A domain specific language for complex dynamic decision making

BARAT, souvik, KULKARNI, vinay, CLARK, Tony <http://orcid.org/0000-0003-3167-0739> and BARN, Balbir

A DOMAIN SPECIFIC LANGUAGE FOR COMPLEX DYNAMIC DECISION MAKING

Souvik Barat
Tata Consultancy Services Research
Pune, India

Vinay Kulkarni
Tata Consultancy Services Research
Pune, India

Tony Clark
Sheffield Hallam University
Sheffield, United Kingdom

Balbir Barn
Middlesex University
London, United Kingdom

KEYWORDS
Organisational decision making, Conceptual model, Simulation model, Domain specific language.

ABSTRACT

Effective decision making of organisation requires deep understanding of various organisational aspects such as its goals, structure, business-as-usual operational processes in the context of dynamic, socio-technical and uncertain business envi-ronment. Decision making approaches adopt a range of modelling and analysis techniques for effective decision making. The current state-of-practice of deci-sion-making typically relies heavily on human experts using intuition aided by ad-hoc representation of an organisation. Existing technologies for decision mak-ing are not able to represent all constructs that are needed for effective decision making nor do they comprehensively address the analysis needs. This paper pro-poses a meta-model to represent organisation and decision artifacts in a compre-hensive, relatable and analysable form that serves as a basis for a domain specific language (DSL) for complex dynamic decision making. The efficacy of the pro-posed meta-model as regards specification and analysis is evaluated using a real-life scenario.

INTRODUCTION

Modern organisations need to meet their stated goals by adopting appropriate courses of action in increasingly dynamic environment subjected to a variety of change-drivers. It calls for the precise understanding of various aspects such as organisational goals, organisation structure, operational processes and past data (Shapira 2002). The large size of modern enterprises where the necessary information is both heterogeneous and distributed make compilation and analysis difficult. Furthermore, the socio-technical nature of enterprise (McDermott et al. 2013), inherent uncertainty (Rumsfeld 2011), non-linear causality in business interactions, and high business dynamics chiefly contribute towards organisational decision making being a complex dynamic decision making (CDDM) endeavour.

The common industry practice of organisational decision-making relies on human experts who typically use tools such as spreadsheets, word processors, and diagram editors (Locke 2009). Though adequate for capturing and collating the required information, these tools provide limited analysis support thus putting the decision making onus solely on intuition and interpretation by human experts. Moreover, these tools can only capture a static snapshot of enterprise and are largely devoid of the capability of supporting the dynamism. As a result, decision making with these tools tends to be time-, effort- and intellectually-intensive.

The state-of-the-art specification and analysis techniques approach the decision making problem in two ways namely, data-centric approach and model-centric approach. The data-centric approach makes use of sophisticated AI-based pattern recognition and predictive analysis techniques on relevant past data to predict future outcomes. This approach has worked well when the past data is comprehensive and the future is typically a linear extrapolation of the past. However, the two conditions are increasingly not being met for modern large enterprises thus leading to inappropriate decisions for emerging business contex (HBR 2014).

The model-centric approaches, in contrast, characterise the real organisation using representative models which span across a wide spectrum. At one extreme of the spectrum are enterprise specifications that provide a well-defined structure for the organisational aspects of interest and rely on a variety of visualisation techniques to help humans obtain the desired understanding of the organisation. For instance, ArchiMate (Iacob et al. 2003) is one such specification. At the other extreme of the spectrum are machine interpretable andr simulatable specifications such as i* (Yu et al. 2005), BPMN (OMG 2011), and System Dynamics (SD) model (Meadows and Wright 2008). Principally they adopt reductionist view (Beckermann 1992) to help analyse enterprises where the mechanistic world view holds. On the other hand, the languages and specifications advocating actor model of computation (Hewitt 2010) and agent-based systems (Macal and North 2010) support emergentism (O'Connor and Hong 2002) through bottom-up simulation. They fare better in analysis of systems with socio-technical elements.

However, the above mentioned techniques and technologies capture only a fragment of what ought to be captured and analysed for effective CDDM (Kurt et al. 2016). For example, enterprise modeling languages are incapable of specifying uncertainty as well as emergent behaviour (Barat et al. 2016), and actor/agent languages are inadequate to express complex goal structure, organisational hierarchies, and behavioural uncertainty in a relatable form (Bonabeau 2002). Moreover, as none of the EM specifications and actor based languages are designed for decision making purpose they are found lacking in expressing the necessary decision

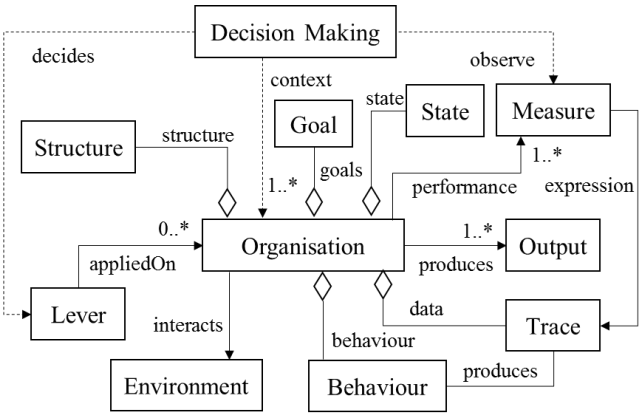

Figure 1: Structural description of decision making

making concepts in an intuitive and closer-to-the-problem manner (Bonabeau 2002, OMG 2016).

This paper discerns a structure on the required information for CDDM and presents a conceptual meta-model, OrgML, to better support CDDM overcoming some of the present limitations. The OrgML is capable of specifying the relevant aspects of organisation and their inter-relationship in a formal machine interpretable form. The key research contributions of this paper are three-fold: (i) a meta-model that represent the structure on the required information for CDDM, (ii) definition of OrgML, i.e., a DSL that captures the above structure in a precise form, and (iii) a systematic derivation mechanism from OrgML to a simulatable form. We claim that the proposed structured representation of decision making problem is an advancement over recent standardisation initiative on Decision Model and Notation (OMG 2016). We also claim that OrgML as a Domain Specific Language (DSL) for CDDM is an advancement over existing enterprise modeling and actor languages for the very same purpose.

The rest of the paper is structured as follows: section 2 describes the modelling and analysis requirements of CDDM, and it briefly evaluates the state-of-the-art techniques and technologies for the same. Section 3 introduces OrgML, establishes the conceptual relationships with foundational concepts, and proposes an approach to use OrgML effectively. Section 4 presents the validation of our claims through a case-study from real life; this section also demonstrates how OrgML is an advancement over existing EM techniques, actor languages, and some of our earlier work. Section 5 provides conclusions and future work.

PROBLEM FORMULATION

This section formulates a structure on the required information for CDDM inspired by some of the decision making models from management sciences literature, and evaluates the state-of-the-art of modelling and analysis techniques for supporting this structure thus establishing a background for our research.

CDDM Structure and Requirements

The philosophical basis of our solution is largely inspired by the decision making models from management sciences such as rational model (Simon 1955), Incremental model (Cyert and March 1992), Carnegie model (Mintzberg et al. 1976) and Garbage Can (Cohen et al. 1972). Though adopting different methodological styles, these models agree on the core concepts of decision making namely, objective or *goal*, course of action or *lever*, and performance indicator or *measures* (Yu 2012). Further they rely on *contextual information* as the basis to analyse achievability of a goal or efficacy of a lever for achieving goal.

Table 1: Requirements of CDDM

	Requirement	Description
Aspect	Why	Goals, objectives and intentions of multiple stakeholders
	What	Structural Specification with complex hierarchy and interactions
	How	Behavioural specification with interactions
	Who	Stakeholders and human actors of the system
	Where	Information about location
	When	Temporality in behaviour and adaptation
Socio-technical Characteristics	Modular	A system can be decomposed into multiple parts.
	Compositio-nal	Multiple parts should be composed to a consistent whole.
	Reactive	Must respond appropriately to its environment
	Autonomou s	Possible to produce output without any external stimulus.
	Intentional	Intent defines the behaviour
	Adaptive	Adapt itself based on context and situation
	Uncertain	Precise intention and behaviour are not known a-priori.
	Temporal	Indefinite time-delay between an action and its response
DC	Measure	Ability to specify what needs to be measured
	Lever	Ability to specify possible courses of action
Analysis	Machine Interpretable	Models that are interpretable by machine (*i.e.*, support for simulation/execution)
	Top-down and Bottom-up	Support for top-down and bottom-up modelling and simulation to support reductionist view and emergentism

Therefore, we argue that the activity of decision making largely depends on two key factors: (i) the ability to capture the core decision making concepts and contextual information in a formal manner, and (ii) the ability to perform *what-if* and *if-what* analyses on the information captured. The former requires completeness and the latter expects the efficacy. We argue that comprehensive information about six interrogative aspects namely *why*, *what*, *how*, *when*, *where*, and *who* as recommended in Zachman framework (Zachman 1987) ensures completeness, and a suitable processor, say a simulator or an interpreter, of the specification ensures efficient and effective analysis.

The class diagram in Figure 1 overlays a structure on the information necessary for CDDM. It depicts the relevant concepts borrowed from management sciences namely, *Goal*, *Measure* and *Lever*, and the concepts necessary to capture contextual information namely, *Goal*, *Structure*, *State*, *Trace*, *Behaviour* and *Environment*.

An *Organisation* relies on its Structure and Behaviour to produce Output so as to achieve the stated Goals while operating in an Environment. Behaviour induces State changes thus producing Trace (i.e. historical record comprising of State and Output) over a period of time. Goal

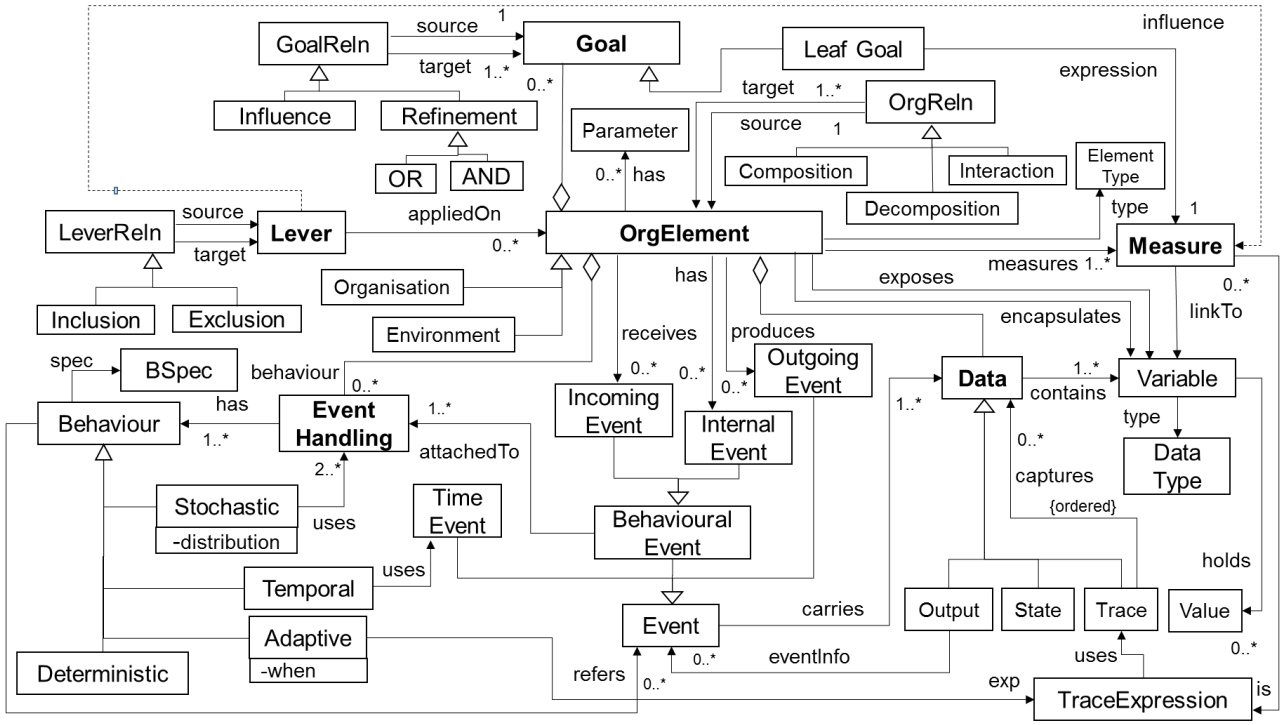

Figure 2: OrgML – a metamodel to represent organisation

is a conditional expression over Measures which are views over Trace. A Lever is possible modification to Goal and/or Behaviour, and/or Trace.

The complex dynamic decision making (CDDM) for large and complex organisation with volatile operating environment calls for additional requirements on the specification described in Figure 1. A large organisation often contains complex hierarchy with large number of socio-technical elements as part of its structure. The constituent socio-technical elements are mostly autonomous, reactive and goal-directed. Also, the organisation elements exhibit uncertainty, temporality and adaptability. Thus, complex structure, socio-technical characteristics, and inherent uncertainty form additional specification requirements for CDDM. The specification requirements also demand the necessary constructs to represent decision making related concepts namely Goal, Measure, Lever. A-priori assessment of decisions is suggestive of simulation capability. Further, the simulation can be approached using *top-down* or *bottom-up* manner. Therefore a CDDM may need any of these two approaches or it may demand a middle-out approach. Table 1 enumerates the specification and analysis needs for CDDM.

Review of state of the art and practices

In (Barat et al. 2016), we systematically evaluated the suitability of EM techniques to support CDDM. The evaluation concluded with a critical observation that the existing EM techniques are capable of satisfying the expected requirements of CDDM described in Table 1 only in parts. In particular, we found the EM techniques that support necessary aspects of CDDM (such as Zachman Framework (Zachman 1987) and ArchiMate (Iacob et al. 2003)) are not machine interpretable and thus not amenable for rigorous analyses. Similarly, prevalent general purpose conceptual enterprise model, MEMO (Frank 2002), supports most of the specification needs except decision making related constructs. In contrast, specifications capable of precise analyses, such as BPMN (OMG 2011), i* (Yu et al. 2006) and System Dynamic (SD) (Meadows and Wright 2008) models, on their own, are not capable of representing all necessary aspects. For instance, BPMN analyses and simulates the process aspect, i* analyses the high level goals and objectives, and SD model simulates complex dynamic behaviour of the system. On the other hand, the multi-modelling and co-simulation environments, such as DEVS (Camus et al. 2015) and AA4MM (Siebert et al. 2010), collectively support the analysis needs for all aspects depicted in Table 1. However, they are not capable of expressing many socio-technical characteristics such as autonomy, uncertainty and temporal behaviour. Moreover, they are not suitable for bottom-up construction of a system that results into emergent behaviour.

The general purpose actor languages and frameworks, such as Scala Actors (Haller and Odersky 2009) and Akka (Allen 2013), are capable of specifying and analysing a range of socio-technical characteristics and emergent behaviour. However, we find the effective use of these general purpose languages in the context of decision making is a hard proposition as they do not have the language support to represent goal, measure and lever in an explicit form. Moreover, a large and complex organisation may need top-down and bottom-up analysis (Bonabeau 2002) which is not supported in general purpose actor languages.

PROPOSED SOLUTION - OrgML

This section presents the core contribution of this paper. It first introduces OrgML meta-model, then conceptually correlates the model with requirements of CDDM, and proposes an approach on using OrgML for CDDM.

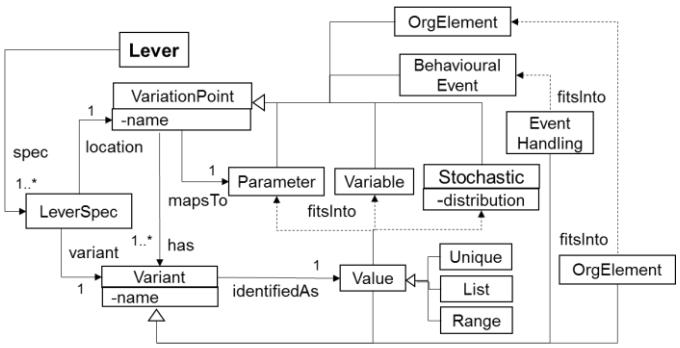

Figure 3: Lever specification of OrgML

OrgML meta-model

OrgML is a meta-model to represent the information required for decision making in a structured and machine interpretable form as shown in Figure 2. It extends the concepts depicted in Figure 1 along two dimensions – (i) to capture the specification requirements described in Table 1, and (ii) to enable the top-down/ bottom-up modelling and simulation.

The core element of OrgML is *OrgElement*, which is a parametric entity that can have: a set of *Goals* to represent its intention or objective, a set of *EventHandling* units to represent behaviour, and *Data* to capture *state* and *trace* of OrgElement. An OrgElement is an event-centric abstraction with <*eventName, eventHandlingSpecification*> tuples constituting its behaviour. Amongst the events being processed, *IncomingEvent* (*i.e.*, events received) and *OutgoingEvent* (i.e., events produced) are exposed to the environment whereas *InternalEvents* are not. The OrgElement Data is a set of typed *Variables* that can hold Values. An OrgElement may expose Variables to other OrgElement thus relaxing data hiding. An OrgElement can have a set of *Measures* to represent key performance indicators. Thus, OrgElement is the basic building block for specifying organisations for decision making. Detailed description of OrgML follows:

Goal: OrgML enables a goal to be decomposed into sub-goals, sub-sub-goals etc. to the desired level of detail. At each level, a goal can be related to other goals through relationships, for instance, meeting a goal g1 disables the related goal g2 to be met, or meeting a goal g1 subsumes the related goal g2 etc. We group such dependency relationships under *Influence* kind of relationship. The other kind of relationship is *Refinement* which enables hierarchical decomposition of goals. OrgML supports all the key relationships from i* goal specification (Yu et al 2006). A *LeafGoal* is a conditional expression over Measures. The conditional expression language includes temporal operators. As Measures have values, it is possible to compute if a LeafGoal is met or not. The semantics of Influence and Refinement relationships determine evaluation of the overall goal as a bottom-up traversal of the graph.

Structure: An OrgElement supports structural composition, decomposition and interactions through *OrgReln*, and represents type using *ElementType*. Organisational units, stakeholders and system of an organisation can be represented using ElementType. One can also consider the *Active*- and *Passive*- Structure defined in ArchiMate (Iacob et al. 2003) as the basis for type definition. Further, we consider Organisation and its *Environment* are two specialised OrgElement. An Organisation or Environment cannot be composed with other OrgElement. However, they can interact with other OrgElement and decompose into finer level of granularity. For example, an Organisation can interact with Environment, Organisation can be decomposed into Organisational units, an Environment of an Organisation can be visualised as multiple competitors, *etc*.

Data: Data of OrgElement is of three kinds namely *Output*, *State*, and *Trace*. The *Output* represents outcome of executing an OrgElement. It represents: (i) Variables of the OrgElement , (ii) Events produced internally, and (iii) Events communicated to other OrgElements. The *State* represents Variables of the OrgElement. *Trace* is collection of Data of the OrgUnit from a point in time in the past till now.

Behaviour: Elements Event and EventHandling describe the behaviour of an OrgElement. The Event can be classified into three categories namely *OutgoingEvent*, *TimeEvent*, and *BehaviouralEvent*. The *OutgoingEvent* specifies the Data from OrgElement accompanying the Event. The *TimeEvent* is an Event that represents Time information. The *BehaviouralEvent* specifies the Event definition and its implementation using EventHandling element. The BehaviouralEvent is further classified into IncomingEvent and InternalEvent wherein the former represent the Events that are consumed by the OrgElement and the latter represent the Events that are internal to the OrgElement. The EventHandling element is the primitive behavioural unit for describing the Behaviour of an OrgElement which can be of four kinds namely *Deterministic*, *Stochastic*, *Temporal* and *Adaptive*. We consider standard language constructs such as *assignment*, *expression evaluation*, *loop*, *recursion*, *message passing*, *etc.*, to express Deterministic Behaviour. The Stochastic Behaviour specifies the uncertainty along two dimensions – uncertainty in raising an OutgoingEvents and uncertainty in responding to an IncomingEvent. Specification of both requires use of special constructs providing probability distribution guided choice. The Temporal Behaviour uses TimeEvent to express temporal relationships within behavioral specification. Adaptive Behaviour describes adaptation rules. Essentially, it express a Behavior that activates when a specific condition is matched – it uses TraceExpression, *i.e.*, an expression over Trace element, to define the conditions. We consider element *BSpec* as a placeholder for behavioural specification.

Measure: *Measure* are a set of *TraceExpression* that essentially represents Variables of interest.

Lever: *Lever* represents possible courses of action that can be applied on OrgElement. A lever specification contains two kinds of specification: (i) lever usage specification and (ii) lever definition. Lever usage specification is illustrated in Figure 2 using *LeverReln* and its specialisation. The Lever inclusion and exclusion relationships can be defined using *LeverReln*.

The lever definition specifies a modification to either structure or data or behaviour of an OrgElement or a combination thereof. We adopt the concept of *Variation Point* and *Variant* of variability modelling (Kulkarni 2012) to define lever specification as depicted in Figure 3. Essentially, a lever is set of *LeverSpec* where each LeverSpec

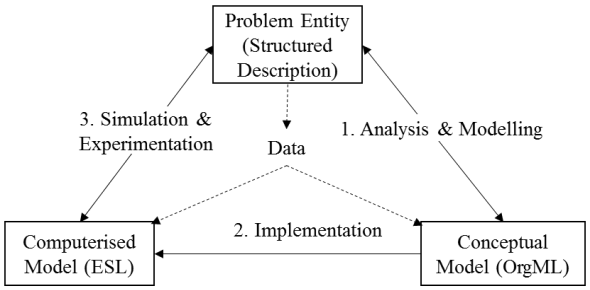

Figure 4: Overview of modelling and simulation approach

describes the change specification using two named elements namely *VariationPoint* and *Variant*. The VariationPoint describes the location of a change, and the Variant describes the changed element. A predefined set of core elements of OrgML can act as VariationPoint and Variant. Further, there is a notion of compatibility between VariationPoint and Variant. The element Parameter, Variable, Stochastic Behaviour, Behavioural Event and OrgElement of OrgML can act as VariationPoints wherein the element Value can fit into Parameter, Variable, Stochastic Behavior; the element EventHandling can fit into BehaviouralEvent; and an OrgElement can fit into OrgElement. We consider VariationPoint as a Parameter, and Variant as a Value to realise fitsInto relationship.

Analysis of OrgML as a decision making aid

Conceptually, the elements of OrgML refines the structure defined in Figure 1 and enables the characteristics described in Table 1. Event definition, Data, and OrgElement structure specify the *what* aspect, OrgElement help specify the *who* aspect, Goal specification specifies the *why* aspect, and Behaviour specifies the *how* and *when* aspects. The concept of OrgElement ensures desired modularity and encapsulation: the Event helps to specify reactive nature, InternalEvent and TimeEvent collectively specify the autonomous behaviour, Stochastic Behaviour helps in specifying required uncertainty, the Temporal Behaviour and TimeEvent specify the temporal behaviour, and Adaptive Behaviour is capable of specifying the adaptive nature of an OrgElement. We argue that the *Composition* relationship of OrgElement and *Influence* relationship of Goal specification together help in bottom-up design, whereas the *Decomposition* relationship of OrgElement, *Goal Refinement Relationship*, and an ability to share Variables using *exposes* relationship help in top-down design.

The proposed meta-model is grounded with a set of existing concepts. The modularisation and unit hierarchy are taken from component model concepts. Goal-directed reactive and autonomous behaviour can be traced to actor behaviour (Hewitt 2010). Defining states in terms of a type model is borrowed from UML. An event driven architecture is introduced for reactive behaviour. The concept of intentional modelling (Yu et al. 2006) is adopted to enable specification of goals. The behavioural classification and uncertainty is defined from the *uncertainty theory* by Donald Rumsfeld (Rumsfeld 2011) wherein the *known knowns* behaviour can be specified using Deterministic Behaviour and the *known unknowns* (KU) can be specified using Stochastic Behaviour.

Table 2: Guideline for constructing OrgML model from existing specifications

OrgML Concept	Possible sources (concept mapping from existing languages)
OrgElement	UML Class Diagram:: Class that represents Organisational elements such Organisation, Organisational Unit, Environment. ArchiMate:: Business Actor, Business Role, Business Object, Application Component, System Software
Data	UML Class Diagram:: Class that represents entities ArchiMate:: Data Object, Artifacts. BPMN:: Data Object
Goal	i* specification:: Goal. ArchiMate:: Meaning
Behaviour	UML State Machine:: State, Transition ArchiMate:: Business Service, Business Process, Business Function, Application Function, Infrastructure Function. BPMN:: process definition
Event	UML State Machine: Transition. BPMN:: Event. ArchiMate:: Business Interaction, Business Event
Measure	i*:: Task, Leaf level Goal. BPMN:: KPI
Lever	Description about possible courses of action

Enabling CDDM using OrgML

We adopt a simplified modelling method recommended by Robert Sargent in (Sargent 2005) to capture organisation specification using OrgML and enable required analysis. Method uses three distinct representations namely problem entity, conceptual model and computerized model, to systematically transform a real-life problem into analyzable model and perform analysis/simulation as shown in Figure 4. The problem entity is a description of real environment, conceptual model is a purpose specific representation of the problem entity, and computerised model is an executable/simulatable model of conceptual model. In our approach, we consider textual description, i*, class diagram, state-machines, BPMN, and ArchiMate as the possible specification aids to describe a problem entity, OrgML is a specification aid for conceptual model, and an actor-based language named as *Enterprise Simulation Language* (ESL) (Tony et al. 2017) as computerised model.
ESL, like standard actor languages (Haller and Odersky 2009, Allen 2013), supports the notion of *event*, *state*, *reactive*, *autonomous* and *adaptive* behavior. ESL supports further concepts such as *Time*, *Stochastic* behaviour (using *probability* distributions) and *Temporal* behavior as shown in Figure 5. These additions (depicted with dotted box and lines in Figure 5) are beneficial in making ESL a transformation

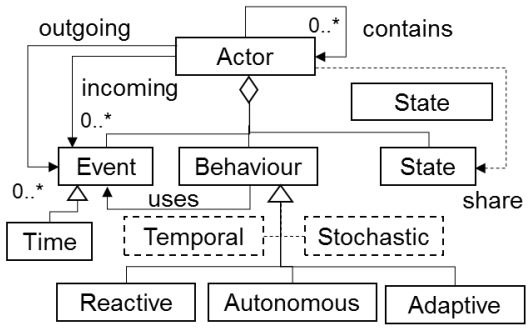

Figure 5: Characteristics of ESL

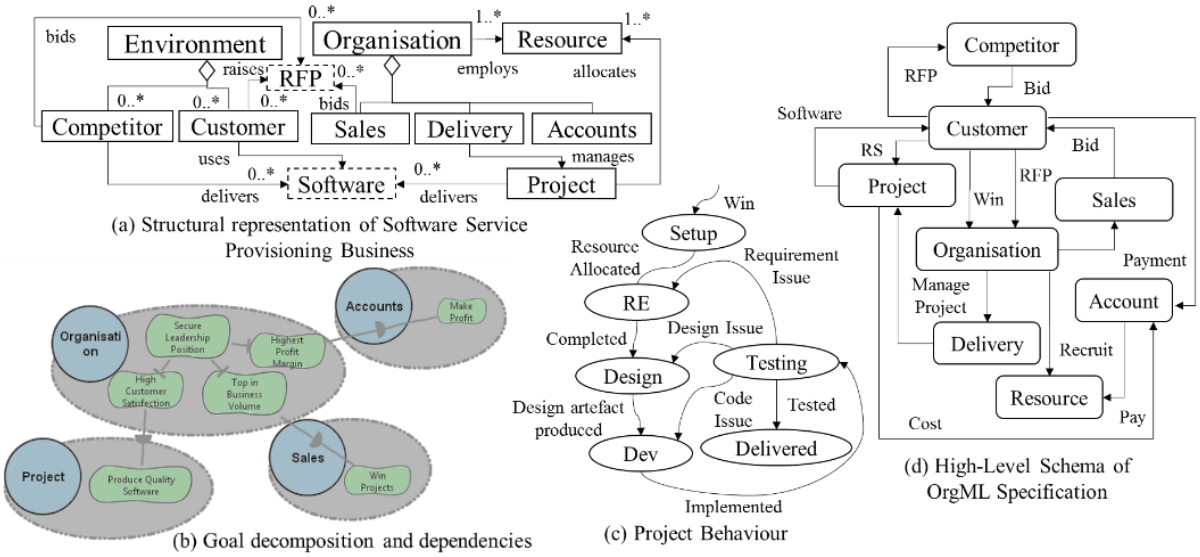

Figure 6: Specifications of Software Service Provisioning Organisation Case Study

target for OrgML.

Step 1 *(Analysis and Modelling)* manually converts a *problem entity* specification into an OrgML specification by identifying primitive elements, such as *Organisation*, *Organisational Unit*, and *Stakeholders*, and specifying their goals, behaviour, and measures. For example, the goals represented using i* model can be translated into the OrgML goal specification wherein the *'decomposition'* relationships of i* can be translated into OrgML *Refinement* relationships, A basic guideline for constructing OrgML specification from existing specification languages is depicted in Table 2.

Step 2 *(Implementation)* uses a fixed set of transformation rules to transform OrgML specification into an ESL specification and is depicted in Table 3. We use a java based ESL simulator to simulate converted ESL specifications. A simulation of transformed specification progresses with primitive Time events (of ESL) wherein each translated OrgElement performs its own behaviour by reacting to IncomingEvents and InternalEvents. It updates state variables, persists trace information, produce OutgoingEvents, and evaluates goal expressions. An OrgElement adapt to new behaviour when adaptation logic is satisfied. The simulator displays identified measures in the form of graphics and animation.

EVALUATION

We present an evaluation of OrgML and the proposed modelling approach using an established real-life case study of a software service-provisioning organisation (Kulkarni et al. 2015a, Kulkarni et al. 2015b)

The case study, primarily, focuses on the decision making of a *Software Service-Provisioning Organisation* (SSPO) that aims to *Secure Leadership Position* by focusing on three sub-goals namely *High Customer Satisfaction*, *Top in Business Volume* and *Highest Profit Margin*. The SSPO aim to achieve its goals by provisioning high-quality bespoke software in a dynamic and competitive environment that contains multiple customers who outsource their development activities to service provisioning organisations, and a set of competitors who have similar goals as SSPO.

Internally, SSPO contains three autonomous organisational units namely, *Sales*, *Delivery*, and *Account*, a dynamic organisational unit, terms as *Project*, which is formed on demand, and a set of skilled *Resources*. A simplified structure of the case study is depicted using a class diagram in Figure 6 (a). Each organisational unit of SSPO has its goals that contribute to the organisational goal as shown in Figure 6(b); and each organisational unit has its own behaviour to achieve their individual goals. A simplified behaviour of Project unit is illustrated using a state-machine in Figure 6 (c). In general, Customers raise *request for proposal* (RFP) for software provisioning; SSPO organisation and *Competitors* bid RFPs; and Customers evaluate bids and select one organisation for service provisioning. Once a bid is won by SSPO, it forms a development *Project* by allocating appropriate *Resources*, executes the project using standard software development process (as depicted in Figure 6 (c)), and finally delivers *Software* to the respective *Customer*.

Table 3. OrgML to ESL Transformation Rule

OrgML	ESL
OrgElement	Actor
Data	Actor Variables
Goal	Expression over Actor variables
Event	Event
EvenHandling	Event Handling specification
Measure	Expression over Actor variables
Lever	ESL specification
Parameter	Input parameter in 'new' operator of ESL Actor
Trace	Actor Variable
BSpec	
Deterministic	Standard Behavioural specification
Stochastic	Probabilistic Behaviour
Temporal	Behaviour with temporal expression
Adaptive	Behavioural with guarded expression
Deterministic	Behavioural specification

The case study explores the possibility of achieving high level goal, *i.e., Secure Leadership Position*, and sub-goals *i.e., High Customer Satisfaction*, *Top in Business Volume* and *Highest Profit Margin*, of SSPO for a given environment that contains a set of Customers with varying outsourcing needs, and a set of Competitors with specialised bidding strategies and project execution strategy. The case study further explores various Levers, such as a *competitive bidding strategy*, *increase of resource strength*, *recruitment of skilled resources*, and *reskilling of existing resources*, as possible options to increase the possibility of achieving its goals.

In our evaluation, we considered an extended form of SSPO specification depicted in Figure 6 (a)-(c) as the *problem entity* specification (extended figures are not included due to space limitation). In our approach, we first constructed OrgML specification of SSPO from problem entity specification using the guideline described using Table 2. An OrgML specification specify SSPO, organisational units (*i.e.*, Sales, Delivery, Account, and Projects), and the identities that describes Environment (*i.e.*, Competitor and Customer) as parameterised OrgElement. The goals (as depicted in i* model) are translated into the OrgML goal specification. Three Measures are identified from LeafGoals. The Measures are: *Customer Satisfaction Index*, *Business Volume* and *Profit Margin*. The behavioural specifications that are represented as state-machines are translated into OrgML Events and Event specifications. A schema of converted OrgElements and their interactions of constructed OrgML specification is depicted in Figure 6 (d). Next, we converted SSPO OrgML specification into SSPO ESL specification using transformation rules depicted in Table 3 to simulate SSPO using ESL simulator and validated simulation results with earlier experimentation presented in (Kulkarni et al. 2015a).

We used SSPO case study to validate multiple enabling techniques and technologies that have potential to serve as effective aids for CDDM. In (Kulkarni et al. 2015b), we evaluated a multi-modelling and co-simulation approach that used three prominent EM techniques namely i*, BPMN and Stock-and-Flow in a coordinated manner. In (Kulkarni et al. 2015a) we experimented ESL to encode SSPO case study and evaluate decision alternatives. This paper proposes OrgML supported with a method as an affective modelling and analysis aid for CDDM. The effectiveness of four alternatives, i.e., EM based approach (from (Kulkarni et al. 2015b)), pure actor language based approach (from existing literature such as (Bonabeau 2002), ESL based approach (from (Kulkarni et al. 2015a)), and OrgML based approach are summarised in Table 4. As shown in the table, an EM based approach and an actor language based approach are complementary in nature. The former one supports aspect (*i.e.*, *why*, *what*, *how*, *etc.*) specification and a top-down simulation approach, whereas actor language based approach is more effective for representing socio-technical characteristics and bottom-up simulation approach. But, it is not convenient for aspect specification. ESL bridge the gaps between two class of specifications with explicit support for uncertainty, temporal behaviour, and the bottom-up and top-down combination. Therefore we argue that ESL is technically complete for CDDM requirements described in Table 1. However, ESL is a general purpose simulation language and it is not convenient to specify most of the aspects (*e.g.*, *why*, *who*, *where*, *when*), and decision making constructs namely *goal*, *measure* and *lever*. Hence OrgML is further improvement towards the infrastructure that we envision for CDDM. It helps in expressing the most of the requirements in a convenient and machine interpretable for leading simulation based CDDM.

Table 4. Evaluation Summary

Requirement	EM Spec.	Actor Lang.	ESL	Org ML	Enabling OrgML Concepts
Why	√	⊥	⊥	√	Goal
What	√	√	√	√	OrgElement
How	√	√	√	√	EventHandling
Who	√	⊥	⊥	√	OrgElement
Where	√	⊥	⊥	⊥	OrgElement
When	√	⊥	⊥	⊥	Time Event
Modular	√	√	√	√	OrgElement
Compositional	⊥	√	√	√	Composition Relationship
Reactive	⊥	√	√	√	IncomingEvent, OutgoingEvent
Autonomous	×	√	√	√	InternalEvent
Intentional	√	√	√	√	Goal
Adaptive	⊥	√	√	√	Adaptive Behaviour
Uncertainty	×	⊥	√	√	Stochastic Behaviour
Temporal	⊥	×	√	√	Temporal Behaviour
Measure Spec	⊥	⊥	⊥	√	Measure
Lever Spec	⊥	⊥	⊥	√	Lever
Top-down/ Bottom-up	Top-down	Bottom-up	Hybrid	Hybrid	Composition Relationship, Shared State Variable

Legends: √ : Supports adequately, ⊥ can be specified with difficulties, × : not supported

CONCLUSION

Currently, there is a gap in technologies available for decision making notably in the precision of aspects such as precision of organisational goals, structure, operational processes, environment and expressing uncertainty. To address this gap, we have presented OrgML - a meta-model structure over information required for decision making. The model content is inspired by key concepts advocated in various decision making models from management sciences. The formal structure is achieved by integrating appropriate concepts from Enterprise Modelling, actor model of computation, uncertainty theory and variability modeling.

A rationale for how OrgML can enable a decision making approach that can potentially overcome some of the limitations of current state of art and practice of CDDM has been presented. The principal benefits are derived from an extended form of actor model of computation; is composable; is capable of specifying uncertainty in behavior; and is simulatable. A systematic approach on how to derived an simulatable specification from an OrgML model instance using a model map has been outlined. The derivation process and the efficacy of the OrgML model was evaluated using a real-life scenario. We acknowledge this paper does not discuss in detail the language constructs of ESL, the transformation rules from problem entity specification to OrgML, and automatable transformation rules to translate

OrgML to ESL. We restricted the principal objectives of this paper to: establishing the core concepts of CDDM, defining a specification language that can act as an effective aid for specifying organisations for CDDM and enabling a simulation based approach to CDDM.

Our research is an example of technical action research (Wieringa and Moralı 2012) in that it presents a validation of a design science artifact wherein we have demonstrated how OrgML and the proposed approach is relevant and effective for practitioners to adopt in CDDM. As part of future research, we intend to further validate this in real business scenarios as well as proposing further extensions to OrgML for introducing game theoretic approach in simulations for CDDM.

REFERENCES

Allen, J. 2013. Effective Akka. O'Reilly Media.

Barat, S., Kulkarni, V. Clark, T. and Barn, B. 2016. "Enterprise modeling as a decision making aid: A systematic mapping study." The Practice of Enterprise Modeling, pp. 289-298.

Beckermann, A., Hans F., and Jaegwon K. 1992. "Emergence or reduction?: Essays on the prospects of nonreductive physicalism". Walter de Gruyter.

Bonabeau, E. 2002. "Agent-based modeling: Methods and techniques for simulating human systems." Proceedings of the National Academy of Sciences 99, no. suppl 3: 7280-7287.

Camus, B.; Bourjot, C.; Chevrier, V. 2015. "Combining DEVS with Multi-agent Concepts to Design and Simulate Multi-models of Complex Systems." <hal-01103892>

Clark, T.; Kulkarni, V.; Barat, S.; Barn, B. 2017. "ESL: An actor-based platform for developing emergent behaviour organisation simulations". In: International Conference on Practical Applications of Agents and Multi-Agent Systems. pp. 311–315.

Cohen, M. D.; March, J. G.; Olsen, J. P. 1972. "A Garbage Can Model of Organisational Choice". Administrative Science Quarterly 17 (1): 1–25.

Cyert, R.; March, J. G. 1992. A Behavioral Theory of the Firm (2 ed.). Wiley-Blackwell. ISBN 0-631-17451-6.

Frank, U. 2002. "Multi-perspective enterprise modeling (memo) conceptual framework and modeling languages." In System Sciences, 2002. HICSS. pp. 1258-1267.

Haller, P. and Odersky, M., 2009. "Scala actors: Unifying thread-based and event-based programming". Theoretical Computer Science, 410(2), pp.202-220.

HBR 2014, "9 Habits That Lead to Terrible Decisions." https://hbr.org/2014/09/9-habits-that-lead-to-terrible-decisions.

Hewitt, C., 2010. "Actor model of computation: scalable robust information systems". arXiv preprint arXiv:1008.1459.

Iacob M, et al. 2003. "State of the art in architecture support, ArchiMate deliverable D3.1." Enschede, The Netherlands: Telematica Instituut.

Kulkarni, V.; Barat, S.; and Roychoudhury, S. 2012. "Towards business application product lines." Model Driven Engineering Languages and Systems, pp. 285-301.

Kulkarni, V.; Barat, S.; Clark, T.; Barn, B. 2015a. "Using simulation to address intrinsic complexity in multi-modelling of enterprises for decision making." In Proceedings of the Conference on Summer Computer Simulation, pp. 1-11.

Kulkarni, V.; Barat, S.; Clark, T.; Barn, B. 2015b. "Toward overcoming accidental complexity in organisational decision-making." Model Driven Engineering Languages and Systems (MoDELS 15).

Kurt, S. et al. 2016. "Enterprise Modelling for the Masses–From Elitist Discipline to Common Practice.". The Practice of Enterprise Modeling, pp. 225-240.

Locke, E. 2009. Handbook of Principles of Organisational Behavior: Indispensable Knowledge for Evidence-Based Management. ISBN: 978-0-470-74095-8

Macal, C.M. and North, M.J., 2010. "Tutorial on agent-based modelling and simulation". Journal of simulation, 4(3), pp.151-162.

McDermott, T.; Rouse, W.; Goodman, S.; Loper, M. 2013. "Multi-level Modeling of Complex Socio-Technical Systems." Conference on Systems Engineering Research (CSER'13).

Meadows, D.H. and Wright, D. 2008. Thinking in systems: A primer. chelsea green publishing.

Mintzberg, H.; Raisinghani, D.; and Theoret, A. 1976. "The structure of unstructured decision processes". Administrative Science Quarterly, 21, 246-275

O'Connor, T., and Hong Y. W. 2002. "Emergent properties.".

OMG. 2011. Business Process Model and Notation. http://www.omg.org/spec/BPMN/2.0/, formal/2011-01-03

OMG. 2016. Decision Model and Notation (DMN), www.omg.org/spec/DMN/1.1/

Rumsfeld, D. 2011. Known and unknown: a memoir. Penguin.

Sargent, R.G., 2005. "Verification and validation of simulation models". In Proceedings of the 37th conference on Winter simulation (pp. 130-143).

Shapira, Z. 2002. Organisational Decision Making. Part of Cambridge Series on Judgment and Decision Making. ISBN: 9780521890502

Siebert, J.; Ciarletta, L., and Chevrier, V. 2010. "Agents and artefacts for multiple models co-evolution: building complex system simulation as a set of interacting models." Autonomous Agents and Multiagent Systems: volume 1-Volume 1, pp. 509-516.

Simon, H. A. 1955. "A behavioral model of rational choice." The quarterly journal of economics 69, no. 1: 99-118.

Wieringa, R.; Moralı, A. 2012. "Technical action research as a validation method in information systems design science." In International Conference on Design Science Research in Information Systems, pp. 220-238. Springer Berlin Heidelberg.

Yu, E.; Strohmaier, M; Deng, X. 2006. "Exploring intentional modeling and analysis for enterprise architecture". In EDOC Conference Workshops.

Yu, P. 2012. "Multiple-criteria decision making: concepts, techniques, and extensions". Vol. 30. Springer Science & Business Media.

Zachman, J.A. 1987. "A framework for information systems architecture." IBM systems journal, 26(3):276-292.

BIOGRAPHY

Souvik Barat is senior researcher at Tata Consultancy Services Research (TCSR). He holds a Masters Degree in Computer Science and Engineering from Indian Institute of Technology Madras.

Vinay Kulkarni is distinguished Scientist at Tata Consultancy Services Research (TCSR). He holds a Masters Degree in Electrical Engineering from Indian Institute of Technology Madras.

Tony Clark is Head of the Department of Computing and a Professor of Software Engineering at Sheffield Hallam University in the UK.

Balbir Barn is Deputy Dean and Professor of Computer Science at Middlesex University London.