The WaaSaBE model: Marrying WaaS and Business-Entities to Support Cross-Organization Collaboration

Lior Limonad
IBM Research - Haifa,
Carmel Mountain, Haifa 31905, Israel
liorli@il.ibm.com

Lav R. Varshney
IBM Thomas J. Watson Research Center
Hawthorne, New York, USA
lrvarshn@us.ibm.com

Daniel V. Oppenheim
IBM Thomas J. Watson Research Center
Hawthorne, New York, USA
music@us.ibm.com

Elad Fein
IBM Research - Haifa,
Carmel Mountain, Haifa 31905, Israel
eladf@il.ibm.com

Pnina Soffer
Information Systems Department, University of Haifa
Carmel Mountain, Haifa 31905, Israel
spnina@is.haifa.ac.il

Yair Wand
MIS division, Sauder School of Business,
University of British Columbia,
Vancouver, Canada
yair.wand@sauder.ubc.ca

Moran Gavish
IBM Research - Haifa,
Carmel Mountain, Haifa 31905, Israel
morang@il.ibm.com

Ateret Anaby-Tavor
IBM Research - Haifa,
Carmel Mountain, Haifa 31905, Israel
atereta@il.ibm.com

Abstract* — With a growing services-based focus in enterprises, functionally-tiered organizational structures have emerged. Synchronizing among the tiers is difficult due to differing concerns and vocabularies, especially when cross-enterprise collaboration is involved. Furthermore, deficiencies in work handoff among different roles and parties also occur within tiers. Building on the notions of work-as-a-service for work execution and business entities in operations, this paper proposes the WaaSaBE model as a boundary object for integrated management across and within tiers and across enterprises. We describe the framework, a formal model arising from the framework, and its basic instantiation for a given problem domain.

Keywords-cross enterprise collaboration; business artifact; business entity; work as a service;

I. INTRODUCTION[1]

When enterprises move from a technology-oriented focus to one that is services-based, it is common to split the enterprise domain into three functional tiers: business, operations, and work execution. As illustrated in Fig. 1, the *business* tier focuses on strategic decisions and goal setting. The *operational* tier is a realization of the business tier, focused on synchronization and control of work. Finally, the *work execution* tier accomplishes operational objectives through efficient utilization of human competencies, technology, and information. With rising globalization and specialization, each of these functions may be notably split across multiple geographies and organizations that collaborate to realize larger opportunities [1].

Managing the interplay among the three enterprise tiers is fundamentally difficult. It requires synchronization among three separated efforts, each with its own concerns and vocabulary. Currently, a variety of methods and supporting technologies are aimed at handling subsets of concerns arising from each of the organizational layers. Pragmatically, this necessitates integrating and manually synchronizing between independent supporting IT systems, each used for the design, management, and execution of a fraction of a given tier's requirements.

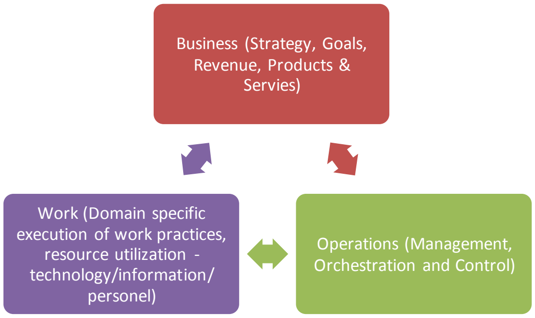
Business (Strategy, Goals, Revenue, Products & Servies)
Work (Domain specific execution of work practices, resource utilization - technology/information/ personel)
Operations (Management, Orchestration and Control)

Fig. 1 - Enterprise as an intertwined effort between business, operations, and work

One might hope that existing methods and technologies from a single tier could be trivially extended to apply across all tiers. For example, In the operational tier, Business Process Management (BPM) [2–4] provides a collection of modeling standards that are centered around the creation of symbolic representations depicting how businesses conduct

[1] The research leading to these results has received funding in part from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement no. 257593.

their operations. Yet, our contention is that these models are neither adequate for modeling the actual "doing of work" nor the setting of business objectives.

A need exists for an integrated conceptual framework (model, methodology, and corresponding tools) that allows management across enterprise tiers. As a first step towards the desired framework, this paper describes the WaaSaBE model, having its roots in the Work-as-a-Service (WaaS) framework for work execution [5], Guard-Stage-Milestone models for operations [6], [7], and Business Entities (BE) for business concerns [8], cf. Fig. 2.

Such a framework would not only be useful within single enterprises, but also streamline cross-enterprise collaborations by facilitating cohesive communication and understanding between all participants, while capturing the unique traits and concerns expressed by each; this even more so in flexible organizational structures and adhocracies.

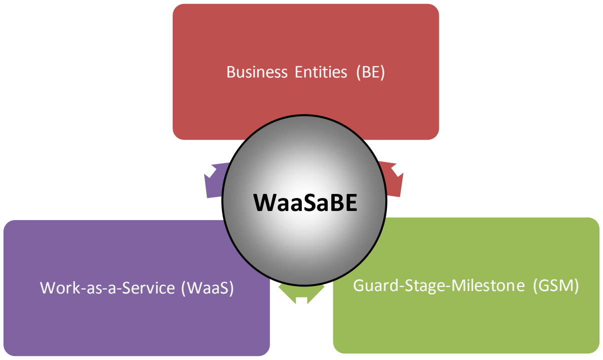

Fig. 2 - WaaSaBE as a boundary object among base models that exist for business, operations, and work.

II. General Approach

We aim to construct a modeling aid that can simultaneously address the concrete concerns of the three tiers:

1. From a *business* perspective: We require a model that is intuitive to business practitioners, giving a clear indication of the current state of affairs with respect to business-level strategies and goals.

2. From an *operational* perspective: We require a model that can adequately express operational realization of goals, local adherence to policies, and that is sufficiently flexible to cope with change, coordination, and decomposition.

3. From a *work execution* perspective: We require a model that allows efficient utilization or reuse of resources and human competencies, and accordingly can accurately express both efforts to be delegated to external contractors or capabilities to be offered to external consumers.

To do so while maintaining the separate vocabularies and concerns, we want WaaSaBE to be a *boundary object* or pidgin language. Originally arising in the field of sociology [9], the notion of a boundary object refers to an entity shared by different communities of practice and which may be interpreted or used by each community in a different way. The goal of the boundary object is to facilitate cross-functional and cross-organization communication and coordination. This makes the boundary object itself an interface among these communities.

In related work in conceptual modeling, UML was used as a two-sided boundary object to bridge between stakeholders and analysts [10]. Here we construct a three-sided boundary object that combines three extant modeling grammars into a unified one. The unified modeling grammar becomes a consistent representation of key concerns across business, operations, and its realization in actual work execution.

Further, we view our domain of analysis as a socio-technical space in which human resources and IT coincide and cooperate to accomplish business objectives; this follows Alter's notion of a work system [11], [12]: a "system in which human participants and/or machines perform work using information, technology, and other resources to produce products and/or services for internal or external customers."

The work system view enables us to break down an organizational domain into three sub-domains:

1. *Consumer's/requestor's domain*: The work system's *environment* comprising external customers whose expectations from the work system reflect high-level business goals the system is aimed to achieve

2. *Provider's domain*: The work system itself, which is both committed and operationally constructed to satisfy the goals being set by its consumers.

3. *Contractual domain*: The boundary space between the environment and the work system specifying the nature of interactions between consumers and providers; such interactions are detailed to specify the products and/or services the work system is capable of delivering to its customers, including the terms under which such interactions may be accepted by the consumers.

In light of the above, WaaSaBE should be equipped to illustrate both the characteristics of each work-system's commitments to its potential consumers, and the actual realization of such commitments by internal participants and/or machines as work execution. Correspondingly, we aim to create a designated modeling grammar to separately enable expression of each analysis perspective (i.e., work commitment to consumers and work realization).

The rest of the paper is structured as follows: Section III describes the theoretical foundations that were used as the basis for the clarification of business semantics. Section IV presents the three underlying grammars: *WaaS, BE, and GSM*. The illustration of each includes a meta-model that is used to define model constructs and relationships, and corresponding business semantics. Section V then focuses on the nature of the conceptual ties between the grammars, yielding WaaSaBE as a unified model. Finally, Section VI demonstrates an actual instantiation of the unified grammar given a real-world example.

WaaSaBE is designed to be fully scalable, supporting the capability to both outsource some of the specified work commitment in a given provider's domain to other providers, and also to promote existing capabilities to be offered as independent capabilities to other consumer domains.

III. Theoretical Foundation for Business Semantics

To provide the constructs in the model with well-defined meaning that can be related to stakeholders' perspectives, we adapt an ontological view that represents a business domain as a system of interacting elements. The domain and the changes that can occur in it are described in terms of the states of the elements. Specifically, we anchor our interpretation in the *Generic Process Model* (GPM) [13], a notation-independent framework for analyzing business processes, based on Bunge's ontology [14], [15] and its adaptation to information systems [16–18]. We use the conceptualizations in GPM as the basis for providing real-world semantics to key constructs in the model being developed and clarifying the nature of the relationships among the different constructs.

The main premise of Bunge's ontology is that the real world is composed of things that possess properties. Properties can be intrinsic, such as mass, or mutual, such as the salary of a person who works for a company. Properties are perceived by humans as attributes. For example, the property of reflecting a certain wavelength of light is conceived as a color, and the property mass is conceived as weight. Attributes are modeled as attribute functions that attain values that can change with time (and a reference frame). A view of set of similar things can be formalized as a set of attribute functions, termed a functional schema. Each such individual function is termed a *state-variable*, and the state of a thing is the set of values of all its state variables at a given time. Changes of state are termed *events*. The state of a thing can change either as a result of its own actions (internal events) or due to the actions of other things (external events). The latter are termed interactions and are manifested by mutual properties.

A state will be *stable* if it can only change due to the actions of other things, otherwise it will be *unstable.* In our context of business domains, we distinguish between the things in the domain and the things in the environment of the domain, which can interact with things in the domain but are not considered part of it. In this context, a *stable state* can only change as a result of an action of something outside the *domain* (e.g., interaction with things in the *environment*). An *unstable state* is a state in the domain that must change.

Whether a state is stable or unstable and how an unstable state might change is defined in terms of *state laws* and *transition laws* correspondingly.

The partition of things into those in the *domain* and those in the *environment* determines the scope of analysis. The behavior of the domain will be analyzed in terms of its state changes where the state will be defined by a set of state variables which reflect domain aspects of importance to stakeholders. In particular, the interaction between the domain and its environment will be manifested by state variables representing mutual properties.

This conceptualization may be further specialized to clarify business domain dynamics through the notion of *business process*, which is defined as a sequence of *unstable states*, leading to a *stable state* reflecting the process's *goal*. More broadly, GPM defines a *process model* in terms of a quadruple $< S, L, I, G >$, where S is a set of states, L a set of transition laws over S, I is a set of initial unstable states, and G a set of stable goal states. The goal is a set of stable states defined by some business conditions, denoting what the process is intended to achieve. Such a goal is termed a *hard-goal*. In addition, soft-goals are functions defining order relations over states in the goal set. In particular, soft goals can reflect performance measures by which different executions of the process can be evaluated.

Finally, we relate the above concepts to the common concepts of actor and role used in analyzing business domains. First, an *actor* will be anything in the domain (or its environment) whose internal state changes (due to its own actions) can cause state changes in other things in the domain. Consider a scenario in which similar actors that are modeled by the same functional schema (same state definition) and are subject to the same transition laws. These things can exhibit the same state changes and can cause the same changes in other things (manifested by the same mutual state variables in the functional schema). The functional schema and the transition laws of these things comprise the definition of a *role*.

IV. Integrating three Models: WaaS, BE, and GSM

To integrate the three organizational concerns—business, operations, and work-execution—three base models have been employed to construct WaaSaBE as a unified solution. First, we use WaaS [5] to conceptualize consumer-to-vendor engagements. In WaaS, interaction between requestor and provider is encapsulated such that how the work is done by the provider need not be shared. Second, we use Business Entities (BE) (*a.k.a.* Business Artifacts) [8] to conceptualize the organizational domain as being composed of key conceptual entities. These entities are central to guiding the operations of a business and possess content that changes as they move through those operations. The BE model has been found effective in specifying high-level business operations as expressed by business people. Third, and complementary to the BE model, to represent the execution aspects of BEs we use the Guards-Stages-Milestones (GSM) grammar [6], [7]. This is a declarative, lifecycle specification grammar developed to specify behavioral aspects of BEs as an alternative to other conventional BPM techniques.

Here we specify the set of constructs and relationships for each of the above grammars in the form of a UML meta-model (namely the 'abstract syntax'), and also use the ontology-based theoretical framework described in the previous section to define the business semantics of the key constructs.

A. The Work-as-a-Service model

A meta-model illustrating the constructs and relationships in the WaaS model is illustrated in Fig. 3. The fundamental construct is the *WaaS Type*, which specifies possible work engagements between *requestors* and *providers*. Such engagements are aimed to achieve a predetermined outcome, subject to a given input and the preservation of a set of constraints. This specification of a WaaS type is further elaborated by two subcomponents: *payload information*, specifying essential information about the work to be accomplished (e.g., required input), and *coordination (or assurance) information*, specifying all other information to be monitored and enforced, such as quality, cost, and dependencies on other WaaS type executions (e.g., temporal precedence). The specification of a WaaS type may also include a possible specification of a *service scheme*, which is a set of service calls between the requestor and the provider, expected to progress the work from initiation to fulfillment.

The enactment of a WaaS unit type yields a *WaaS-instance*, which is a specific engagement matching an individual requestor and an individual provider through a sequence of interactions leading to the completion of work.

These concepts can be clarified using the ontological terms of Section III.

Definition 1: A *work-execution* $w =< i, g >$, (or simply *work*) is a *state transition* from the *initial state i* of a given domain (i.e., a set of state variables) to a *goal state g* in that domain.

We distinguish the above definition from the broader notion of arbitrary work-execution, which may terminate in an arbitrary state and is not necessarily pre-specified as a business goal. Correspondingly, we define:

Definition 2: A *work-type* W is a set of possible work-executions in a given domain, specified as $< Payload, Coordination >$ such that:

- $Payload =< Initial, Goal >$ is defined as a set of possible work-executions, each transforming between a given state $s \in Initial$ and a state $g \in Goal$. This specifies what needs to be achieved from relative to the initial state but does not state how the work should be carried out.
- $Coordination$ is defined as a set of functions and information that is used to monitor the provider's work executions. In particular, this information can be used to establish order relations over the goal states in $Goal$. Optionally, it can also satisfy constraints over values of these functions, so states must satisfy these constraints to be in the goal.

As an example of an order relation function, consider product quality measured by defect rate for states in which products have been delivered. Clearly, a state in which the defect rate is 0.01% is more desirable (higher ranked) than a state in which the defect rate is 3%, but both states are in the process goal. A constraint included in the coordination information can specify that states in which the defect rate is over 5% are not acceptable as goal states.

Fig. 4 is an illustration of work-execution and work-type, with respect to the two states (i.e., initial and goal). This conceptualization of work does not specify existence and type of actors involved in execution.

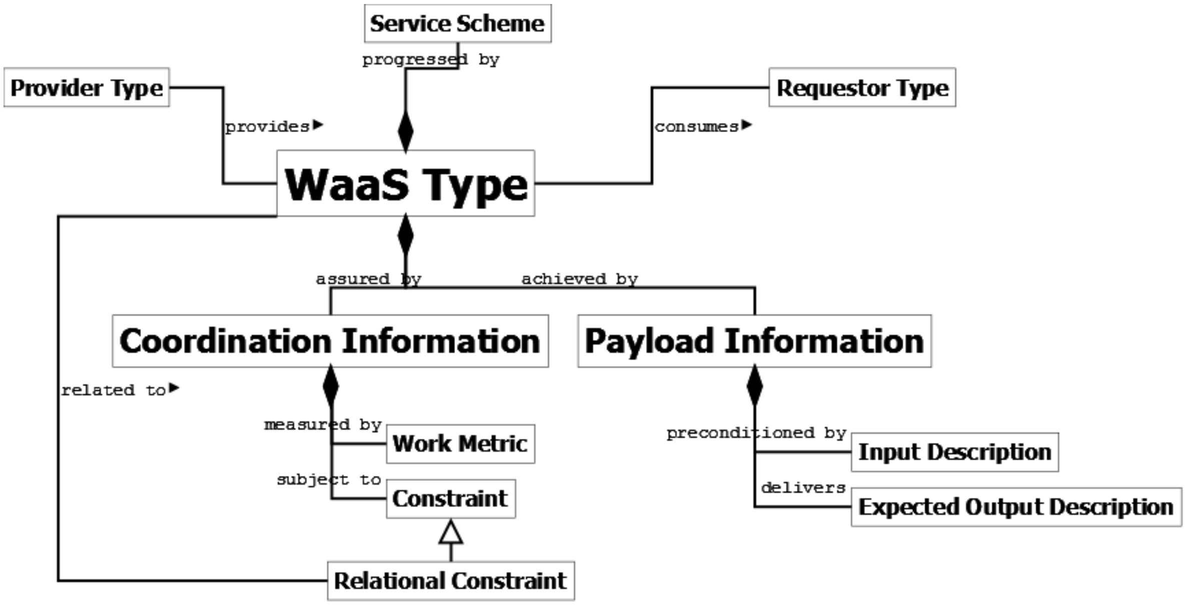

Fig. 3 - WaaS Meta Model

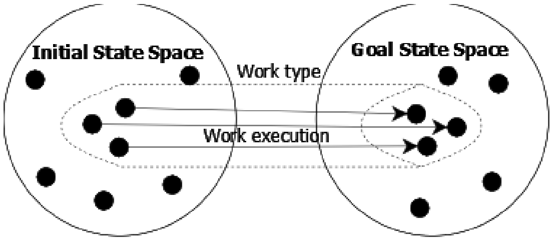

Fig. 4 - work and work-execution

Next, we expand our conceptualizations to include additional components in the contractual domain and more concisely describe work within the explicit context of possible interactions among providers and consumers in the business domain. Specifically, we define:

Definition 3: A *commitment c* is an initial *mutual state i* of two *actors*, one of which (the provider) indicates it will act to change the mutual state to a certain *hard goal state g*.

Note that the state is termed *mutual* as the action committed to by provider is expected to affect the state of the other actor (namely, the consumer), and this will be manifested by values of mutual state variables.

Correspondingly, we define:

Definition 4: A *commitment type* is a set of commitments specified by the *roles* of the actors involved and some conditions on the initial and goal states.

Using the above definitions, we can deduce the conceptualizations of *WaaS instance* and *WaaS type* to be as follows:

Definition 5: A *WaaS-instance* is the establishment of a specific commitment between a specific provider and a specific consumer for the fulfillment of a concrete work-execution.

Correspondingly:

Definition 6: A *WaaS-type* is a specification of a concrete *commitment type*, in which the corresponding action is a specification of a *work-type*, including all details about the expected *roles* of the actors who may be involved in its execution and all related conditions about its accomplishment.

An important point is the meaning of execution of work 'as-a-Service'. Clarifying requires formalizing the notion of *service* using ontological terms. Informally, a *service* is a type of a *commitment*, in which the requestor may be completely oblivious to how the bringing about of its goal state is being attained by the provider. Note the connection to the notion of *value co-creation* in services.

To better explain how the two notions differ, we further distinguish between two (sub) state-spaces, both being part of the mutual state-space between the two actors as illustrated in Fig. 5:

- The *work state-space* **-** includes all states in the domain of work, including all states being considered initial, intermediary, and goals by the requestor.
- The *coordination state-space* **-** all states in the domain of interactions between the actors reflecting the medium used for the purpose of message exchange between the two actors.

Once we distinguish between the two state-spaces, it is easier to synthesize the notion of a service. Specifically, using ontological terms and previous definitions, we define:

Definition 7: A *service* is a *commitment* type in which the two sub state-spaces (i.e., work and coordination) are exclusive, such that the requestor may affect the states in the coordination state-space, which as a result the provider will bring about the state of the work state-space to its requestor's goal state.

In principle, it is possible for the two state spaces to overlap (i.e., not being exclusive). This may happen in cases where the requestor has to take part in affecting the work in order to ensure its progression towards accomplishment (as opposed to just coordinating with the requestor). Such circumstances might be considered sometimes a reduction in service quality. For example, if a customer of an ISP Company has to modify herself the configuration of her own router in order to fix connectivity, this might be well acknowledged as poor service quality.

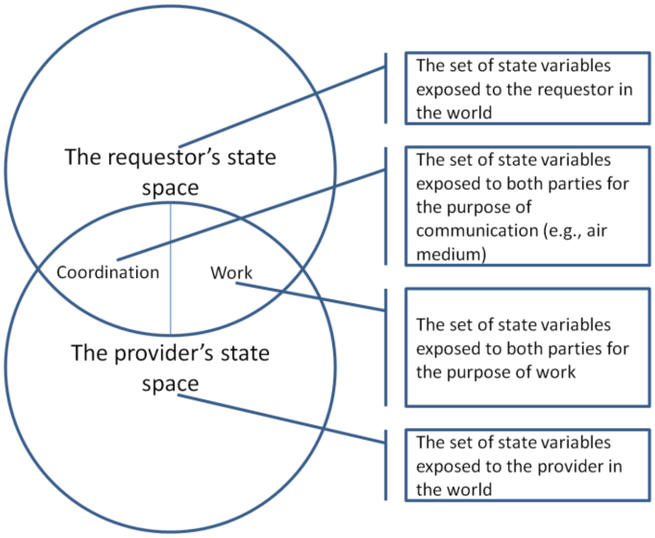
The requestor's state space
Coordination
Work
The provider's state space
The set of state variables exposed to the requestor in the world
The set of state variables exposed to both parties for the purpose of communication (e.g., air medium)
The set of state variables exposed to both parties for the purpose of work
The set of state variables exposed to the provider in the world

Fig. 5 - Mutual state space

B. The Business-Entity model

With respect to its modeling purpose, we use BE as a building block to express the *business* perspective – what happens in the provider business when a work-type instance is executed. Previous experiences with the BE model reveal that its core conceptualizations can serve as "first class citizens" for modeling, specifying, and implementing business processes and services, providing a shared vocabulary for multiple stakeholders, and facilitating understanding and communication among them [19–21].

Fig. 6 is a meta-model illustrating the constructs and relationships in the BE meta-model.

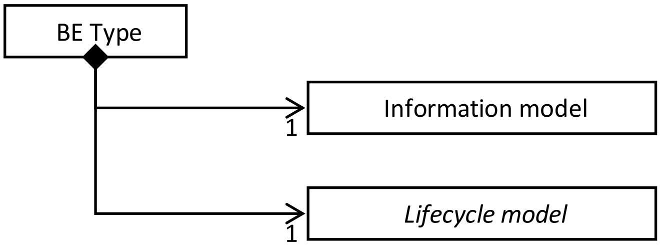

Fig. 6 - A Business-Entity meta-model

The fundamental concept in the BE model is a *BE type*, specified as a key conceptual dynamic entity (or object) that arises in and flows through a portion of the operations of a business [8], [22–25]. As opposed to BPM process models, the BE view is not that of a "linear" process and may involve different business processes needed to accomplish a business goal. As illustrated, in its very core, the internal form of a BE type is a marriage of process (life-cycle model) and data (information model), such that its execution (namely, a BE instance) has access to all relevant information that has been created by the business as the instance moves through the business operations.

More precisely, a BE type specification comprises an *Information model* and a *Lifecycle model*. An *Information model* is typically further classified into two subsets of attributes—*data attributes*, which hold all business-relevant data about the instance as it moves through the business, and *status attributes*, which hold information indicating the current 'phase' in the lifecycle the instance is at, reflecting a set of states of the business domain. A *Lifecycle model* describes the possible ways an entity of that type might progress through a business by responding to events and invoking services, including human activities. Various modeling grammars may be used to describe the lifecycle part in the specification of a BE type. Two common grammars employed in previous work are Finite-State-Machines and the Guards-Stages-Milestones model, which is described in the next section.

Our main objective in this section is to provide a definition using the ontological terms to clarify the above notion of a BE type in the context of the business domain. In essence, we also consider a BE type to be strongly associated with the notion of a *commitment* defined in the previous section. Specifically, we define:

Definition 8: A *Business-Entity-instance* is a specific realization of a *commitment*.

Correspondingly:

Definition 9: A *Business-Entity-type* denotes a set of possible realizations for a *commitment type*.

A direct relationship between the notions of a WaaS-type and a BE-type is already apparent in the above definitions. The specification of the former requires existence of the latter. Such conceptual relationships are in the roots of WaaSaBE as a uniform model, which we address in Section V.

C. The Guards-Stages-Milestones model

GSM was developed in recent years as a declarative approach to specifying the lifecycles of Business Entities [6], [7], [26]. It is well-suited for specifying the *operational* perspective.

A meta-model illustrating the core constructs and relationships in the GSM model is illustrated in Fig. 7. As illustrated, the GSM model itself should be considered a special type of BE lifecycle model. In its root structure, the model comprises a set of hierarchical *stages*, each designating possible changes between a set of pre-conditions ('*guards*') and a set of goals ('*milestones*'). As opposed to traditional process models, the specification of each stage

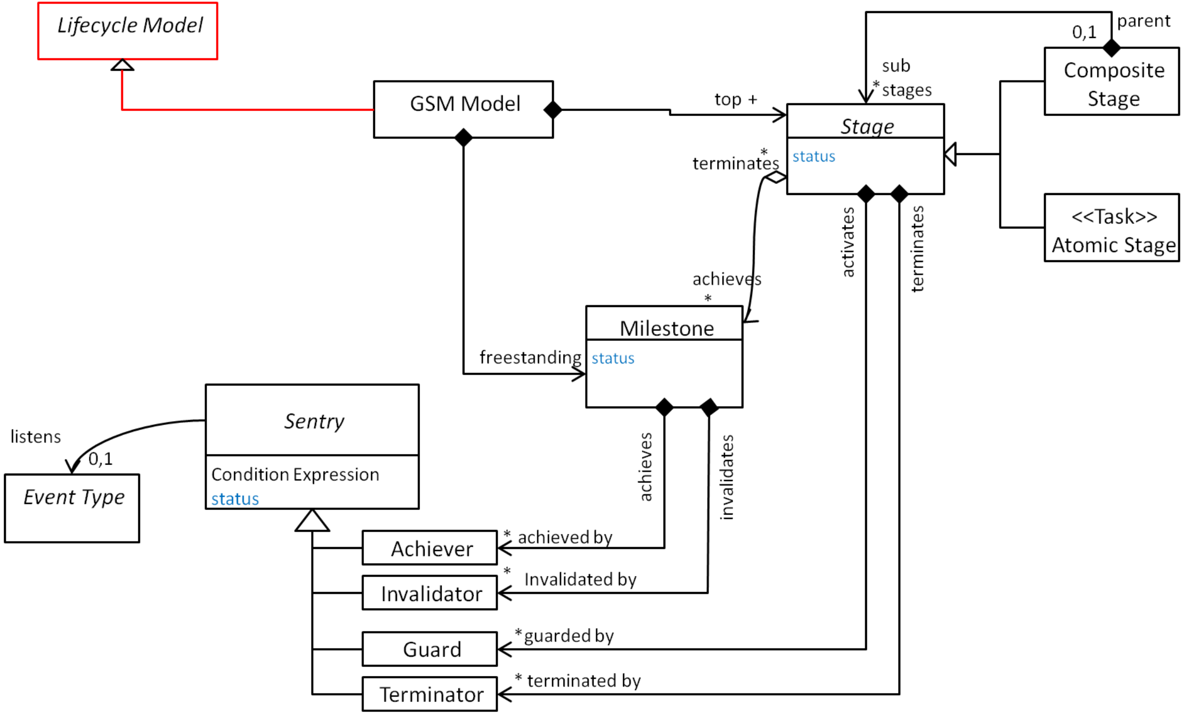

Fig. 7 - GSM-core meta-model

with respect to its sub-stages is not necessarily deterministic in the order of its enactment. This includes various aspects that enable flexibility in the execution of stages. For example, each individual stage may be associated with a set of guarding conditions and a set of milestones, such that various execution paths may be enacted and correspondingly terminate the enactment of a stage. Additionally, the set of sub-stages for any given stage may not necessarily be enacted in a predefined order (sequence). This kind of behavior is well in line with emerging paradigms such as *case-management* and *flexible process modeling*, e.g., [27].

The enactment of a specific GSM instance is triggered by *events* being listened-to by four types of *sentries*, which can conditionally change the status of milestones and stages. Events may include either external occurrences such as user activities or incoming service calls, or internal events such as stage termination or milestone achievement. Finally, each tree branch in the hierarchy has an *atomic stage* in its end (namely, a *task*). Possible tasks may include service calls, value updates to attributes in the information model of the corresponding BE instance, and activities to be performed by external human actors.

The constructs in the GSM model are conceptually aligned with the ontological view adopted above. Considering the GSM model in the context of a BE type (i.e., specifying its lifecycle), the corresponding BE's *information model* represents a set of state-variables (i.e., the BE's scope that reflects domain aspects relevant to stakeholders) over which the GSM model operates. Correspondingly, some of the constructs in the GSM model have almost intuitive mappings to the ontological concepts. This includes the following:

Definition 10: A GSM-model is equivalent to GPM's *process-model*, such that:

Definition 11: A *GSM-stage* is a *work-type* (Definition 2).

Definition 12: A *GSM-milestone* is a *hard goal* of a stage (namely, it is a state of the domain).

Definition 13: A *GSM-event* is equivalent to the *event* concept in Bunge's ontology (i.e., an ordered pair of states).

Definition 14: A *GSM-sentry* is (an implementation of) a transition-law, having a typical form of an ECA rule <*event, condition, action*>, in which *event* is a GSM-event, *condition* is an expression specified in the form of a *state law*, and *action* is a state-transition that depends on the specific type of the sentry. Specifically, four different sentry types in the GSM model are each associated with a different action. They can be described using the ontological terms as follows:

Definition 14.1: A *GSM-achiever* is a *sentry* whose action specifies change to one or more state variables of a corresponding *GSM-milestone*, transforming its current unstable state to one of its *stable goal states.*

Definition 14.2: A *GSM-invalidator* is a *sentry* whose action specifies a change to one or more state variables of a corresponding *GSM-milestone*, transforming its current goal state back to an *unstable state* (i.e., not one of its *goal states*).

Definition 14.3: A *GSM-guard* is a *sentry* whose action specifies a change to one or more state variables of a corresponding *GSM-stage*, transforming its current stable goal state to an unstable state in the $Initial$ set of states, thus triggering its initiation.

Definition 14.4: A *GSM-terminator* is a *sentry* whose action specifies a change to one or more state variables of a corresponding *GSM-stage*, transforming its current unstable state to one of its stable goal states.

V. "WAASABE" – A UNIFIED MODEL

In this section we describe how we merge the three models presented in the previous section to construct a unified model that is conceptually equipped to describe and synchronize among the triad of concerns: business, operations, and work-execution. To this aim, we analyze the conceptual ties that bind the three models together. To ensure that our analysis covers all possible aspects of all possible inter-model relationships, we systematically conduct the analysis over all possible inter-model combinations: (1) the WaaS model vis-à-vis the BE model, (2) the GSM model vis-à-vis the WaaS model, and (3) the BE model vis-à-vis the GSM model, explained above.

A. *WaaS vis-à-vis BE*

Both WaaS and BE models are centered around the notion of a business *commitment* (Definitions 5,6 and 7,8). Hence, each *commitment* identification in the business domain entails both the specification of a corresponding WaaS-type, and of a corresponding BE-type, such that the former is a realization of the latter. Both specifications should be made on the provider's side only. From a pragmatic perspective, the specification of a BE-type and its corresponding WaaS-type may be interpreted as follows: the WaaS-type is a contractual agreement with an external party in which its *payload* part details all functional requirements expected to be fulfilled by the corresponding BE-type, and its *coordination* part specifies all other requirements to be adhered to in the implementation of the BE-type.

B. *GSM vis-à-vis WaaS*

Although it may appear from the above analysis and the definition of a WaaS-type as if the relationship between BE-types and WaaS-types implies a simple one-to-one correspondence, the relationship between the GSM model and the WaaS model reveals an asymmetry in the notion of a WaaS-type towards its requestor and towards its provider, entailing a more complicated correspondence. Specifically, a WaaS-type has two complementary possible interpretations:

1. From a provider's perspective, WaaS-type reflects the specification of a commitment that is then realized by a BE-type.
2. From a requestor's perspective, WaaS-type may be considered a specification of an outsourced effort (i.e., a commitment of the other party involved in the engagement) that can be invoked by a service call.

Hence, a concrete GSM model specification may yield the specification of additional WaaS-types whenever the specification includes an atomic task that reflects an outsourced activity. In such circumstances, the WaaS-type will be specified from a requestor's perspective (i.e., the BE-type) who is expecting to be engaged with external vendors for the fulfillment of the corresponding atomic stage.

Aside from the above possible relationship between a WaaS-type and an atomic stage in GSM, we also designate a special case in which the organization wishes to expose some existing parts of its pre-specified operations (specified in GSM) as an independent service provider to external consumers. In such circumstances, the partial operation to be exposed may already be modeled as a sub-stage in an existing GSM model. To preserve both scalability and consistency in the unified model, we suggest that such circumstances would be handled as follows: (1) by creating a new BE-type that corresponds to the (partial) operation originally performed by the extracted sub-stage, (2) by refactoring the existing sub-stage into an atomic stage interact directly with the newly created BE-type, and (3) by creating a corresponding WaaS-type as the specification of the commitment associated with the extracted operation, being its provider.

Let us summarize all inter-model relationships constituting WaaSaBE as a unified model. Overall, analysis of an organizational unit using WaaSaBE would yield a specification of: (1) scope; (2) a set of business identified commitments of the organizational unit as a provider towards its business clients; (3) a set of WaaS-type specifications, each corresponding to a single commitment and detailing payload and coordination information; (4) a set of BE-types specifications, each designed as the realization of a WaaS-type in (3) and detailing both the BE's information model and lifecycle model in GSM; and (5) for each atomic-stage being an outsourced task in a GSM model, a corresponding WaaS-type specification in which the organizational unit is considered a requestor towards sub-contractors. This overall structure is illustrated in Fig. 8 in which the organizational unit being analyzed is an engine-design company. We further instantiate this example in the next section.

VI. Model Instantiation

Experience shows that lack of an integrated framework like the one proposed here can directly cause costly failures [1]. For example, in the case of aircraft design in the aerospace industry, the lack of an integrated framework has been blamed for the two-year delay and $6 billion in losses for the Airbus A380 as well as for the billions of dollars in losses in the design and deployment of the Boeing 787 Dreamliner, both of were designed using cross-enterprise collaboration.

In this section, we present an example in the domain of aircraft engine design as a baseline scenario to demonstrate the capability of the WaaSaBE model to cohesively express the desired triad of organizational concerns. The design of an aircraft engine is a complicated and large-scale undertaking, requiring innovation and depending on collaboration across multiple disciplines of the larger aircraft design industry, such as avionics, materials, hydraulics, electrical engineering, and embedded software. The design process uses a partner collaboration development chain that introduces strong dependencies among different organizations, which may have varying internal processes, data, tools, and policies. As such, this case study provides a proof point for typical projects, which in many cases are simpler than aircraft engine design.

Given the overall operational plan for the design, the high-level system integrator, for example, might request an engine company to provide an engine design and impose several constraints.

- *Scope of analysis:* An engine company accepts a commitment to provide an engine design to the system integrator. That is, in this context the engine company is considered the *provider* and the system integrator is the *requestor* with a goal of obtaining an engine's design. In this case, we will use WaaSaBE to do an analysis from the engine company side.
- *Identified Commitment(s):*
  To provide engine design (hard goal) to the system integrator that has requested it.
- *WaaS specification(s): (provider's side)*
  For each of the identified commitments (note: for simplicity, we use only one above), specify a corresponding WaaS type:
  - *WaaS-type:* engine design
    Payload specification: Description of all functional requirements expected to be provided such as power, fuel efficiency, shape, hydraulic interfaces, sensor interfaces, etc.

    Coordination specification: Description of all milestones, requirements and constraints to be monitored and fulfilled, such as design cost, manufacturing cost, maintenance cost, design schedule, warranty, etc.
- *Business Artifacts:*
  For each WaaS-type above, specify a corresponding BE-type:
  - *BE*-type: Designing-an-engine—this would be a BE-type aimed to specify how an engine design process may be produced. Specifically, we use the GSM's grammar to detail its production:
    GSM-specification: Describe a specific hierarchy of stages/milestones leading to the creation of an engine design plan. The following is given as an extreme simplification of the actual process borrowed from [28], only detailing only one possible stage hierarchy:
  - Preliminary studies:
    - Choice of cycle, type of turbomachinery layout.
    - Thermodynamics design point studies.
  - Aerodynamics of compressors, turbine, inlet, nozzle.
  - Mechanical design [outsourced]
  - Detail design and manufacture.

  As marked above, the engine company has made the choice to delegate the responsibility for the creation of the mechanical design to a subcontractor. Hence, in the context of such engagements, the engine company is considered a *requestor* and each subcontractor is considered a *provider* as illustrated in Fig. 8. Correspondingly, each such engagement yields a new WaaS-type specification as follows.

- *WaaS specification(s): (consumer's side)*
  For each GSM-stage above that represents an activity to be outsourced, a corresponding WaaS-type specification is generated:
  - WaaS-type: engine mechanical design
    Payload specification: detailed mechanical design requirement that includes stressing of discs. blades. casting. vibration. Whirling, and bearings.
    Coordination specification: cost, etc.

The WaaSaBE model gives a boundary object that moves across functional tiers and can be understood in local terms within the tiers.

In particular, since coordination information is directly encapsulated in terms of business commitments being realized as key business entities, it is readily understandable in the business tier. Moreover, since the payload information within the encapsulated WaaS specifies the specific details of the work to be done, it is readily understood for work execution, whether by electrical engineers, mechanical engineers, or other domain practitioners. Finally, since the complex coordination of work is gradually decomposing the overall business lifecycle into restricted state transitions, abstracted as stages attaining internal milestones, operational managers charged with planning and scheduling the aircraft design can readily understand.

VII. Future Applications

The model presented in this paper lays down core conceptual foundations that can serve as the basis for the systematic identification, coordination, and execution of collaborative business engagements. Specifically, WaaSaBE pertains to circumstances that require decentralization of resources and capabilities across and among organizational silos. Based on the theorized model, a wide range of infrastructural technologies (tools and techniques) are aimed to be developed to spread the paradigm of WaaS-based e-markets. This includes the following:

- WaaSaBE-based directory engines equipped to facilitate WaaS-based matching between providers and consumers. With respect to the aforementioned asymmetry in WaaS-types, this includes both the capability to publish work capabilities and demands.
- WaaSaBE-based execution engines being structured as a combination of a BE-based orchestration engine for managing the execution of BE's lifecycle (e.g., [29]), and a WaaS-based execution engine for managing the lifecycle of WaaS engagements.
- WaaSaBE-based modeling and design tool for the creation of models such as the aircraft engine example to be used as boundary object facilitating better communication and understanding in CeC settings.

Aside from the technological directions listed above, possible theoretical directions to explore include the following:

- Further enrich the core conceptual WaaSaBE model to be capable of expressing cross-organizational aspects such as analytic effort estimation, optimization,

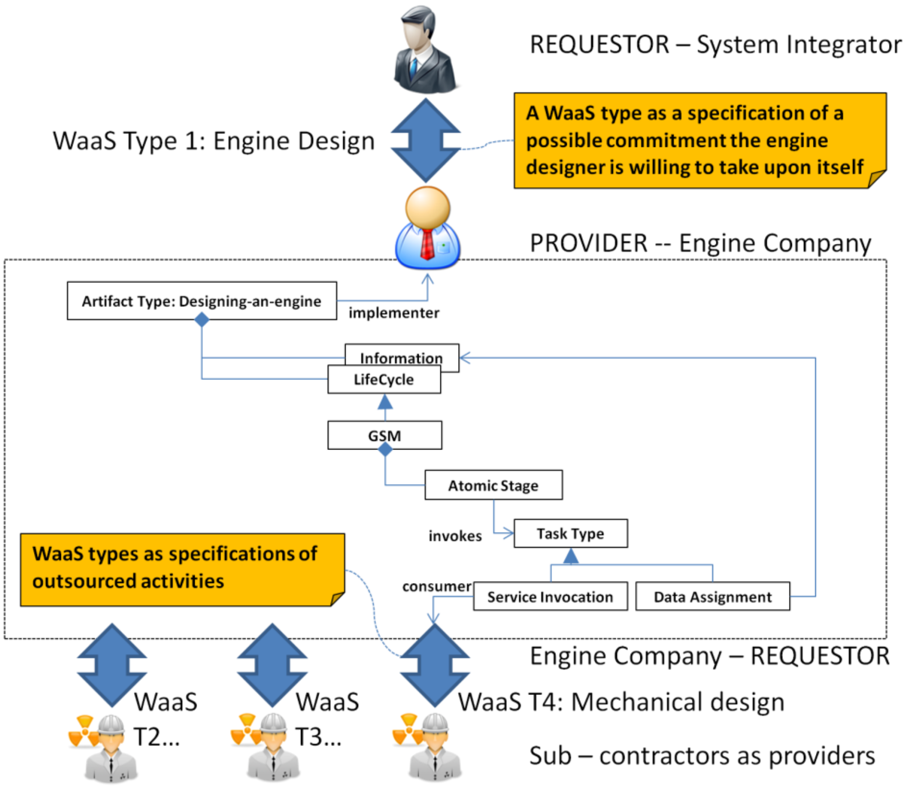
REQUESTOR – System Integrator
WaaS Type 1: Engine Design
A WaaS type as a specification of a possible commitment the engine designer is willing to take upon itself
PROVIDER -- Engine Company
Artifact Type: Designing-an-engine
implementer
Information
LifeCycle
GSM
Atomic Stage
invokes
Task Type
WaaS types as specifications of outsourced activities
consumer
Service Invocation
Data Assignment
Engine Company – REQUESTOR
WaaS T2…
WaaS T3…
WaaS T4: Mechanical design
Sub – contractors as providers

Fig. 8 - WaaSeBE - relationships between WaaS, BE, and GSM

governance, security, and also meta-information underlying design decisions.

- Identify and produce WaaSaBE domain specific repositories including prototypical WaaS types enabling re-use and more effective analysis.
- Use an existing service-delivery approach as a testbed for further refinement of the WaaSaBE model. Specifically, decompose and transform current methods for delivery of pre-packaged solutions (e.g., IBM's Business Analytics and Optimization service) as actual use cases to be represented. This effort will be aimed to further test and improve the capacity of the WaaSaBE model to handle realistic business circumstances and decomposition of work. Additionally, use such experience to conclude methodological principles for effective use of the model.

Focusing on the contractual domain, we envision the WaaSaBE model providing the conceptual basis for future system architectures and techniques that will put special attention to the management and execution of WaaS engagements, governance of such contractual commitments while being executed, and integration with traditional business operations technology, such as business-process and workflow engines.

References

[1] L. R. Varshney and D. V. Oppenheim, "On cross-enterprise collaboration," *in Business Process Management*, 2011.

[2] T. H. Davenport, *Process innovation: reengineering work through information technology*. Harvard Business School Press, 1993.

[3] M. Hammer, "Reengineering work: don't automate, obliterate," *Harvard business review*, vol. 68, no. 4, pp. 104-112, 1990.

[4] V. Grover and W. Kettinger, *Business process change: Reengineering concepts, methods and technologies*. IGI Publishing Hershey, PA, USA, 1995.

[5] D. Oppenheim, L. Varshney, and Y.-M. Chee, "Work as a Service," in *Service-Oriented Computing*, vol. 7084, G. Kappel, Z. Maamar, and H. Motahari-Nezhad, Eds. Springer Berlin / Heidelberg, 2011, pp. 669-678.

[6] R. Hull et al., "Introducing the Guard-Stage-Milestone Approach for Specifying Business Entity Lifecycles," in *Proc. of 7th Intl. Workshop on Web Services and Formal Methods (WS-FM 2010)*, 2010.

[7] R. Hull et al., "A Formal Introduction to Business Artifacts with Guard-Stage-Milestone Lifecycles." .

[8] R. Cohn, D. and Hull, "Business artifacts: A data-centric approach to modeling business operations and processes," *IEEE Data Eng. Bull*, vol. 32, no. 3, pp. 3-9, 2009.

[9] S. L. Star and J. R. Griesemer, "Institutional ecology,'translations' and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39," *Social Studies of Science*, vol. 19, no. 3, pp. 387-420, 1989.

[10] K. Masri, A. Gemino, and D. Parker, "Combining Diagrams to Enhance Understanding: Forging a Common Language for Different World Views Combining Diagrams to Enhance Understanding: Forging a Common Language for Different World," *Information Systems*, 2009.

[11] S. Alter, "Service system fundamentals: Work system, value chain, and life cycle," *IBM Systems Journal*, vol. 47, no. 1, pp. 71-85, 2008.

[12] S. Alter, *The work system method: Connecting people, processes, and IT for business results*. Work System Press, 2006.

[13] P. Soffer and Y. Wand, "Goal-Driven Analysis of Process Model Validity," 2004, vol. 4.

[14] M. Bunge, "Treatise on basic philosophy (Vol. 3)," *Ontology I: The Furniture of the World, Boston: Reidel*, 1977.

[15] M. Bunge, "A world of systems, Treatise on basic philosophy, Vol. 4," *Reidel Publ.Company, Dordrecht*, 1979.

[16] Y. Wand and R. Weber, "On the ontological expressiveness of information systems analysis and design grammars," *Information Systems Journal*, vol. 3, no. 4, pp. 217-237, 1993.

[17] Y. Wand and R. Weber, "On the deep structure of information systems," *Information Systems Journal*, vol. 5, no. 3, pp. 203-223, 1995.

[18] Y. Wand and R. Weber, "An ontological model of an information system," *IEEE Transactions on Software Engineering*, vol. 16, no. 11, pp. 1282-1292, 1990.

[19] K. Bhattacharya, N. S. Caswell, S. Kumaran, A. Nigam, and F. Y. Wu, "Artifact-centered operational modeling□: Lessons from customer engagements," *International Business*, vol. 46, no. 4, pp. 703-721, 2007.

[20] T. Chao et al., "Artifact-based transformation of IBM Global Financing," *Business Process Management*, p. 261--277, 2009.

[21] M. D. Bordella, R. Liu, A. Ravarini, F. Y. Wu, and A. Nigam, "Towards a Method for Realizing Sustained Competitive Advantage through Business Entity Analysis," *Enterprise, Business-Process and Information Systems Modeling: 12th International Conference, BPMDS 2011, and 16th International Conference, EMMSAD 2011, Held at CAiSE 2011, London, UK, June 20-21, 2011. Proceedings*, vol. 81, p. 216, 2011.

[22] A. Nigam and N. S. Caswell, "Business artifacts: An approach to operational specification," *IBM Systems Journal*, vol. 42, no. 3, pp. 428-445, 2003.

[23] R. Hull, "Towards Flexible Service Interoperation Using Business Artifacts," *2011 IEEE 15th International Enterprise Distributed Object Computing Conference*, pp. 20-21, Aug. 2011.

[24] P. Nandi et al., "Data4BPM, Part 1: Introducing Business Entities and the Business Entity Definition Language (BEDL)." IBM developerWorks, Apr-2010.

[25] R. Hull, "Artifact-centric business process models: Brief survey of research results and challenges," *On the Move to Meaningful Internet Systems: OTM 2008*, pp. 1152-1163, 2008.

[26] E. Damaggio, R. Hull, and R. Vaculín, "On the equivalence of incremental and fixpoint semantics for business artifacts with guard-stage-milestone lifecycles." 2011.

[27] W. Vanderaalst, M. Weske, D. Grunbauer, W. M. P. van der Aalst, and D. Grünbauer, "Case handling: a new paradigm for business process support," *Data & Knowledge Engineering*, vol. 53, no. 2, pp. 129-162, May 2005.

[28] J. D. Mattingly, W. H. Heiser, and D. T. Pratt, *Aircraft engine design*, vol. 1. Aiaa, 2002.

[29] D. Cohn, P. Dhoolia, F. Heath, F. Pinel, and J. Vergo, "Siena: From PowerPoint to Web App in 5 Minutes," *Service-Oriented Computing--ICSOC 2008*, pp. 722-723, 2008.