[1]
Definition 4:Let a marking M$\in$M and s set of marking X$\subseteq$M be given:
(i) M is a home marking iff: $\forall$ M' $\in$[M0>:M$\in$[M'>.
(ii) X is a home space iff: $\forall$ M' $\in$[M0>:X$\cap$[M'>$\neq\Phi$.
We use HOME to denote the set of all home markings.

3. A Formal Approach For UML Diagram

A UML class diagram (includes two kinds of classes,i.e. general class and association class),which describes static structure and is a basis of system implementing.A UML dynamic diagram(e.g.activity diagram,sequence daigram)describing behaviour and characteristic must be based on the class diagram.So,UML class diagram plays a key role in the system modeling.However,UML is a semiformal modeling language,and it is difficult to formally analyze and verify the models.While CPN has a graphical representation,a number of modeling techniques and a well-defined semantics allowing formal analysis. Abviously,for obtaining increased confidence in the correctness of the model,it is necessary to transform UML diagran to CPN model.

Before transforming,two works must be done[9]:

- A association class between two classes need be created,which only includes attributes.The attributes are got from UML activity diagram.
- A general class including attributes and operations need be transformed into a whole class that only includes operations and one or more part classes that only include attributes.

Below,by using Hierarchical Coloured Petri Net an algorithm of formal modeling is given:

- A substitution transition presents a class which has two and more operations.Each operation corresponds with a transition in substitution transition subpage.The socket place of the substitution transition is related with the port place in the subpage.The contents of its arc expressions are decided by the message that is transmited between different action states in UML activity diagram.
- A general transition presents a class that only has one operation.
- A place presents a class that only has static attributes that correspond with coloured sets of the place.
- The direction of a directed arc in HCPN is determined by UML class diagram.

4. Formal Analysis and Validation

4.1 Modeling Based on RUP

RUP is an iterative software development process framework(from requirement to implementation).It may be applied to a variety of different types of software systems, applications, organizations and projects.At present UML combined with RUP implements best software practices.

Now, I present an example that is a goods ordering system to explain how to formalize UML diagram and use CPN tools to verify the behaviour of the modelled system[6].

4.1.1 Requirement modeling

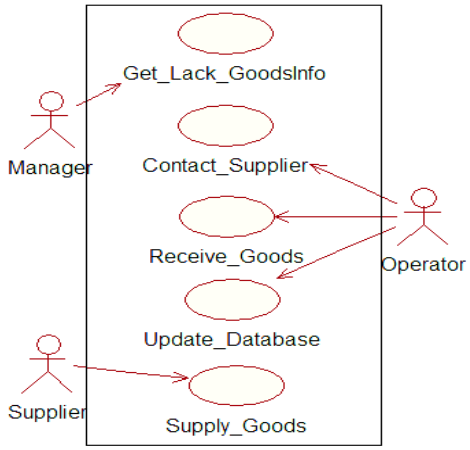

Fig. 1. a use case diagram of ordering system

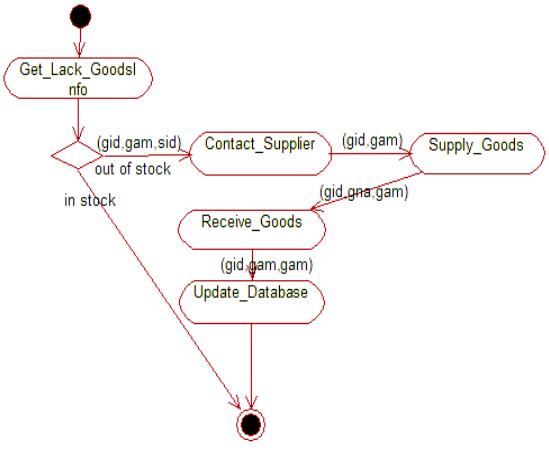

Fig. 2. an activity diagram of ordering system

4.1.2 Design modeling

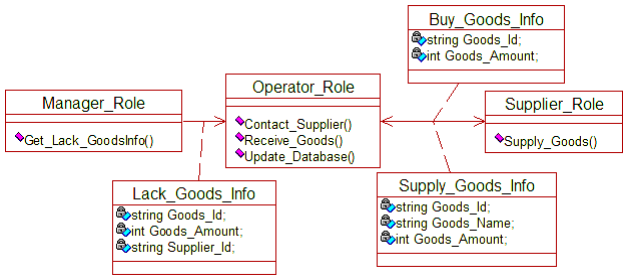

Fig. 3. a class diagram of ordering system

4.2 Formal Modeling and Validation

According to the transforming algorithm,a HCPN is given below.the colour sets and variables are described by CPN ML[5].

```
colset Goods_Id=string;
colset Goods_Amount=int;
colset Supplier_Id=string;
colset Goods_Name=string;
colset Lack_Goods_Info=product Goods_Id*
  Goods_Amount*Supplier_Id;
colset Supply_Goods_Info=product Goods_Id*
  Goods_Name*Goods_Amount;
colset Buy_Goods_Info=product Goods_Id*
  Goods_Amount;
var gid: Goods_Id;
var gna:Goods_Name;
var gam:Goods_Amount;
var sid:Supplier_Id;
```

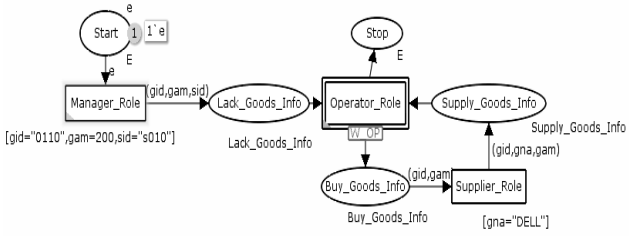

Fig. 4. the prime HCPN model of ordering system

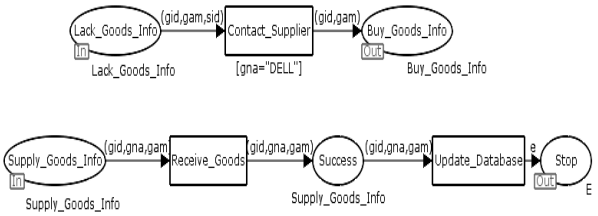

Fig. 5. subpage correspond with substitution transition Operator_Role

Full state spaces represent all possible executions of the model being analysed.The state space of a CPN model make it possible to verify,i.e., prove in the mathematical sense of the word that the model possesses a certain formally specified property[3][7].

```
                    Boundedness Properties
------------------------------------------------------------------------
                    Best Integer Bounds
                                 Upper          Lower
HCPN'Buy_Goods_Info 1            1              0
HCPN'Lack_Goods_Info 1           1              0
HCPN'Start 1                     1              0
HCPN'Stop 1                      1              0
HCPN'Supply_Goods_Info1 1        1              0
W_OP'Success 1                   1              0
Best Upper Multi-set Bounds
HCPN'Buy_Goods_Info 1 1`("0110",200)
HCPN'Lack_Goods_Info 1 1`("0110",200,"s010")
HCPN'Start 1                1`e
HCPN'Stop 1                 1`e
HCPN'Supply_Goods_Info 1 1`("0110","DELL",200)
W_OP'Success 1              1`("0110","DELL",200)
Best Lower Multi-set Bounds
HCPN'Buy_Goods_Info 1 empty
HCPN'Lack_Goods_Info 1 empty
HCPN'Start 1                empty
HCPN'Stop 1                 empty
HCPN'Supply_Goods_Info 1 empty
W_OP'Success 1              empty
                      Home Properties
------------------------------------------------------------------------
Home Markings
[6]
                    Liveness Properties
------------------------------------------------------------------------
Dead Markings
[6]
Dead Transition Instances
None
Live Transition Instances
None
                    Fairness Properties
------------------------------------------------------------------------
No infinite occurrence sequences.
```

Fig.6. Rport of CPN state space

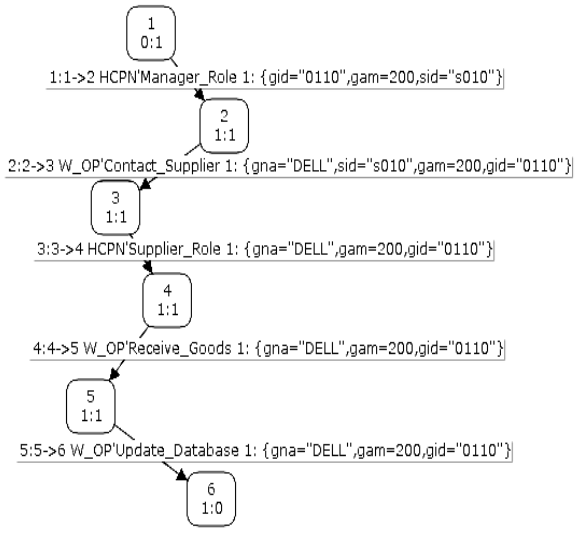

Fig. 7. State space graph

The state space is calculated by CPN state tools that use a state construction algorithm. A node of the directed graph correspond to reachable marking,and an arc corresponds to occurring binding elements.The best upper(lower) integer bound of a place specifies the maximal(minimal) number of tokens residing on a place in any reachable marking. For example,the best upper integer bound of the place Buy_Goods_Info is 1 that means at most there is one token,and there exists reachable markings where there is a token on it.This is what the modeler would expect,since Buy_Goods_Info is always supposed to contain a single token with a colour corresponding to the data which has been received.The other places is similar to it.Fig. 6 shows there is a single home marking that has the node number corresponding to node 6 of fig. 6,which means that it is possible to have occurrence sequences that can be extended to reach Home Marking. The fact that node 6 is the only Dead Marking clarify that the ordering system as specified by the CPN model is correct,i.e. if execution terminates we have the correct results. Meanwhile, because node 6 is also a home marking it is always possible to end the system with the correct result.Finally, Figure 6 shows that no dead transitions exist,which indirectly tells us that UML model has no redundant methods.

5. Conclusion

As an important modeling approach,UML has been successfully applied in many fields of software engineering.Because of lackness of formal semantics,UML is diffficult to verify the correctness of models,which may cause disastrous consequences for a vital sysytem.CPN has precise mathematic semantics and automated verification tools.CPN combined with UML is a better modeling approach,which not only

easily describes the software but also detects errors and obtains increased confidence in the correctness of the model ,and thereby the system.

References

[1] Kurt Jensen,Coloured Petri Nets - Basic Concepts,Analysis Methods and Practical Use,vol.1, pp.89-155,1992.
[2] Sun, Xiaoxing, A CPN based method for aspect-oriented modeling and analysis of fault tolerance, Advanced Materials Research, pp.891-898, 2012.
[3] Kurt Jensen, Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems,International Journal on Software Tools for Technology Transfer, Vol.9, pp.213-233,2007.
[4]Ali Kamandi, Mohammad Abdollahi Azgomi, Ali Movaghar,Transformation of UML Models into Analyzable OSAN Models,Electronic Notes in Theoretical Computer Science (ENTCS),vol.159, pp.3-22, May, 2006.
[5] Maja Pesic,Modelling work distribution mechanisms using Colored Petri Nets,International Journal on Software Tools for Technology Transfer, Vol.9, pp. 327-352,2007.
[6] Noguera, Manuel, Ontology-driven analysis of UML-based collaborative processes using OWL-DL and CPN, Science of Computer Programming,Vol.75,no.8,pp.726-760,2010.
[7] Fan Haijun, Performance analysis based on UML and hierarchical colored petri, International Journal of Advancements in Computing Technology, Vol.4, no.23, pp.97-107,2012.
[8] Lars Michael Kristensen,Soren Christensen, Implementing Coloured Petri Nets Using a Functional Programming Language, Higher-Order and Symbolic Computation, vol.17,NO.3,pp.207-243,Sep. 2004.
[9] Tian Baojun, The Research of Formalizing UML Diagram Based on HCPNs, Proceedings 2010 IEEE International conference on Software Engineering and Service Science,pp.523-526.