A Temporal-Rule Based Verification System for Business Collaboration Reliability

Haiyang Sun[1]
Jian Yang[1]
Weiliang Zhao[1]

Abstract: Based on the temporal rules defined for the business processe participating in a business collaboration, we present an implementation for a system called *Ti*CoBTS to verify the reliability of the business collaboration.

1 Introduction

Collaborative business processes can easily become unreliable in a loosely-coupled environment where partners' processes are interacted in a peer based fashion without central coordination. Temporal errors can occur anywhere in collaborative processes which affect the normal execution of business collaboration. Therefore a system to properly model and manage business collaboration in a peer-based execution environment and provide verification methods for collaboration reliability in terms of temporal rules is needed. The purpose of this demo is to demonstrate the functionality of a verification system named **Ti**med **Cho**reographical **B**usiness **T**ransaction net **S**ystem (*Ti*CoBTS). It has been developed with the following features: (1) *Ti*CoBTS is developed based on Timed Hierarchical Colored Petri Net [1, 2] for individual business participants to specify the collaboration in a peer-based fashion. (2) Temporal rules are captured and operated within and across organizations. (3) An algebraic verification approach is developed based on the temporal rules to verify collaborative processes reliability. By itself, the problem of collaborative business reliability verification without global coordination that we address is challenging, and the method we provide here for temporal rule based verification is unique and notably differs from any existing works, which focus on either workflow-type business process management within a single organization [3] or manage collaboration from a global point of view [4].

2 *Ti*CoBTx-Net and Its Temporal Rules

A *Ti*CoBTx-Net is constructed for each organization to understand and analyze its own behavior as well as those of its business partners. In a business collaboration, each organization's internal tasks can not be seen by other organizations. However we can assume

[1] Department of Computing, Macquarie University, Sydney, NSW 2109, Australia
{hsun,jian,wzhao@ics.mq.edu.au}

that its service interfaces and business protocols can be publicly observed by others. Based on this assumption, five subnets are defined in the *Ti*CoBTx-Net of a participating organization, three of which are *Execution subnet* (Exe), *Abstraction subnet* (Abs) and *Communication subnet* (Com) corresponding to the internal business task, service interface and business protocol respectively. These subnets are linked by the **Refinement Functions** which are used to specify the stratified structure of collaborative business processes. The rest two subnets are *External Communication subnet* (ExCom) and *External Abstract subnet* (ExAbs) which are the projections of its collaborators' business protocols and service interfaces. In a *Ti*CoBTx-Net, Application Oriented-Tokens (AO-Token) are transferred among subnets to form the message flow. The duration when the AO-Token stay in a place is specified as **message transfer time**; while the duration of tokens staying in a transition is defined as **message execution time**. The detailed discussion on the net structure and execution policy can be found in [5].

Two types of temporal rules are used in *Ti*CoBTx-Net: *Generic Temporal Rules* and *Application Domain Specific Rules*. (1) The *Generic Temporal Rules* are used by the system to restrict token movement within and across various sunbets. For example, the real message execution time in an event (transition) should not be more than the defined maximum message execution time. Another example is that the start time of a transition in Abs-subnet should not be later than the start time of a transition in Exe-subnet, where such two transitions are linked by the refinement function. The reason is that the start time of an internal task which supports specific service should NOT commence to operate earlier than the start time of the supported service. (2) The *Application Domain Specific Rules* specify the time constraints designed by users for the application. For instance, *Application Domain Specific Rules* can be defined to present that the latest finish time of an event T_j should be 4 days later than the finish time of T_{j-2} when T_{j-3} starts after 5:00am. Note, *Application Domain Specific Rules* are not allowed to be conflicted with *Generic Temporal Rules*. Based on the temporal rules defined above, a reliability property named as **time-embedded dead marking freeness** is defined to verify the business collaboration reliability in terms of temporal rules in *Ti*CoBTx-Net. See [5] for the detailed algebraic verification.

3 Demonstration of *Ti*CoBTS

*Ti*CoBTS simulates the execution of business collaboration, and provides a mechanism to detect the occurrence of the violation of temporal rules. This implementation is developed in JAVA/SWT, under the Eclipse environment. *Ti*CoBTS consists of three components (See Fig.1): (1) **Business Process Editor**: it is used to input *Ti*CoBTx-Net and relevant temporal rules either through *Process Choreography Editor* or by *BPEL engine*. The *Exception Handling Editor* is used to describe conflict specifications and corresponding exception handling in addition to time constraint violations discussed in the paper. Considering BPEL as the de facto standard, *BPEL engine* is used to take BPEL code and convert it into our *Ti*CoBTx-Net. In order to simplify and reuse the configuration, *Token Games Warehouse* is used to store the token movement rules. In addition, it saves the status of execution in a consistent storage so that the simulation can be paused and continued or rolled back if necessary. (2) **Simulator**: it performs computation and presents verification result. The *Modelling* works on tokens' movements between different subnets among various organizations, accompanied by the calculation and verification processing. In the process of *verification*, user can choose to execute the simulation in either of two modes: `Normal Mode`, which means that tokens move continuously, or `Single-Step Mode`, which allows the user to capture every single step of token movements. Once the time constraint violation is detected, the *Time Conflict Viewer* is enabled to display violation content in the form of alerts or query, or in the form user previously defined. (3) **External Interface**: it allows the systems to be integrated with or used by other external systems. Graphic and simulation model interfaces are provided to exchange data with other systems. Additionally, the simulator can also be accessed by a set of APIs after authorization checking.

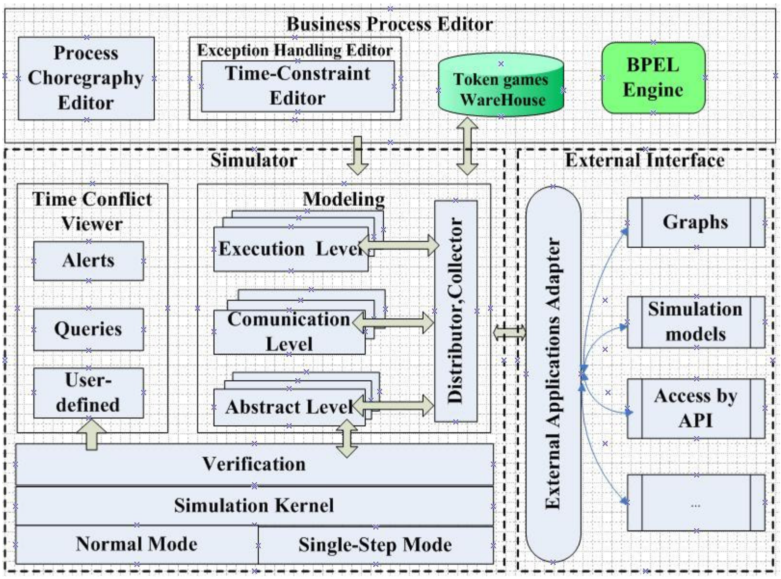

Fig.1 System Architecture of ***Ti*****CoBTS**

Here we will demonstrate how to use the *Ti*CoBTS. (i) **Construct *Ti*CoBTx-Net**: the main interface of *Ti*CoBTS consists of five areas: *Menus*, *Main drawing*, *Information*, *Matrix computing*, and *Monitor*. In order to construct a *Ti*CoBTx-Net for a specific organization, the user need to drag place (depicted as a circle) and transition (depicted as a bar) icons to the *Main drawing*, followed by drawing arrows to track the flows and message exchanges. (ii) **Design Temporal Rules**: time constraints for a place--Minimum and Maximum duration span--can be set up in system. Besides, the global start time is defined in the initial place.

The time constraints for a transition are specified by duration as well as earliest start time, earliest finish time, latest start time and latest finish time, which can be further elaborated by two options by clicking on "more" button. In *option 1* area, each line represents an *Application Domain Specific Rule* by indicating condition, expression and its relationship at the corresponding text boxes. Multiple rules can be joined together to become a composite *Application Domain Specific Rule* by selecting the type of the logical operators in the front pull-down menu. *Option 2* enables the advanced user to specify more complex rules and exert more control over the combination of multiple conditions or expressions. In addition, temporal rules can also be formulated in the pathway of BPEL engine. (iii) **Execution of *Ti*CoBTx-Net**: firstly, initial and final places are defined by shortcut keys. Then there will be an AO-Token (depicted as a small hollow dot) in the initial place. Lastly the user can choose `single-step mode` or `normal mode` to initiate a verification. Every screenshot of tokens' movements is captured in the *drawing window* with the time conflicts being detected by the *monitor window*. The accompanying matrixes representing the status of *Ti*CoBTx-Net are simultaneously illustrated in the *matrix computing window*. Moreover, the attributes of the selected component in every step are displayed in the upper right-hand *information window* during the tokens' movements. (iv) **Detection of Temporal Rules Violation**: The system will verify business collaboration reliability in terms of temporal rules by checking reliability property-**time-embedded dead marking freeness** in *Ti*CoBTx-Net. Relevant matrix computation is used to calculate the status of each step of *Ti*CoBTx-Net execution.

4 Conclusion

In this paper, we demonstrate a system which is used to (1) model collaborative business processes from a single organization's point of view and (2) verify business collaboration reliability based on temporal rules.

References

[1] J. Tsai, S. Yang, Y. Chang, E. Juan.: Verifying Timing Properties for Distributed Real-Time Systems Using Timing Constraint Petri Nets, COMPSAC, 1996.

[2] J. Toussaint, F. Simonot-Lion, J. Thomesse.: Time Constraints Verification Methods Based on Time Petri Nets, FTDCS 1997.

[3] F.L. Tiplea, and G.I. Macovei.: Timed Workflow Nets, SYNASC, 2005.

[4] B. Benatallah, P. Chrzastowski-Wachtel, R. Hamadi, M. O'Dell, A. Susanto.: HiWorD: A Petri Net-Based Hierarchical Workflow Designer. In Proceedings of ACSD, pp. 235-236, 2003.

[5] H. Sun, J. Yang.: Exploiting CoBTx-Net to Verify the Reliability of Collaborative Business Transactions, in Proceedings of IEEE APSCC, pp.415-422, 2007.