Abstract—Web service choreography has two points of view, public and local. The observable behavior of collaborating services to obtain the common goal based on WS-CDL language is called public choreography. The behavior of each participant services in the form of UML state machines to participate in collaboration is called local choreography. This paper proposes a mapping mechanism of WS-CDL (public view) to generate partner’s UML state machines (local view).

Keywords—WS-CDL, UML state machine, choreography, behavioral interface.

I. INTRODUCTION

SERVICE composition enables service reuse and accelerates application development. Three important concepts in the service composition are Orchestration, choreography and behavioral Interface. Choreography depicts the interaction between various services to gain a common goal. Based on definition of W3C, behavioral Interface shows how service does behave in choreography. In other words, it explains the observable behavior of a service relate with other parties which declare the dependencies between its interactions. An Orchestration view describes how a service manages the internal activities such as invocation and data transformation to supply its capabilities. An Orchestration model defines a set of “active rules” are executed to manage the behavior of a participant which is described in choreography model [1], [2], [11]. Web service choreography has two points of view, public and local. The observable behavior of collaborating services to obtain the common goal based on WS-CDL language is called public choreography. The behavior of each participant services in the form of UML state machines to participate in collaboration is called local choreography. This paper uses WS-CDL language as an input and generate UML state machine describes interaction behavior of each service. The remainder of this essay is as follows. Section 2 establishes the background of our work by giving a sketch of WS-CDL as well as UML state machine. Section 3 talks about some related work. Section 4 describes our Mapping approach. In order to understand better the modeling of partner’s behavioral interface involved in WS-CDL code with UML state machine, a case study is given in section 5. Finally, section 6 concludes routes for current and future work.

II. BACKGROUND

In this section, we give a short introduction into WS-CDL also about UML state machine before going into the details of our approach.

A. WS-CDL

WS-CDL [13] is an XML based language developed by W3C, proposed to describe the common and collaborative observable behavior of multiple services that need to interact in order to achieve some goal. Fig. 1 describes the WS-CDL package in detail.

Fig. 1 WS-CDL package details
In a WS-CDL choreography definition, the package element is the root of every choreography definition and is the container for all other WS-CDL elements. Choreography definition includes InformationType, token and tokenLocator elements. RoleType element shows the observable behavior of a special participating service and relationshipType element exactly contains two roles to define 1-to-1 relation among the participants. ParticipantType element is an entity playing a particular set of roles in the choreography. A ChannelType element elaborates on the role the receiver of the request or responded message and performed behavior on this channel. A WS-CDL package includes several choreography definition elements that are the core of collaboration. Within a choreography definition element there are activities called basic activities, control flow activities and work unit activity. The first type, basic activity, includes “assign”, “perform”, “interaction”, “noAction”, “silentAction”, and “finalize”. In order to create or change the value of the variables we use the value of another variable or expression in An assign activity. These variables are explained through the choreography definition, to gain information about objects in the choreography in away elaborated by their usage. Variables in WS-CDL are right to used in WS-CDL XPath 1.0 extension functions including getVariable (name, path, role), a predefined choreography is called by means of a performed choreography. The major element of all in WS-CDL is an interaction activity. It includes: the participants involved, the exchanged information and information exchanging channel. During an interaction the information sent or received via interaction activity. It includes: the participants involved, the exchanged information and information exchanging channel. These machines use new concepts such as hierarchical states, orthogonal (the combination of concurrent and complementary regions) and taking using of implementation of two Moor and Mealy patterns. Such specifications enable the UML state machines to modeling the behavior of each service participate in collaboration [2].

III. RELATED WORK

There are similar essays about choreography and orchestrations modeling using BPMN, UML activity for modeling web service orchestrations, sequence and statecharts they are as follows:

In [3] BPMN interface service is mapped to UML activity diagram and these diagrams using for generating WS-CDL code.

In [11] first, a central choreography process is developed regarding the business collaboration, second, this process transformed to a decentralized process across participating boundaries. Third, mediating the decentralized choreography process with the inner orchestration process. In [5], it shows how BPEL process definitions of parties involved in choreography can be semi automatic derived from the global WS-CDL model and what the limitations of such a derivation are.

In [15], presents a choreography example in four different visual modeling notations .They are BPMN, UML activity, sequence and statecharts then compares them with regarding the semantics of WS-CDL. The results are useful for establishing a reliable visual approach to modeling web services choreography.

IV. GRAPHICAL MAPPING

Mapping of the WS-CDL choreography to UML state machines for each participants is described here. Our approach pays attention to choreography elements based on control flow elements used to gain information about objects in the choreography, although they are not referred to in this mapping. The boundaries mapping regarding WS-CDL elements to UML state machine is briefly stated in Fig. 2.

The choreography element is define to map a UML state machine diagram including activities relating to the activities nested in the choreography definition element. The assign activity is mapped to a state called “Assign” in appropriate participants’ state machine and Entry Action in this state for doing assignment. The perform Activity, noAction and silentAction Activity is not mapped because they aren’t observable from participants’ view .The Finalize Activity mapped to final state in each state machines. The sequence activity is mapped to a sequence sub state machine containing a sequence of activities corresponding to the activities nested in the WS-CDL sequence activity for each participant. The Choice Activity is mapped to a block of option states displays activities corresponding to the activities nested in the Choice Activity. This block starts with a choice pseudo-state followed by the alternative activities. The parallel activity is mapped to a sub-state machine called “Parallel” corresponding to the
activities nested in the Parallel Activity. This block is started with a Fork pseudo-state and merged with a Join pseudo-state at the end.

V. CASE STUDY

In order to demonstrate our mapping, an instance is presented here which is a simplified form of WS-CDL.
example code from W3C draft but some pieces of information deleted. In this section we used the previously stated rules. In this example we have two participants: buyer and seller. Seller recommends certain price for the goods, now they are in a repetitive circle. The seller updates the price as the buyer demands the goods and this circle goes on till the time, the buyer decides to order the goods. Fig 4 shows the code of WS-CDL choreography, whereby the code does not include the declaration of the WS-CDL elements which are used to capture information about objects in the choreography. These factors include informationType, token, tokenLocator, roleType, relationType, participantType, channelType, relationships and variable Definitions. The main parts of the corresponding process described by using UML state machine notations are represented in Fig. 5 and 6. Using UML state machine notation has advantages over WS-CDL code including at least representing the behavior of each service notation explicitly.

VI. CONCLUSION AND FUTURE WORK

In this article we have pointed out a new mapping mechanism for modeling participant’s behavioral interfaces using UML state machine. These UML state machine diagrams are originated from the WS-CDL code. To model WS-CDL partner’s behavioral interfaces with UML state machine we have specified a mapping from WS-CDL to UML state machine for each participants. This mapping focuses on the choreography elements which depend on control flow. Elements, which are used to get information about objects in the WS-CDL choreography, were not addressed by this mapping. Future work develops a tool to generate abstract orchestration process using UML state machine from the WS-CDL code for each participant involved in WS-CDL process.
ACKNOWLEDGMENT

The author would like to thank the reviewers for their helpful comments and suggestions.

REFERENCES


