Computer Standards & Interfaces 35 (2013) 277–293

Contents lists available at SciVerse ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Hierarchy-oriented modeling of enterprise architecture using reference-model of open distributed processing

Lam-Son Lê [a,*], Alain Wegmann [b]

[a] School of Computer Science and Software Engineering, Faculty of Informatics, University of Wollongong, New South Wales 2522, Australia
[b] School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne 1015, Switzerland

ARTICLE INFO

Available online 16 February 2012

Keywords:
Enterprise architecture
Enterprise modeling
Systems theory
RM-ODP
SEAM
Alloy

ABSTRACT

Modeling Enterprise Architecture (EA) requires the representation of multiple views for an enterprise. This could be done by a team of stakeholders that essentially have different backgrounds. One way to do this is to structure the model into hierarchical levels each of which can be of interest to just some, not all, stakeholders. Due to the multidisciplinary nature of EA, stakeholders simply cannot choose a single modeling approach, even a widely-recognized one, to build their enterprise model and make it viewable and understandable to the whole team. Developing a modeling framework that can be applied uniformly throughout the entire enterprise model and that can be used by all stakeholders is thus challenging. We based our work on the RM-ODP (Reference Model of Open Distributed Processing) – a standardization effort that defines essential concepts for modeling distributed systems, as well as ODP-related international standards/recommendations, to develop such a modeling framework that we call SeamCAD. This framework consists of a computer-aided tool and a language behind the tool for modeling EA in a hierarchical manner. SeamCAD makes RM-ODP applicable in the context of multi-level EA and consolidates the SEAM – a family of methods for seamless integration between disciplines.

1. Introduction

Enterprise Architecture (EA) captures the whole vision of an enterprise in various aspects regarding both business and information technology (IT) resources [47]. In EA, the goal is to align the business resources and IT resources to maintain or improve the competitiveness of the enterprise. EA is a discipline that analyzes the services offered by an enterprise and its partners to the customer, the services offered by the enterprise to its partners and the organization of the enterprise itself and of its IT. Making an EA project can, for example, help the enterprise gain more customers, reduce the operation costs or increase its agility. This can be done by better identifying the services that the enterprise provides to the customer, by removing the duplication and inconsistencies in business processes and/or information flow, by giving the management more IT-supported facts for facilitating decision making.

During an EA project, an EA team – typically a multidisciplinary team – develops an enterprise model that represents the enterprise, its environment and its internals. The representation of the enterprise can include various aspects such as the services offered by the enterprise, the IT systems, as well as their implementation in terms of business processes and IT applications. Working with a model is important. When making the model, the team develops an agreed and shared representation of the enterprise, its environment and its internals. They also define what the project needs to achieve.

Being motivated by the challenges of defining a modeling technique that can be applied uniformly across hierarchical levels of an enterprise model, we developed a toolkit and eventually came up with a language definition for modeling EA in a hierarchy-oriented manner. We called this framework SeamCAD. This work was part of (and actually consolidates) an EA methodology called Systemic Enterprise Architecture Methodology (SEAM) that has an established pedigree in the literature and consulting services. The underlying rationale of SeamCAD was originally established in SEAM [57,54], which is based on the Reference Model of Open Distributed Processing (RM-ODP) – a joint effort by ISO[1]/IEC[2] and ITU-T,[3] which provides a co-ordinating framework for the standardization of open distributed processing [23]. To be able to make diagrammatic representations capturing different aspects of EA and to put them together in a coherent model that is manageable in a computer-aided tool, we i) further refine the SEAM building blocks using viewpoint-specific modeling concepts of the RM-ODP; ii) formally define the well-formedness of enterprise models; iii) explicitly describe the traceability between model elements of an enterprise model by leveraging the viewpoint correspondences defined in RM-

* Corresponding author.
E-mail address: lelamson@gmail.com (L.-S. Lê).

[1] International Organization for Standardization.
[2] International Electrotechnical Commission.
[3] International Telecommunication Union.

0920-5489/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2012.01.008

ODP. To make the diagrammatic representations more precise, we annotate them with declarative specifications written in a formal language. We had a few applications of SeamCAD both in the industry and within the academic setting. We invited EA practitioners and our master's students to test the framework.

The rest of this article is structured as follows. Section 2 outlines the key principles of the SEAM method. Section 3 discusses the motivation for our work on modeling EA. Section 4 proposes how the concepts defined in Part 2 and Part 3 of RM-ODP can be combined to enrich EA models, especially on the aspects of business functions and information processing. Section 5 comes up with a meta-model and provides some insight into the SeamCAD toolkit. Section 6 presents the applicability of SeamCAD and the user's feedback we obtained from external practitioners and our master's students. Section 7 surveys related work. Section 8 concludes the article and points out our future work.

2. Systemic Enterprise Architecture Methodology (SEAM) and its RM-ODP foundations

SEAM is a family of methods for seamless integration between disciplines. SEAM for Enterprise Architecture is an enterprise architecture method that belongs to the change management category [59]. To simplify the discourse, we will use the term SEAM to designate SEAM for Enterprise Architecture in the remainder of this article. Enterprise architects can use SEAM, to develop an enterprise model, a model that represents the relevant features of the organization and its environment. These features depend on the nature of the project for which the model is necessary. They may span the markets in which the organization operates and the implementation of the IT systems that support its operations.

One of the key principles of SEAM is that EA modeling should be done in a systematic way across all hierarchical levels that are created by perceiving enterprises as hierarchical [52]. Note that this perception does not necessarily mean that enterprises are essentially hierarchical. However, we believe that modeling EA in this way has an advantage because people tend to reason in terms of hierarchy [7]. SEAM has another essential principle that is about how enterprise entities are interpreted and represented in an enterprise model. In SEAM, all entities are systematically treated either *as a whole* or *as a composite*, depending on the view [57].

SEAM takes its foundations from the RM-ODP. This standard defines a modeling infrastructure for complex distributed IT systems. RM-ODP is composed of four parts [23]. Part 1 is an overview of RM-ODP and is non-normative. Part 2 defines the fundamental concepts needed for modeling Open Distributed Processing systems. Part 3 presents an application of Part 2 for particular viewpoint specification languages (i.e. enterprise, information, computational, engineering, technology viewpoints). Part 4 is a partial formalization of the previous parts.

SEAM does not rely on the RM-ODP viewpoint specification languages but seek to extend basic modeling concepts defined in Part 2 in order to represent systems that span business and IT systematically [53,30]. The rationale behind this systemic principle was to uniformly apply the same modeling techniques regardless of the subject to be modeled (e.g. business or IT systems) and to have a relatively small set of heuristics for specific aspects of each subject [57].

RM-ODP Part 2 defines the terms *abstraction* and *atomicity*. It is specified that *fixing a given level of abstraction may involve identifying which elements are atomic*. SEAM has two different kinds of levels of abstraction: functional and organizational. In the functional hierarchy, the element that determines the level of abstraction is the action. In the organizational hierarchy, it is the object the modeler considers as a whole that determines the level of abstraction.

3. Motivation

3.1. Example

Let us consider an example of a bookstore whose management decides to provide the company's services via the Internet. The management has a goal to specify the services that the bookstore can provide its customers with and to describe how to implement them using business and IT resources. A book-selling market contains a `BookValueNetwork` and a `Customer`. The value network consists of three companies: a bookstore company named `BookCo` (responsible for the service of processing the orders placed by the customer), a shipping company called `ShipCo` (responsible for shipping the books ordered) and a publishing company `PubCo` (responsible for supplying the books that were ordered but not yet available in the inventory of the bookstore company). The departmental structure of the bookstore company shows two departments: one for coping with the purchasing data (`PurchasingDep`) and the other for managing an inventory of books (`WarehouseDep`). We might have an additional level showing the IT infrastructure of these departments.

Fig. 1 gives a simplified representation of the organizational structure and services in the bookstore enterprise context using an ad-hoc notation. A regular rectangle represents a business entity or an IT system. A rounded rectangle can be attached to a regular rectangle to represent the main service offered by the business entity or the IT system drawn under the regular rectangle. The smile symbol stands for people. The lines connecting these entities and people denote the containment hierarchy. In this project, the EA team needs to model the business entities, the IT systems (drawn under regular rectangles in Fig. 1) and their environment, the services offered to the customer by these entities, the company to company (and department to department) business processes, information flow and interaction between the IT system and a clerk who operates it and possibly the overall architecture of the IT system.

3.2. Challenges in modeling EA hierarchically

As exemplified in the Bookstore example presented in Subsection 1, modeling EA involves presenting multiple views of an enterprise that show multiples business entities, IT systems and the services they offer. One way to do this is to structure the model into hierarchical levels (e.g. market level, value network level, company level) each of which can be of interest of just some, not all, stakeholders. Due to differences in their background, the stakeholders may not want to use a single common modeling approach, even a widely-recognized one, to build the enterprise model.

Developing a modeling framework that can be applied uniformly throughout the entire enterprise model and that can be used by all stakeholders is challenging. Firstly, the framework should have a uniform approach for specifying the services (e.g. `mall Selling book`, `Processing order`) offered by business entities and IT systems and for describing their implementation across hierarchical levels. Secondly, the framework should allow the stakeholders to represent the service specification and the service implementation of multiple business entities and IT systems, even within the same hierarchical level (e.g. both `BookCo` and `ShipCo` are to be represented in detail). Thirdly, the services offered by those entities and systems should be expressed at different levels of granularity (e.g. `Selling book` is broken down into `Getting order`, `Payment` and `Delivery`). Fourthly, the modeling framework should maintain the well-formedness of the enterprise model and the consistency between different views opened by different stakeholders of the team (e.g. `BookCo` appears in multiple views of which one shows the value network level and another shows the company level). Table 1 summarizes this analysis.

The work presented in this article addresses the aforementioned challenges. It was initially developed as part of the SEAM method.

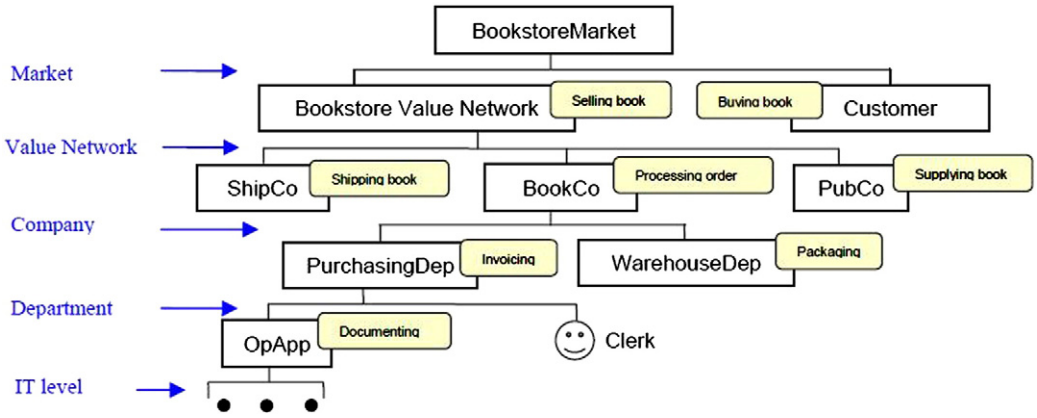

Fig. 1. Initial representation of the enterprise model of the Bookstore example.

Recent developments of RM-ODP in viewpoint-specific languages lead to an ISO/IEC recommendation that adds more modeling concepts to the enterprise language and gives it more expressive power in modeling EA [19]. Nevertheless, the RM-ODP and this recommendation address the challenges listed in Table 1 only implicitly. To make the point, let us try to model the organizational structure of EA using concepts defined in RM-ODP. In ODP, a community has a number of enterprise objects, which expresses the initial hierarchical level. To be able to represent additional hierarchical levels, we may turn some of these enterprise objects into communities. We need to repeat this practice several times in order to obtain a decent hierarchy. Moreover, in RM-ODP we might not have explicit correlations between EA services provided by a certain business entity and the interaction the entity takes part in. We aim to have a set of crisp modeling terms that permit reasoning about hierarchy and EA services in an explicit and systematic way by leveraging the modeling concepts of RM-ODP and the modeling principles of SEAM. At the same time, we recommend the use existing standards such as the UML for ODP system specifications [20] when it comes to the representation of software architecture and technology layers in EA.

4. Leveraging RM-ODP and SEAM

The RM-ODP Part 3 [22] specifies a viewpoint specification, namely the enterprise viewpoint specification or enterprise viewpoint for short, that is dedicated to the enterprise level of ODP systems. This specification particularly matters if we explicitly represent the context where the ODP systems exist. Part 3 of RM-ODP also specifies four other viewpoint specifications with dedicated modeling concepts to target other aspects such as information processing, computing capabilities and technology choices, etc. of the ODP systems being modeled. They are information viewpoint, computational viewpoint, engineering viewpoint and technology viewpoint. For each of these specifications, RM-ODP defines a language that comprises concepts, rules and structures.

The *enterprise language* defined in the original version of RM-ODP (published in 1995/1996) has a few concepts that address the structure and policy in an enterprise specification [23]. However, viewpoints-specific concepts for behavior modeling are still missing in this language. Recently, a number of ISO/IEC international standards and recommendations came along with RM-ODP. Most notably, the Enterprise Language recommendation extends and refines the original RM-ODP enterprise language by introducing more concepts to better capture enterprise processes [19]. The *information language* provides us with concepts to model the semantics of information and information processing for enterprise objects that may have been represented in the enterprise specification. The focus here is to develop a commonly-agreed understanding of information exchanged when they collaborate. Our goal is to leverage the concepts of these two languages and fundamental concepts defined in RM-ODP Part 2 to establish SEAM-inspired building blocks needed for modeling EA hierarchically (Subsection 1). We also establish bindings between building blocks based on the RM-ODP viewpoint correspondences (Subsection 2).

In RM-ODP, viewpoint languages are defined for the purpose of capturing one or more aspects of an ODP system and its environment. The definition of *ODP system* is given in Part 2, Section 3.2.4 which refers to the basic interpretation concept of *system* in Section 6.5 of Part 2. Systems in RM-ODP are usually computerized and capable of processing information. As such, they are comprised of subsystems. However, as suggested by a note in Section 6.5 of Part 2, we may interpret the term *system* in a broader scope such as in the systems theory [37] for modeling purposes. In this sense, we can safely assume that the term "ODP systems" also refers to IT-enable business entities found at various organizational levels (e.g. value network, company, department) of an enterprise [31] resulting in ODP enterprise & information viewpoint specifications being applicable to all business entities of interest in the enterprise context.

4.1. Building blocks

In RM-ODP, one can build modeling constructs by applying concepts defined in Part 2 to the modeling concepts of Part 3. We follow this principle in defining the building blocks of SeamCAD. Specifically, we apply the basic interpretation concepts and specification concepts

Table 1
Four challenges in modeling EA hierarchically.

Challenge	Brief description
Uniformness	To have a uniform modeling approach for capturing the specification and the implementation of services provided by all business entities and IT systems across hierarchical levels showing the organization of the enterprise
Multi-entity	To represent the service specification and the service implementation of multiple business entities and IT systems in the enterprise
Granularity	To model services of the business entities and the IT systems at different levels of granularity
Well-formedness	To maintain the well-formedness of the enterprise model and the consistency between its views

Table 2
ODP-based building blocks for modeling EA hierarchically.

Building blocks	RM-ODP concepts	
	Part 2	Part 3
Business/IT Working Object as a whole	Atomicity	(EV) Community Object
Business/IT Working Object as a composite	Decomposition	(EV) Enterprise Community
Human Working Object as a whole	Atomicity	(EV) Enterprise Party
Parameter Working Object as a whole	Atomicity	(EV) Enterprise Artifact
Parameter Working Object as a composite	Decomposition	(EV) Enterprise Artifact
Business Collaboration as a whole	Atomicity	(EV) Enterprise Step
Business Collaboration as a composite	Decomposition	(EV) Enterprise Process
Information Object as a whole	Atomicity	(IV) Information Object
Information Object as a composite	Decomposition	(IV) Information Object
Localized Action as a whole	Atomicity	(IV) Information Action
Localized Action as a composite	Decomposition	(IV) Information Action
Action Spec	non-applicable	(IV) Dynamic Schema

defined in Part 2 [21] to viewpoint-specific modeling concepts of the information language in Part 3 [22] and the extended enterprise language [19]. In Table 2, building blocks for SeamCAD are listed in the most left column. The sub-column in the middle lists specification concepts taken from RM-ODP Part 2. The most right sub-column is dedicated to concepts coming from RM-ODP Part 3 or the enterprise language recommendation. Each concept in this sub-column is prefixed by an abbreviation put in parentheses indicating which ODP viewpoint language the concept comes from.

As can been seen from Table 2, we systematically apply the interpretation concept of *Atomicity* and the specification concept of *Decomposition* (they both come from Part 2) to viewpoint-specific concepts. This application results in building blocks being viewed dually *as a whole* and *as a composite* – one of the key principles of SEAM (see Section 2). Exceptions apply to human working objects, which are to be seen only as a whole. Note that business working objects are viewed as a whole in a slightly different way than other building blocks are. Being seen as a whole,[4] they exhibit information objects and localized actions as their externally-visible properties. Other building blocks are simply atomic when viewed as a whole in SeamCAD. The externally-visible properties of a business working object are about information processing and information actions of the business entity being represented. They are actually captured by the IV concepts listed in Table 2.

The building blocks presented in Table 2 permit us to address three out of the four challenges listed in Table 1 as follows.[5] We will illustrate this standpoint in subsections that follow.

- *Uniformness*: we dually apply the two concepts Atomicity and Decomposition to EV concepts resulting in the same modeling techniques being systematically applied to all organizational levels.
- *Multi-entity*: we have EV and IV specifications of multiple entities of interest in the enterprise being modeled.
- *Granularity*: the application of the two concepts Atomicity and Decomposition to IV concepts permits the representation of multiple functional levels. For each functional level – which captures certain granularity of business entities and/or IT systems, we can express relationships between model elements shown at the functional level in question by leveraging the ODP viewpoint correspondences.

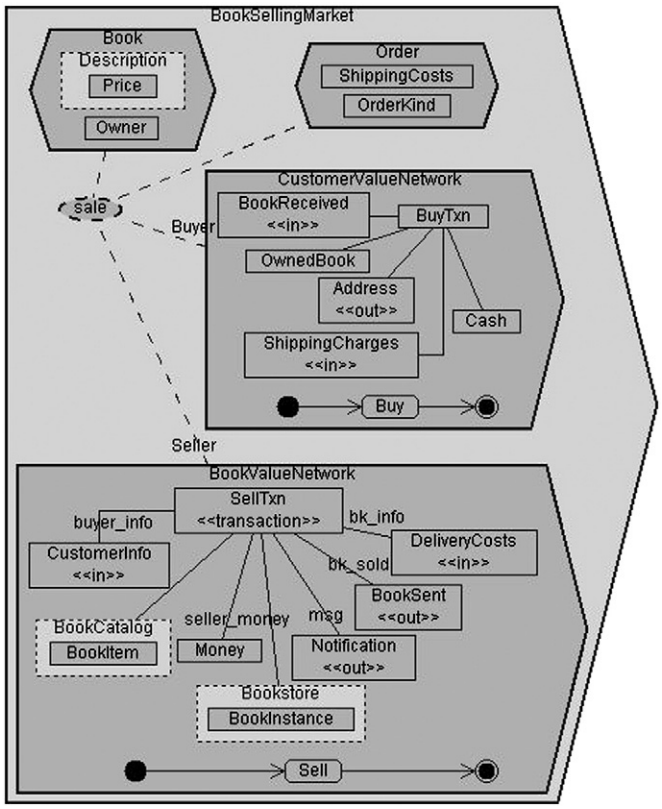

Fig. 2. A view showing the first hierarchical level of the Bookstore enterprise model.

4.1.1. Illustration of building blocks in the Bookstore example – first organizational level

Fig. 2 is a view of the Bookstore enterprise model that diagrammatically represents the first hierarchical level of the Bookstore enterprise model. `BookSellingMarket` is a business working object that is seen as a composite whilst its components are all viewed as wholes (they are business working objects `BookValueNetwork` and `CustomerValueNetwork`). In this view, hierarchical containment is visually expressed using nested pictograms. Diagrammatically, block arrow pictograms represent working objects; ellipse pictograms – collaborations; rectangle – information objects; rounded rectangles – localized actions.

Business collaboration `sale` takes place between business working objects `BookValueNetwork` and `CustomerValueNetwork`. It has `Book` and `Order` as parameter working objects. Being viewed as a whole, `BookValueNetwork` exhibits a localized action called `Sell` and a number of information objects that represent the money it has, the bookstore it operates, a catalog of books it offers, and information about customers as well as the order they have placed. `BookValueNetwork` as a whole has a special information object called `SellTxn`,[6] which represents information processing that is specific to the localized action `Sell`. It captures information items that are created and processed by localized action `Sell` within `BookValueNetwork`. However, these pieces of information, which may themselves be represented as component information objects of `SellTxn`, are not visible in this

[4] Throughout this article, we shall use the words "seen as", "viewed as" and "treated as" almost interchangeably.

[5] We will explain how we address the challenge of *Well-formedness* in Subsections 2 and 3.

[6] We apply a naming convention to this kind of information objects. An information object that represents action-specific information processing is named by appending the postfix *Txn* to the name of the localized action for which it expresses an "information processing *transaction*" (the postfix is actually an abbreviation of the word transaction).

view because SellTxn is seen as a whole (just like business collaboration sale and localized action Sell).

Similarly, CustomerValueNetwork is seen as a whole in this view. Intuitively, CustomerValueNetwork could be regarded as a group of people (e.g. a library, a family) who share interests in purchasing books. As such, it exhibits a localized action called Buy and a number of information objects that represent the cash it possesses, the book it owns, information about the order placed and a special information object called BuyTxn that represents t Buy-specific information processing. Some information objects of BookValueNetwork and CustomerValueNetwork are marked with stereotypes[7] <<in>> or <<out>> indicating they are information objects exchanged between the business working object in which they are defined and the environment.

Now, after having given diagrammatic representations of business working objects and their collaboration, we proceed in precisely capturing the semantics of information processing for localized actions Sell and Buy as well as business collaboration sale that is regarded as dynamic schema in RM-ODP. We use the building block *Action Spec* of Table 2 to complement these localized actions. We opt to use Alloy [24] for writing the spec of localized actions and business collaborations rather than Object Constraint Language (OCL) [41] for the following two reasons.

- Alloy is a declarative language that is capable of processing first-order statements while maintaining an Object-Oriented syntax. This allows us to declare working objects, information objects and formalize collaborations and actions all in a single module of Alloy code.
- The Alloy language comes with a tool called Alloy Analyzer[8] that first serves as a compiler to check if the Alloy code is syntactically correct. It then acts as an interpretor to verify if the code is semantically consistent (i.e. not over-constrained) and simulate it.

Fig. 3 is a code fragment that declares all business working objects and parameter working objects that are visible the view shown in Fig. 2. Another code fragment is given in Fig. 4. This code corresponds to the specification of localized actions Sell and Buy.

As we can see from Fig. 3, business working objects BookValueNetwork and CustomerValueNetwork, together with parameter working objects Book and Order, are declared using Alloy keyword sig. This keyword signifies Alloy *signature*, which is the counterpart of the *class* construct in object-oriented programming [3]. Each signature has a number of Alloy fields declared as sets that are accompanied by Alloy keywords specifying their cardinality (e.g. set and one). In some cases, we see a block structure right after the declaration of an Alloy signature. This block lists Alloy facts – invariants that matter on the very signature being declared.

The Alloy fields in the code fragment of Fig. 3 corresponds to information objects that belongs to the business/parameter working objects being declared by Alloy signatures. Some of them change state (e.g. bookstore and money in BookValueNetwork) before and after the execution of business collaboration sale. We introduce the notion of *time* in this code and we declare them as a mapping from a set to time.

In Fig. 4, localized actions Sell and Buy are coded as Alloy *predicates* sellAction and buyAction, respectively. An Alloy predicate is the counterpart of the *procedure* construct in structured programming [8]. Being a declarative language, Alloy allows us to make high-level processing statements in a computer-interpretable syntax of first-order logic [48]. These statements represent changes made to the information objects that are expressed as Alloy fields within Alloy signatures passed as formal parameters to a predicate. They are grouped into precondition and post-condition using Alloy

```
sig BookItem {
    price: Int
} {
    price >= 0
}

fact unique {
    all bk1, bk2: Book |
        bk1 != bk2 <=>
        bk1.description != bk2.description
}

lone sig BookSellingMarket { }

sig Order {
    wantedBook: one BookItem,
    shipping_costs: Int
} {
    shipping_costs > 0
}

sig Book {
    description: one BookItem,
    owner: String lone -> Time
}

lone sig BookValueNetwork {
    market: one BookSellingMarket,
    catalog: set BookItem,
    bookstore: Book set -> Time,
    money: Int one -> Time
} {
    all b: Book, t: Time |
        (b in bookstore.t =>
        b.description in catalog)
        and (b in bookstore.t => no b.owner.t)
    all s: catalog | some t: Time |
        some b: Book |
        b in bookstore.t and b.description = s
}

lone sig CustomerValueNetwork {
    market: one BookSellingMarket,
    address: one String,
    owned_book: Book lone -> Time,
    cash: Int one -> Time
} {
    all t: Time | int [cash.t] >= 0
}
```

Fig. 3. Alloy code for business and parameter working objects at the first organizational level of the Bookstore example.

comments. The parameters stand for the business working object involved and information objects exchanged during the execution of the localized action being expressed. The last two parameters pre and post represent the moments before and after this execution, respectively. A short predicate given at the end of this code fragment is dedicated to business collaboration sale. This predicate is simply stated as an *and* logical combination of the two previous predicates.

Using Alloy Analyzer, we simulate the code presented in Fig. 3 and Fig. 4 to generate a sample and have it visualized. Fig. 5 and Fig. 6 are the projections of this visualization over time. They show the states[9] of information objects *before* and *after* the execution of the localized actions, respectively.

Fig. 7 depicts another view of the Bookstore enterprise model that gives diagrammatic representation of the same working objects as

[7] In Unified Modeling Language, stereotype is the primary extension construct. We use this construct to express the information flow between business working objects [57].

[8] At the time this article was written, the team developing Alloy Analyzer had released its versions 4.x. They were available at http://alloy.mit.edu/alloy4/.

[9] The Alloy Analyzer tool chose an arbitrary notation for Fig. 5 and Fig. 6, which is unfortunately not analogous to the one used in SeamCAD (Figs. 2, 7, 8, 9 and 10).

```
pred buyAction [
    buyer: one CustomerValueNetwork,
    book_received: one Book,
    shipping_charges: one Int,
    pre: one Time, post: one Time] {
    // invariant
    post = ord/next [pre]
    // pre-condition
    no buyer.owned_book.pre
    int buyer.cash.pre >=
        int book_received.description.price
        + int shipping_charges
    // post-condition
    buyer.owned_book.post = book_received
    int buyer.cash.post = int buyer.cash.pre
        - int book_received.description.price
        - int shipping_charges
}

pred sellAction[
    seller: one BookValueNetwork,
    book_sent: one Book,
    customer_info: one String,
    delivery_costs: one Int,
    pre: one Time, post: one Time] {
    // invariant
    post = ord/next [pre]
    #seller.catalog > 1
    book_sent.description in seller.catalog
    all t: Time, b: Book | pre = ord/next [t] =>
        (b in seller.bookstore.t <=>
        b in seller.bookstore.pre)
    // pre-condition
    book_sent in seller.bookstore.pre
    no book_sent.owner.pre
    // post-condition
    book_sent.owner.post = customer_info
    seller.bookstore.post =
        seller.bookstore.pre - book_sent
    int seller.money.post = int seller.money.pre
        + int book_sent.description.price
        + int delivery_costs
}

pred saleBinding[bookseller: one BookValueNetwork,
        customer: one CustomerValueNetwork,
        bk: one Book, o: one Order,
        pre: one Time, post: one Time] {
    post = ord/next [pre]
    bookseller.market = customer.market
    bookseller.bookstore.pre = Book

    sellAction[bookseller, bk, customer.address,
                o.shipping_costs, pre, post]
    and
    buyAction[customer, bk, o.shipping_costs,
                pre, post]
}
```

Fig. 4. Alloy code for the spec of localized actions `Sell` & `Buy` and business collaboration `sale`.

the one in Fig. 2 does. The main difference between the two views is that Fig. 2 treats collaboration `sale` as a whole whilst Fig. 7 sees it as a composite. Being treated as a composite, `sale` has the following component collaborations.[10]

- `placeOrder`: `CustomerValueNetwork` places an order and `BookValueNetwork` receives it

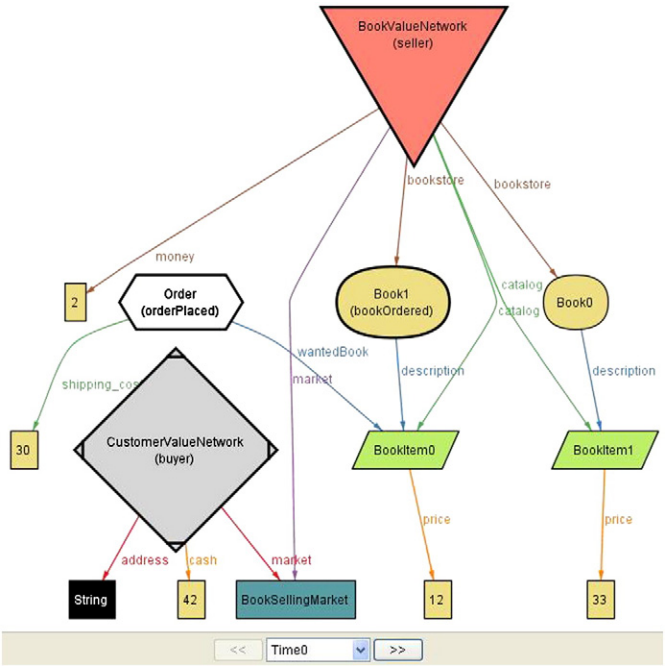

Fig. 5. States of information objects *before* the execution of `Sell` and `Buy`: `Book1` belongs to `bookstore` of business working object `BookValueNetwork`, which has 2 units of `money`; `CustomerValueNetwork` does not own any book and it has 42 units of `cash`.

- `payment`: `CustomerValueNetwork` makes a payment, `BookValueNetwork` verifies and receives it
- `deliver_notify`: `CustomerValueNetwork` receives either a book of which description is specified in the order if payment was successful, or a notification sent by `BookValueNetwork` otherwise.

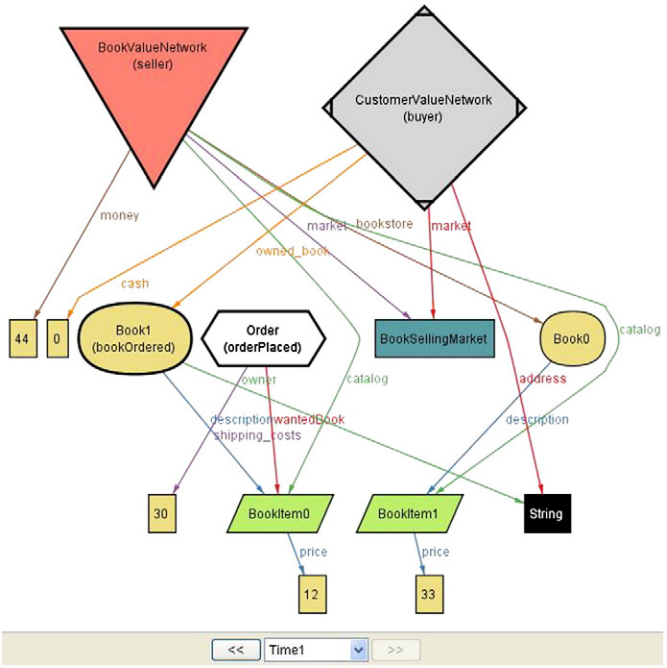

Fig. 6. States of information objects *after* the execution of `Sell` and `Buy`: `Book1` now belongs to `owned_book` of business working object `CustomerValueNetwork`, which has cash deducted to 0; `BookValueNetwork` no longer owns `Book1` and now has 44 units of `money`.

[10] We have an implicit naming convention for business collaborations and localized actions. The names of business collaborations usually start with a lower case character whereas the first letters used in naming localized actions are always put in upper case. Lexically, localized action names always start with a verb. Names of business collaborations commence with either a verb or a noun phase.

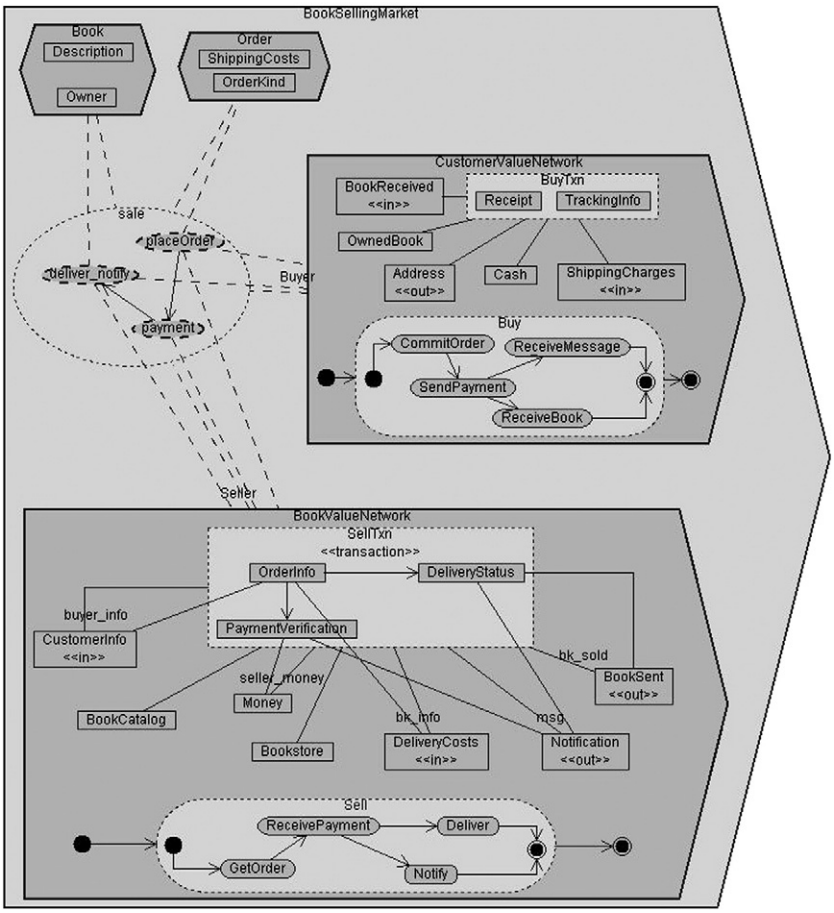

Fig. 7. Another view that shows the same business working objects as the one in Fig. 2 does. The two views differ largely in the way they see business collaboration `sale`.

In this view, solid arrows express the order in which these component business collaborations will take place. Also treated as composites in this view are information objects `Bookstore`, `BookCatalog` and `Description`.

To make the context of information processing clear, viewing business collaboration `sale` as a composite should lead to localized actions `Sell` & `Buy` and information objects `SellTxn` & `BuyTxn` all being viewed as composites because they express the information flow going between the two business value networks through business collaboration `sale`. As a result, component localized actions of `Sell` are made visible inthis view. They are `GetOrder` (obtaining information about the order placed by the customer), `ReceivePayment` (getting payment confirmation or failure notice), `Deliver` (shipping books if payment was successfully received) and `Notify` (informing the customer in case payment was unsuccessfully received).

Component information objects of `SellTxn` are also made visible. They are `OrderInfo` (information about the order the customer has placed, e.g. time, auction or a normal transaction), `PaymentVerification` (whether the payment is successful or failed) and `DeliveryStatus` (whether the book ordered has been posted or not, tracking information if posted, etc.). Note that information object `OrderInfo` is associated with two other information objects, namely `CustomerInfo` and `DeliverCosts`. On the side of business working object `CustomerValueNetwork`, component localized actions of `Buy` and component information objects of `BuyTxn` are made visible in a similar fashion. The following component localized actions of `Buy` are also made visible: `CommitOrder` (preparing and committing an online order for buying books), `SendPayment` (making payment which may or may not succeed), `ReceiveBook` (receiving books if payment was successful) and `ReceiveMessage` (receiving a notification message in case payment was unsuccessfully made).

It is feasible to write the specification of all component localized actions that are represented at this functional level. In our previous work, we made comprehensive Alloy code that specifies the spec of business collaborations and localized actions of the Bookstore enterprise model at various organizational levels and functional levels [29,56].

4.1.2. Illustration of building blocks in the Bookstore example – second organizational level

In the view of Fig. 2, if we toggle the viewing mode of business working objects `BookValueNetwork` and `CustomerValueNetwork` from being seen as wholes to being viewed as composites, we obtain yet another view of the Bookstore enterprise model as shown in Fig. 8. This view is made for the second organizational level.

In this view, both `BookValueNetwork` and `CustomerValueNetwork` are seen as composites. `BookValueNetwork` has four component business working objects representing a total of 3 cooperating companies within this businessvalue network: `BookCo` (responsible for processing orders and inventory management), `ShipCo` (dealing with warehouse and shipping operations), `PubCo` (supplying books) and `CreditCardAgent` (processingonline payments). We also see a counterpart of the company level for `CustomerValueNetwork`. It has three human working objects `Student`, `Businessperson` and `Professor`. Within `BookValueNetwork`, business collaborations, localized actions and information objects represent the information processing of and information exchanged between the companies.

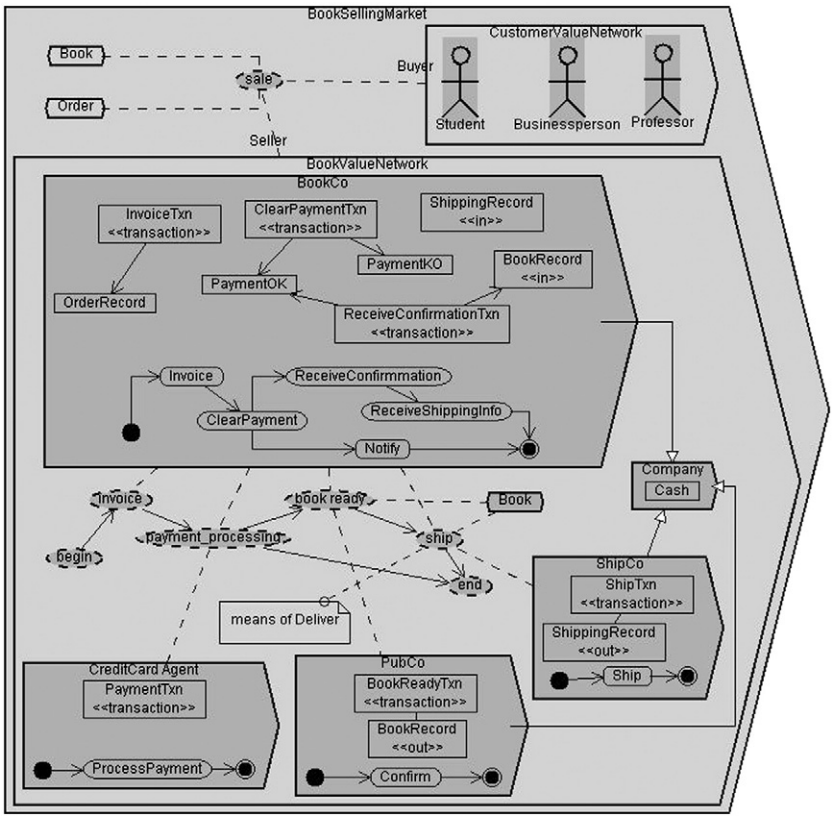

Fig. 8. A view that shows two consecutive organizational levels of the Bookstore model.

4.1.3. Illustration of building blocks in the Bookstore example – third & fourth organizational levels

In the view of Fig. 8, if we toggle the viewing mode of business working object `BookCo` from being seen as a whole to being viewed as a composite, we reach the third organizational level of the Bookstore enterprise model as shown in Fig. 9. In this view, she hides the environment of `BookCo` while describing its departmental structure. At this organizational level, `BookCo` has two component business working objects `PurchasingDept` and `WarehouseDept`. The former is responsible for processing invoices. The latter deals with warehousing. Collaborations between these two business working objects plus their localized actions and information objects are represented in the same manner as we have done to the other views presented previously.

As we deepen the organizational level hierarchy, we may reach the IT level of the Bookstore example. Fig. 10 gives another view capturing the fourth organizational level of the Bookstore enterprise model. In this view, `PurchasingDept` is seen as a composite showing the IT operations of the department represented by this business

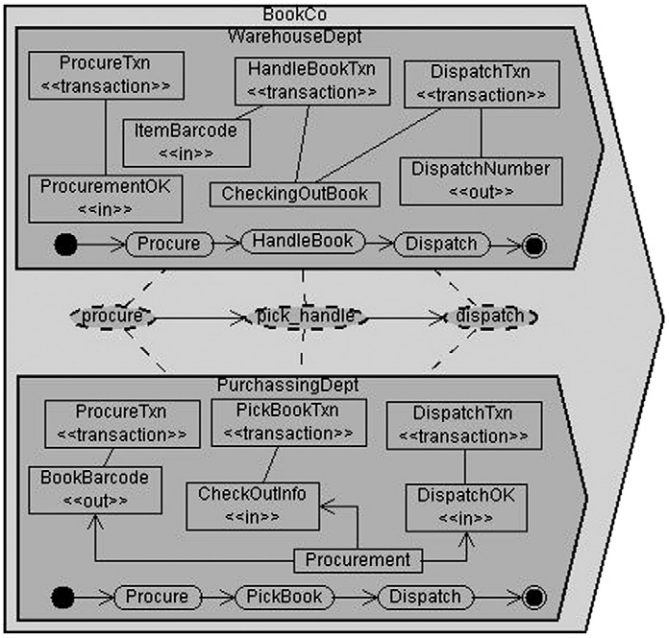

Fig. 9. A view that shows the third organizational level of the Bookstore enterprise model.

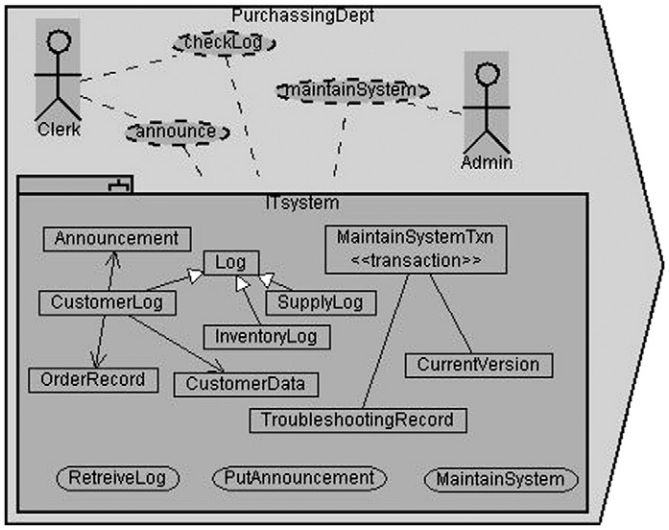

Fig. 10. A view showing the fourth organizational level of the Bookstore enterprise model.

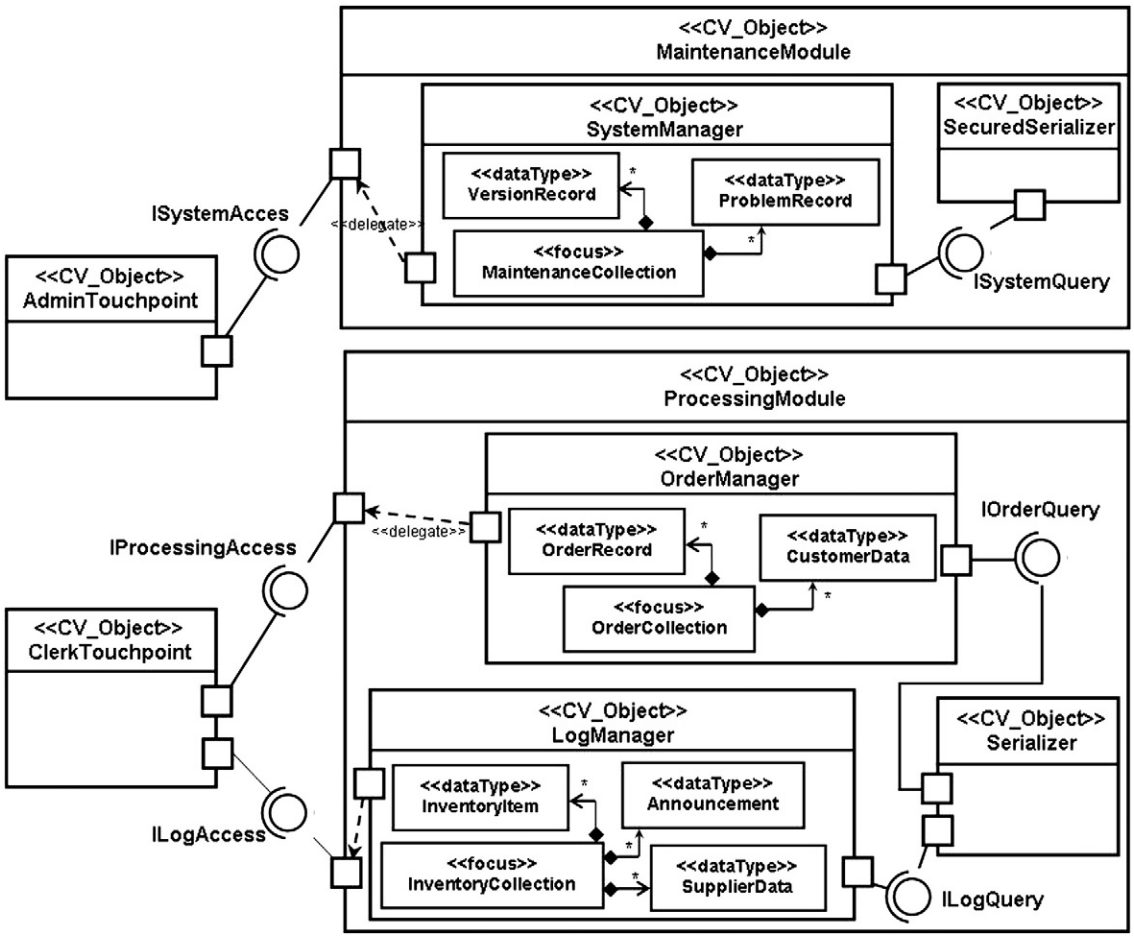

Fig. 11. Software architecture of the IT system of business working object `PurchasingDept`.

working object. The main IT system in this department digitizes customers' orders, customers and suppliers' data and provides a Web interface for various routines. A `Clerk` is performing daily check of customer's orders for abnormality or doubtful payments. From time to time, he puts announcements (e.g. a promotion campaign) on the web. An `Admin` is doing maintenance routines (i.e. troubleshooting, database backup, software update). Like the view shown in Fig. 9, we choose to hide the environment of this business working object.

Being viewed as a whole, the IT system exhibits information objects that represent digitized records of customers' data, orders, inventory and suppliers as well as information related to system administration. Localized actions in this view, namely tt RetrieveLog, `PutAnnouncement`, `Update`, `Troubleshooting` and `Backup`, represent services offered by the IT system (e.g. Web services). As a cognitive matter, the pictogram of the IT system in this view is different from those of other business working objects because it is an IT working object.

4.1.4. Computational viewpoint – extended organizational level

We recommend the use existing ODP standards and recommendations to complement SeamCAD in making ODP computational viewpoints of an enterprise model. Fig. 11 gives a computational specification of the `IT system` that provides IT capabilities of business working object `PurchaseDept`. This CV specification is represented using the UML[11] as recommended [20]. Computational objects `AdminTouchpoint` and `ClerkTouchpoint` stand for the user interface of the IT system. Computational objects `MaintenanceModule` and `ProcessingModule` deal with system maintenance and order/inventory management, respectively. `AdminTouchpoint` interacts with `MaintenanceModule` via an interface named `ISystemAccess` (likewise, we have interface `IProcessingAccess`). In this view, we also see component (computational) objects of `MaintenanceModule` and `ProcessingModule`. They manage data related to orders, customers, inventories and system troubleshooting. Some other computational objects (de-)serialize the data (database operations may not be visible in this view). Note that this representation could be considered a view showing an extended organizational level – the level of software architecture.

4.2. Bindings

In RM-ODP Part 3, viewpoint specifications address different modeling aspects but they are not isolated representations of an ODP model. RM-ODP defines correspondences between the viewpoint specifications [20,46]. As we use the modeling concepts coming from the RM-ODP enterprise language and the information language in defining SeamCAD building blocks, we essentially need to establish the correspondences between enterprise and information specifications. We have a notion of bindings[12] – intrinsic relationships between building blocks defined in the SeamCAD language. A business working object seen as a whole may have localized actions that represent the "responsibility" of this object in participating in a certain business collaboration. There could be information objects that express the information flow going through this business entity (that is viewed as a whole) in this

[11] Unified Modeling Language `http://www.uml.org/`.

[12] Note that this use of the term "binding" is specific to SeamCAD and should not be confused with the ODP concept of binding, which captures the process of establishing a communication context and the state created by that process.

collaboration. To make the context explicit, these business actions and information objects are bound to the collaboration in question. We call these links *goal-binding*s.

For instance, at the first organizational level and the first functional level of the Bookstore enterprise model (see Fig. 2), there are goal-bindings that link `Sell`, `Buy`, `SellTxn` and `small BuyTxn` to business collaboration `sale`. At the first organizational level and the second functional level (see Fig. 7), we have additional goal-bindings that link localized actions `GetOrder` and`CommitOrder` to business collaboration `placeOrder`. These goal-bindings are not diagrammatically represented in any views of the Bookstore enterprise model.

Another form of binding is *means-binding*. A means-binding links localized actions to a business collaboration that belongs to the same business working object but represented in two different views. The semantics of this binding is that business collaborations (visible when a business working object is seen as a composite) usually contribute to the realization of localized actions (visible when the same business working object is seen as a whole) of the same working object. In other words, for a given business working object, its as-a-composite view shows the "realization" for the "specification" represented by its as-a-whole view.

In the Bookstore enterprise model, we have a means-binding that links business collaboration `ship` to localized action `Delivery`. This means-binding is visualized by a little note attached to collaboration `ship` (see Fig. 8). Other examples include a means-binding linking business collaborations `payment_processing` to localized action `ReceivePayment`, and another that links business collaboration `invoice` tolocalized action `GetOrder`. These means-bindings are established for business working object `BookValueNetwork`. Like goal-bindings, they are not diagrammatically represented in SeamCAD.

Being intrinsic to the SeamCAD language, goal-bindings and means-bindings may not be diagrammatically represented but are necessary to link model elements of an enterprise model to make the model coherent. They are captured together with model elements and their diagrammatically-representable relationships in a coherent enterprise model. The goal-bindings partially determine how a view can be rendered to show part of the enterprise model the user wishes to see and work with (e.g. when the user is only interested in a specific collaboration and wishes to hide others).

The traceability between information objects represented in a business working object viewed as a whole to other working objects also translate into RM-ODP viewpoint correspondences. Typically, information objects in a business working object could be considered as images of other business/parameters working objects. The business working object has these images because it interacts with the others through business collaborations. Table 3 gives a summary of how we relate the traceability of SeamCAD to the correspondences between viewpoint specifications in RM-ODP, hence how we address the well-formedness challenge in Table 1.

In the in the Bookstore enterprise model, business working object `BookValueNetwork` seen as a whole has information objects `BookInstance` and `BookItem` that are actually images of parameter working object `1 Book`. This business working object also has information objects `OrderInfo` and `CustomerInfo` that are images of parameter working object `Order` and business working object `CustomerValueNetwork`, respectively.

5. Meta-modeling and tool support

5.1. Type and class

RM-ODP defines *type* and *class* as specification concepts to provide the modelers with mechanisms to describe model elements intentionally and extensionally, respectively. A type is defined by means of predicates. A class is a collection of elements that satisfy all predicates defined for a type [21]. We can intentionally describe types for business working objects, parameter working objects and business collaborations. For example, the following predicate describes a type of business working object `BookValueNetwork`: "Any value network of three companies that deal with the sale of books over the Internet in the Asia-Pacific region". A class of this type can be described by listing specific book-selling value networks that operate in Australia, Singapore, New Zealand, etc. "Any value network of three companies that deal with the sale of novels over the Internet in the Asia-Pacific region" is another predicate that describes a subtype.

Similarly, we can describe a type for business collaboration `sale` as follows: "Any sale happening between a book-selling value network and a customer value network in Europe". The following predicate describes a subtype "Any sale happening between a book-selling value network and a customer value network in Europe for which the total payment is no greater than 100 dollars".

5.2. Meta-model of SeamCAD

Fig. 12 depicts SeamCAD enterprise models at a meta level. The building blocks defined in Subsection 1 are represented as UML classes. Note that we introduce UML abstract classes such as `HierarchicalElement` and `Action` to this meta-model. They are needed for expressing the common patterns among the building blocks and thus make the meta-model succinct. They are not instantiated in SeamCAD enterprise models. We consider as-a-whole and as-a-composite the two viewing modes applied to model elements. For

Table 3
RM-ODP enterprise and information viewpoint specification correspondences in terms of traceability between SeamCAD building blocks.

SeamCAD traceability	RM-ODP viewpoint specification correspondences
Within a business working object seen as a whole, information objects are usually connected to business collaborations via goal-binding.	For each enterprise object in the enterprise specification, a list of those information objects (if any) that model information or information processing concerning the entity modeled by that enterprise object.
Within a business working object seen as a whole, the relationships between information objects are analogous to those of business working objects or parameter working objects they represent.	For each relationship between enterprise objects, the invariant schema (if any) which constrains objects in that relationship.
A business collaboration and its corresponding localized action within a working object seen as a whole are connected by a goal-binding.	For each action in the enterprise specification, the information objects (if any) subject to a dynamic schema constraining that action.
A business working object seen as a whole has information objects corresponding to other business parameters or business entities it collaborates with. We may call them images of business parameters/entities.	For each enterprise object and for each artifact role in an enterprise action, the corresponding configuration of information objects (if any) that model them in the information viewpoint.
For a given business working object, a localized action represented in its as-a-whole mode should be connected to some business collaboration represented in its as-a-composite view via a means-binding.	For each enterprise action and process in the enterprise viewpoint, the corresponding dynamic and invariant schema definitions in the information viewpoint that specify that behavior.

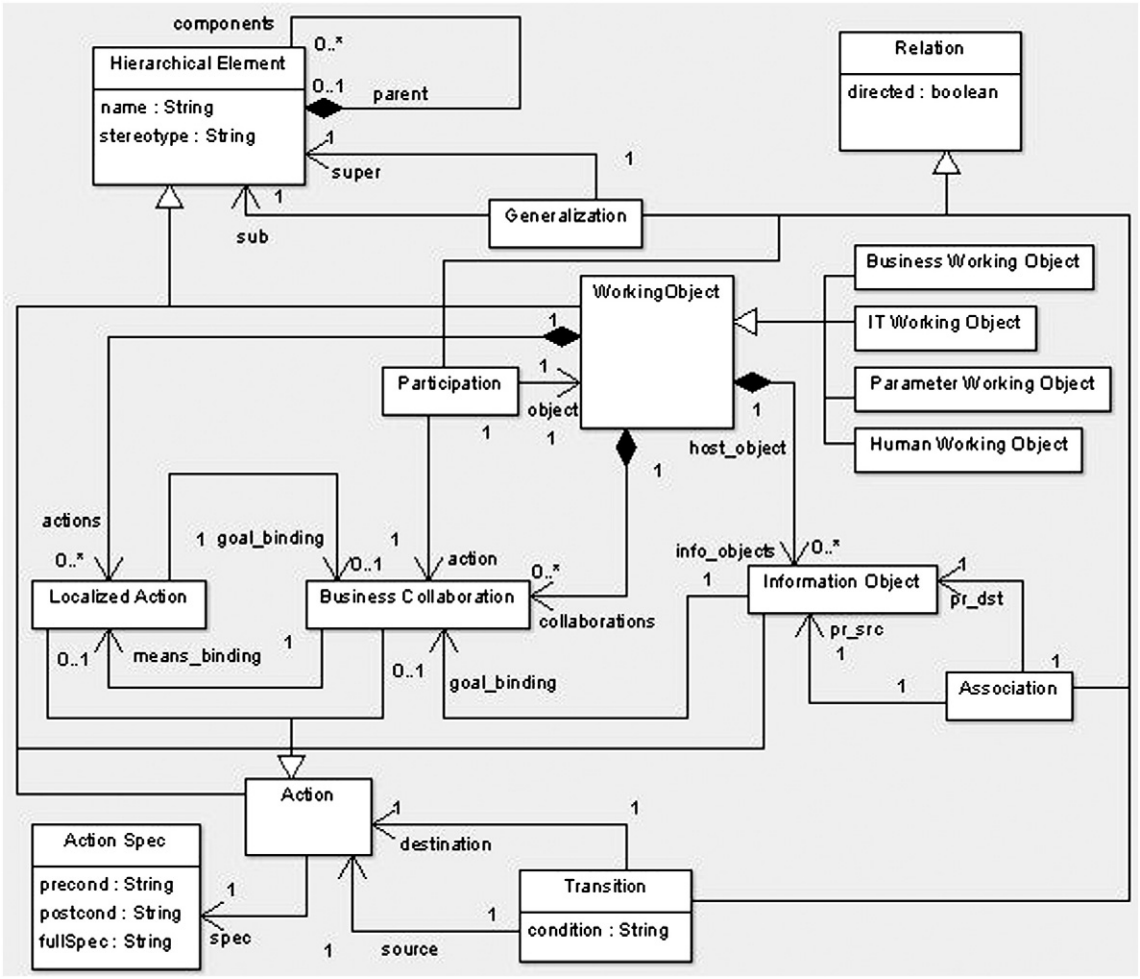

Fig. 12. A UML diagram that describes SeamCAD building blocks and relationships between them at a meta level. This UML diagram serves as a conceptual model for the implementation of the SeamCAD toolkit.

the sake of simplicity, we choose not to describe these viewing modes in this meta-model.

The goal-binding is expressed as an UML association end going from the UML class of information objects/localized actions to the UML class of business collaborations in this meta-model. Similarly, the means-binding is represented as an UML association end going from the concept of business collaborations to the concept of localized actions. Relationships that are diagrammatically represented in SeamCAD views are expressed as UML classes in this meta-model. They are association (between information objects), transition (between localized actions or between business collaboration) and participation (between a business/IT/parameter working object and a business collaboration).

The meta-model comes with well-formedness rules that ensure the enterprise models created in SeamCAD are well-formed. The following rules are indicative examples taken from a total of 19 rules that were defined in the SeamCAD meta-model[13] [29].

- There must be no cycles in the organizational level hierarchy.
- There must be no cycles in the functional level hierarchy.
- For a model element, all of its component model elements must be of the same kind.
- The orders in which localized actions are performed within a business/IT working objects comply with the orders that are specified for business collaborations the working object participates in.

5.2.1. Interpreting SEAM organizational level and functional level

A business/IT working object viewed as a whole has information objects and localized actions that exhibit its behavior and its information processing in collaborating with other working objects within the same organizational level. When viewing a business/IT working object as a composite, we see component working objects and business collaborations. By toggling the viewing mode on a business/IT working object from as-a-whole to as-a-composite, the user descends along the organizational hierarchy of her enterprise model.

In RM-ODP terms, the user goes to the next organizational level if she turns an enterprise object into a community (i.e. a configuration of enterprise objects). A community has a number of enterprise objects each of which can be turned in another community leading to multi-level representation [34].

Once the user breaks down a business collaboration, she descends to the subsequent functional level. To make the context of this collaboration explicit, information objects and localized actions that are bound to this business collaboration via goal-bindings should always be diagrammatically represented at the same level of granularity. Speaking of ODP terms, the user goes to the next functional level if she turns an enterprise step into an enterprise process.

[13] We formally described all these well-formedness rules together with the SeamCAD building blocks using Alloy.

5.3. Toolkit

We had implemented a toolkit[14] for modeling EA hierarchically [32] using the building blocks presented in Subsection 1. All the diagrammatic representations shown in previous subsections are generated by the SeamCAD tool. The implementation of this tool follows the conceptual model shown in Fig. 12 and meets [55] the well-formedness rules mentioned in Subsection 2.

The SeamCAD toolkit has a notation scheme, which defines pictograms for the building blocks and the rules that mandate the way these pictograms are put together in a diagrammatic representation. We have two principles in defining the SeamCAD notation: i) to visually express both the organizational hierarchy and the functional hierarchy in all views; ii) to leverage UML as the widely-used notation in modeling systems and software. To meet the first principle, we chose to nest pictograms to visually show the containment.

The SeamCAD business working object takes the notation of value chain that was made popular by Porter [43]. Parameter working objects take a slightly different notation: a 2-way block arrow. For human working objects, the UML actor notation ischosen. In SeamCAD, we do not model a human being as a whole or as a composite like we do to business/IT working objects. As such, the pictogram of people is not nested.

For IT working objects in SeamCAD, we choose the notation of UML subsystem to convey two meanings: as a classifier and as a package (in the UML meta-model, the subsystem inherits from both the classifier and the package). As a classifier, it represents something that has both structural and behavioral aspects. As a UML package, it can group other model elements, including subsystems, just like a container.

The SeamCAD information object takes the UML class notation without the two compartments for class attributes and class operations. Specifying association and generalization among information objects of the same working object results in a UML class diagram being rendered inside a pictogram of a working object seen as a whole.

The SeamCAD localized action in the modeling language takes the UML action notation. Specifying transitions among localized actions of the same working object results in a UML activity diagram being rendered inside a pictogram of a working object seen as a whole.

The SeamCAD business collaboration takes the UML collaboration notation. Specifying participation links among business collaborations and business/IT working objects result in a UML pattern structure being rendered inside a pictogram of an enclosing business/IT working object seen as a composite.

6. Applications and feedback

6.1. Applications

We provide several applications of SeamCAD one of which was done in conjunction with industry. Each application is briefly presented in a subsection. Further details of these applications can be found in the first author's doctoral thesis [29].

6.1.1. A case-study in a master's course on EA and SOA

At the I&C school of the EPFL, Switzerland, a problem-based subject was given to master's students to teach them how to start a manufacturing-and-sale company following a game case-study. Students were asked to make an enterprise model for their imaginary company. They were divided into groups of four to six. Each group represented a company that was to manufacture and sell diesel-powered engines for lightweight aircrafts [44]. An enterprise model of this company was built in a hierarchical manner. All model elements in this enterprise model were expressed using the SeamCAD building blocks and entered in the SeamCAD toolkit. This enterprise model served as a sample model for our students. It has a total of 3 organizational levels and the second level was represented at four different functional levels. The students could refer to it as one of the solutions after having created an enterprise model of their own company.

6.1.2. Enterprise model for an ERP-seeking company in a market of watch parts manufacturing

This application was in the context of a Swiss company that is active in the development of ERP (Enterprise Resource Planning) solutions and the research of a new ERP-based method for representing customers' needs in the market of watch manufacturing. A project was set up between this company and our group to develop an enterprise model for organizing, presenting information and doing "savoir-faire" ERP, which is systematically elaborated in the integration and deployment phase of an ERP system by customer companies. In addition, this model should allow the company to analyze the needs of current and future customers, with a goal to manage the technological evolution of the company.

This project led to a master's thesis we supervised [9]. Our master's student developed a SEAM model developed in the project and it was initially made on pieces of paper. This model was partially entered in the SeamCAD tool resulting in a total of 4 organizational levels of which the third and the fourth were represented at two different functional levels. We entered the most typical portion of their paper-based model in SeamCAD (we did not enter the entire paper-based model due to limited time – the company agreed that the other portions of their model could be entered in similar ways and this labor thus might not bring any additional benefits).

6.1.3. Designing EA with SEAM and SeamCAD

A project was launched to apply the SEAM method and the SeamCAD toolkit in designing an enterprise model for a project of a new building on the campus of the EPFL, Switzerland. This project led to a master's thesis we supervised [27]. The enterprise model built in this project was useful to specify how the building should be equipped and what IT systems should be installed in the building. It features a total of 3 organizational levels spanning the management of university, the school where the elements of IT equipment were to be installed and the operation of these elements by the school staff/students. In this enterprise model, the second and the third organizational levels were represented at two different functional levels. The model was built using in the very first version of SeamCAD has proved its usefulness when showing different views that would be of interest to different partners in the project.

6.1.4. Lessons learned from building enterprise models in SeamCAD

After having worked with SeamCAD in creating enterprise models, the practitioners, researchers and our master's students (we call them the modelers) who got involved in the projects presented in the previous subsection expressed their opinions and gave their reactions as follows.

- The modelers agreed that structuring enterprise models hierarchically was effective in documenting and presenting businesses. In our project with the Swiss company, they reported that the as-a-composite view was more useful than the as-a-whole one for analyzing and understanding their customers's needs. They also noted that the ability to present as-is and to-be models separately was desirable but it was unfortunately unsupported by the SeamCAD tool.
- The modelers found SeamCAD useful for capturing business functions and organizational structures in their enterprise models but

[14] The SeamCAD toolkit is implemented in standard Java. It can be run on any Java-enabled platforms. We have a plan to make it an open-source tool. At the time this article was written, we made the tool available at tt http://www.uow.edu.au/ lle/SeamCAD.

not for modeling non-functional properties and early phase requirements. In particular, a practitioner mentioned that SeamCAD could further be developed to tackle functional silos.
- Although they could get familiar with the SEAM modeling terms before actually using SeamCAD, the modelers were able to understand the notions of model hierarchy, the systemic principle and the dual views as-a-whole/as-a-composite efficiently by working with the SeamCAD tool in building their enterprise models.
- The modelers learned how to browse their enterprise model along the organizational hierarchy and the functional hierarchy in SeamCAD. They appreciated the capability of generating a view that renders part of their enterprise model they wanted to see and work with.
- A model created using the SeamCAD tool was considered as an electronic version of the enterprise model as opposed to the hard version that was built using pieces of paper and a pencil. The paper-based version of the enterprise model was created using pre-built template diagrams (e.g. templates for the market level, the value network level, the company level and the IT level). In the electronic version, the modelers were able to open more diagrams than they did in the hard version by drawing on pre-built templates on pieces of paper.
- The modelers were able to understand the notion of model well-formedness and the principle of having a single coherent model in SeamCAD as the tool has built-in rules that prevent them from adding model elements to their enterprise model in a way that may violate the well-formedness rules defined for the SeamCAD modeling language.

6.2. Feedback from practitioners, researchers and students

To see how SeamCAD addresses the four modeling challenges in Table 1 and if it brings some value to the industry and university courses, we invited practitioners/researchers in EA-related domains and students who took our problem-based master's courses in Service-Oriented and Enterprise Architecture [44,58] to validate SeamCAD.

6.2.1. Protocol for obtaining users' feedback

A questionnaire was prepared to obtain feedback for SeamCAD from practitioners/researchers in EA-related fields largely about the extent to which SeamCAD addresses the four modeling challenges that are presented in Section 2. It was then extended to cover additional issues of interest (e.g. top-down versus bottom-up modeling, notation intuitiveness). To help them get acquainted with SeamCAD before actually filling in this questionnaire, we provided them with a tutorial having step-by-step instructions for working with the SeamCAD tool. The tutorial and the questionnaire were tested within our research group to estimate time needed for each participant to complete the validation and to check for ambiguity.

Due to the difference of the participants' availability, we decided to proceed with them individually. Where no face-to-face conversation could be established, the participants may follow the tutorial and then fill in the questionnaire themselves while getting assistance either on the phone or via an online chat. For the students who were willing to take part in the validation, we were able to get them as a group of three to five at the same time for substantially longer periods of time. Instead of following a pre-defined tutorial, the students did more extensive practice by building an enterprise model in the SeamCAD tool.

6.2.2. Ratings

A total of 11 practitioners/researchers participated in this validation. They followed a 2-phase tutorial under assistance of the person responsible for the tool (with minimal influence on their assessment). In the first phase, each practitioner viewed an existing model of the online bookstore to get acquainted to basic functionalities of the SeamCAD tool and the SeamCAD building blocks. She/he then proceeded in the second phase to extend one more organizational level and two functional levels of the model of the bookstore. After having practiced with the tool, each practitioner was asked to answer a questionnaire to rate how SeamCAD meets the four modeling challenges (see Table 1) and to leave their suggestions. Each practitioner/researcher completed the tutorial and the questionnaire in about 30–45 min.

A total of 9 students took part in the validation of SeamCAD. They had just finished a master's course on EA and Service Oriented Architecture given by our lab. In this course, students were asked to make an enterprise model for an imaginary company– a company that manufactures and sells diesel-powered aircraft engines. They were divided into groups of less than six. Each group represented a company and built an EA model for their company on paper. Students who participated in the validation of SeamCAD were asked to rebuild the model they made on pieces of paper using the SeamCAD tool. They then answered a questionnaire to evaluate how SeamCAD meets the four modeling challenges and also to give their feedback as if SeamCAD was employed as a teaching tool in their course. It took approximately 7 h for these students to complete the validation.

In total, we had 20 participants giving their feedback. The two questionnaires used in this validation (one for practitioners, one for students) have 7 questions in common. Table 4 presents the distribution of their answers for these questions [29]. Their feedback suggests that the framework has some advantages as well as disadvantages. SeamCAD is relatively good for capturing the SEAM organizational hierarchy and the context view of an enterprise model in a hierarchical manner. Separating the as-a-whole and as-a-composite view with the possibility to toggle any model element between these two views is another good point. When it comes to tool implementation, managing the well-formedness of the enterprise model and the browsing capability are considered as noticeable advantages.

Shortcomings of SeamCAD include the way the tool represents the functional hierarchy of enterprise models, the lack of advanced features for managing large enterprise models and for customizing diagrams (such as hiding/showing the environment of a business working object), the non-intuitive way of expressing information flow as the exchange of input and output information objects and the unreliability of the tool due to software bugs.

Table 4
Distribution of answers given by practitioners and students who practiced SeamCAD.

Main questions in the questionnaire	Answer distribution			
	Excellent (yes)	Good	Bad	Very bad (no)
Do you think that the top-down approach of SeamCAD has some value in your practice?	20/20	–	–	0/20
How do you rate the way SeamCAD manages your organizational levels?	7/20	13/20	0/20	0/20
How do you rate the way SeamCAD manages your functional levels?	4/20	15/20	1/20	0/20
How do you rate the way SeamCAD handles multiple system representation?	10/20	8/20	2/20	0/20
How do you rate the feature of SeamCAD through which diagrams can be opened as partial views of a common model?	13/20	7/20	0/20	0/20
How do you rate the way diagrams can be customized by hiding/showing specific elements in SeamCAD?	9/20	7/20	4/20	0/20
How do you rate the intuitiveness of the notation scheme used in SeamCAD?	4/20	14/20	2/20	0/20

7. Related work

In this section, we discuss existing work and/or standards on EA modeling as well as system/software development with regard to the challenges we have identified for modeling EA hierarchically.

7.1. ODP-based work on enterprise modeling

We provide a brief analysis of ODP-based work on EA modeling in this subsection. Steen et al. proposed a framework for classifying viewpoints for EA to complement existing viewpoint models of RM-ODP [50]. The emphasis was put on multi-viewpoint representation of EA rather than modeling EA hierarchically. Lupu et al. attempted to realize the concepts of the RM-ODP enterprise language using an intuitive, declarative language [39]. This work addresses the EA hierarchy by providing means for capturing roles, relationships and their configurations in nested communities. Milosevic et al. attempted to express and monitor contracts in an enterprise context [40]. Dijkman et al. formalized consistency rules between the enterprise viewpoint specification and the computational viewpoint specification of RM-ODP in order to verify the consistency between the two specifications [11]. Boiten et al. check the consistency between all ODP viewpoints with an emphasis put on the computational and engineering viewpoints [6]. SeamCAD does not rigorously check the consistency between modeling aspects of an enterprise model. Instead, some EV-IV viewpoint correspondences are translated into intrinsic relationships of the SeamCAD language (we call them goal-binding and means-binding). Others are captured via guidelines for building multi-level enterprise models.

At earlier development stages of RM-ODP, the EV was given less attention than other ODP viewpoints were. The relative immaturity of EV had attracted researchers to extend it in one way or another. Most notably, the vocabulary of ODP enterprise language was added with objectives, processes and concepts addressing legal issues [1]. Linington et al. explored the semantics of the concept of enterprise policy within and between communities [35] and proposed a contract language for self-contained enterprise and cross-enterprise settings [36]. Some concepts proposed in these extensions later became part of an international standard and recommendation for EV [19].

An international standard that guides the use of UML in making viewpoint specifications (including the enterprise viewpoint specification) for ODP systems was accepted by ISO/IEC [20]. This ISO/IEC standard demonstrates its guidance using a case-study about a library management system with sufficient levels of detail. The authors of this work also analyzed requirements for ODP-based EA modeling tools [45]. We proposed SeamCAD as an EA modeling tool using RM-ODP (especially its EV and IV languages) in a hierarchy-oriented fashion. Alternatively, enterprise architecture could be regarded as networked business using the federation concept defined in the EV language of RM-ODP [26]. Other pieces of work explicitly address ODP concepts related to enterprise policy, constraints or goals [49,38,34,2], which SeamCAD does not.

7.2. Non-ODP enterprise modeling

A number of methods have been developed for modeling EA, most notably (listed in alphabetical order): Archimate, CIMOSA, DEMO, TOGAF and Zachman. There also exists work that model business services. We analyze them with regards to the challenges of Table 1.

ArchiMate proposes an integrated modeling framework for Enterprise Architecture including organizational structure, business processes, information systems and infrastructure [28]. This framework proposes three layers – namely, business layer, application layer and technology layer. Each of these layers can further be divided into sub layers, making the organizational hierarchy virtually visible. The process can be broken down into smaller processes, implicitly showing the functional hierarchy.

The Computer Integrated Manufacturing Open System Architecture (also known as the ISO EN/IS 19440 standard of CIMOSA) focuses on the modeling of processes in the context of computer integrated manufacturing projects [15]. CIMOSA defines four modeling views: function view, information view, resource view and the organization view. The resource view and the organization view address the structure of resources (e.g. humans, machines, information systems) but do not show an explicit organizational hierarchy. Multiples systems can be represented in CIMOSA. The model well-formedness is not addressed, however.

Design & Engineering Methodology for Organizations (DEMO) is a method for (re)designing organizations [10]. DEMO defines three types of models of an organization: the black-box model, the white-box model, and the flow model. The black-box model deals mainly with the external behavior of a system and supports the functional refinement. In the flow model, a system is conceived as a network of nodes transforming the input flows into output flows. The white-box model defines the constructional refinement of the system. Multiple systems can be represented. The organizational hierarchy and the model well-formedness are not discussed however.

TOGAF[15] and Zachman [60] propose ad-hoc modeling frameworks in which multiple systems can be represented. But they do not have any explicit organizational or functional hierarchy. The well-formedness of model is down to the burden of the modeler in making her model.

Among the work that describes services from non-IT perspectives, most notably we have the Unified Service Description Language (USDL),[16] which is a description language for an Internet of services. In this language, services are described in three perspectives, namely business, operational and technical. Other pieces of work in business service description proposes languages that capture functional and non-functional properties of services from business-oriented angles [42,33]. These languages may address the functional hierarchy via service composition but fall short on the organizational hierarchy (they merely mention service providers and service consumers without organizational structure).

7.3. Software and system modeling

We have witnessed the boom of methods and languages for system engineering and software engineering. We describe here some of the approaches that may be used for modeling EA hierarchically. We selected the methods that have features closest to the aforementioned criteria. They are (listed in alphabetical order) Addora, Catalysis, KobrA and OPM.

Adora is an object-oriented modeling method that features hierarchical decomposition and the integration of all aspects in one coherent model [18]. Objects in Adora are composite by default. The organizational hierarchy can be reasoned about in terms of the composition of objects. This hierarchy is visually depicted in the tool of Adora. The functional hierarchy can be thought of as a tree-like hierarchy of scenarios (in Adora, a scenario is similar to a UML use-case). The hierarchy of scenarios is not visible in the tool, however. Adora is a system-centric modeling language.

Catalysis is a component-based development process that analyzes and designs in three levels: business, IT system and software components [14]. It uses its own notation that is inspired by UML. Catalysis put a lot of effort in making behavior refinement. It made popular the notion of two kinds of action, namely joint action and localized action. In principle, the organizational hierarchy of Catalysis is visible in the containment hierarchy, which is typically up to three

[15] The Open Group Architecture Framework http://www.togaf.org/.
[16] SAP Research's Internet of Services http://internet-of-services.com/.

levels: context level, software level and component level. As Catalysis defines a development process for software-intensive systems, it is a typical system-centric method. In Catalysis, the well-formedness of a model is maintained by keeping traceability between different refinements.

KobrA proposes a recursive model that describes IT systems/components. KobrA takes the notation from UML. In KobrA, each component is described dually in terms of specification and realization [4]. The recursive approach of KobrA implicitly suggests that this method can deal with as many organizational levels as the modeler wishes to. But in practice, KobrA aims at representing only the context level and some component levels. As KobrA tackles software development process, it is a typical system-centric method although the concept of component can be used for representing any business entity or IT system in an enterprise model. In KobrA, it is up to the modeler to maintain the well-formedness of her model by practicing inter-diagram and intra-diagram rules.

Object-Process Methodology (OPM) addresses the modeling of systems in general [12]. It has its own notation and provides a modeling tool called OpCat [13]. The building blocks of OPM are object, process, state and relations. The organizational hierarchy can be described via object aggregation and the functional hierarchy can be reasoned in terms of process aggregation. Multiple systems can be designed in the same OPM model. Although OpCat supports hierarchical modeling by allowing the modeler to zoom in on a specific object or process, it is up to the modeler to maintain the well-formedness of her model by ensuring that all diagrams are consistent.

8. Conclusion

This article presents our work on modeling EA hierarchically. An enterprise model that represents the enterprise and its environment might include various aspects such as the internal structure of the enterprise and the services provided by the enterprise, the business processes and data flow between business entities, the IT components and their interaction. Inspired by a tendency for people often to simplify their perception of the reality by analyzing it hierarchically, we were motivated to develop a hierarchy-oriented framework for modeling EA.

8.1. Contributions

We have developed SeamCAD based on this motivation. We leveraged the ODP modeling concepts and the modeling rationale of SEAM to define a couple of building blocks needed for modeling an EA hierarchically. We implemented a modeling toolkit for SeamCAD. The tool manages a coherent enterprise model and can generate a view to permit reasoning about a certain portion of the model it manages. In this way, SeamCAD actually consolidates the SEAM method [52,59,57] by i) leveraging viewpoint-specific modeling concepts of RM-ODP and the EA modeling principles of SEAM in developing a hierarchy-oriented modeling language; ii) defining well-formedness rules for this language [29,30]; and iii) building a toolkit [32]. This is our first contribution.

Our second contribution is centered around how to make RM-ODP, particularly the ODP languages of EV and IV, applicable in the context of multi-level EA modeling while adopting other ODP viewpoint languages (e.g. the CV language) to capture technical aspects (e.g. software/system architecture). To reach international consensus and to become a standard, the RM-ODP is sometimes excessively general and avoids providing design heuristics [17]. We found a way to apply the modeling concepts defined in these two languages uniformly across all hierarchical levels of EA. We have also explored the ODP correspondences between these two viewpoints in the realm of EA modeling. We proposed the use of Alloy [24] as an alternative to OCL [41] in capturing ODP dynamic schema.

As for the applicability of our framework, we had a total of 3 applications of SeamCAD and have tested it with a total of 20 practitioners/students.

8.2. Discussions

Enterprise Architecture is essentially a multidisciplinary field that includes many aspects. It was not the goal of our work to capture all EA aspects. An enterprise model created in SeamCAD is mainly about the services, information processing and the organization of the enterprise being modeled. SeamCAD does not address non-functional aspects such as finance, governance, quality of services, security, database, network, system interoperability and low-level software design. The modelers may enhance their hierarchical EA models represented in SeamCAD by adding computational, engineering or technology specifications using UML as recommended in ODP standards should they wish to address more technical aspects.

8.3. Future work

We have a few directions for future investigations. To complete the picture of EA modeling, it would be useful to represent enterprise policies and roles as defined in the enterprise language of RM-ODP. Currently, we make an implicit assumption in SeamCAD regarding role modeling: each business working object fulfills a role that emerges because of its existence (for instance, `BookValueNetwork` fulfills role *Seller* and likewise, `CustomerValueNetwork` fulfills role textitBuyer). In enterprise settings, roles can be composed or separated [5]. We need to relate component working objects, localized actions of a business working object to the roles the working object in question plays [16]. Also, the topic of strategic alignment has increasingly become relevant to the field of EA. Naturally, one might interpret that business processes would contribute to the realization of business objectives, or the other way round – objectives may influence the way processes are engineered. We will look into the correlation between business objectives and localized actions/collaborations represented in our EA models.

Another direction is to integrate action specification into enterprise models to make them simulatable. The idea is to annotate Alloy code to business collaborations or localized actions. We aim to use Kodkod – the underlying reasoning engine of Alloy [51], to interpret these annotations (i.e. to check whether information objects are correctly referenced or not) and ultimately simulate the enterprise model for a given business collaboration. Alternatively, we can define a rule for naming a transition between two localized actions of the same working object, e.g. we can put the names of the information objects that are produced or consumed by these actions on this transition to diagrammatically express their semantics. As such, the enterprise model created in SeamCAD can be exported to either a declarative language such as Alloy [24] or a business execution language (e.g. Business Process Execution Language (BPEL), provided that this language is extended to include human activities and services of non-IT entity [25]).

Regarding the SeamCAD toolkit, we aim to improve it taking into account the suggestions given by practitioners and students who tested it. Moreover, we have a plan to make it an ODP-based tool in the sense that it will allow its users to view their enterprise model not only in a hierarchy-oriented manner but also in a viewpoint-oriented fashion. In essence, the tool will generate different views corresponding to an EV specification, an IV specification and additional ODP viewpoint specifications (e.g. a CV specification) for the currently-edited enterprise model. It may enforce ODP viewpoint correspondences between these specifications, especially between the EV specification and others. Alternatively, the tool may assist its users in maintaining the ODP viewpoint correspondences of their enterprise model based on a list of built-in modeling heuristics.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful and thorough comments on the earlier versions of this article.

References

[1] J.-O. Aagedal, Z. Milosevic, ODP enterprise language: UML perspective, Proceedings of the 3rd International Conference on Enterprise Distributed Object Computing, 1999, pp. 60–71, Mannheim, Germany.
[2] J.-P. Almeida, E.-C. Cardoso, G. Guizzardi, On the Goal Domain in the RM-ODP Enterprise Language: An Initial Appraisal based on a Foundational Ontology, Proceedings of 6th International Workshop on ODP for Enterprise Computing, in conjunction with 14th EDOC, 2010, pp. 19–27, Vitória, Brazil.
[3] Deborah J. Armstrong, The quarks of object-oriented development, ACM Communications 490 ((2):0) (February 2006) 123–128.
[4] C. Atkinson, J. Bayer, C. Bunse, E. Kamsties, O. Laitenberger, R. Laqua, D. Muthig, B. Paech, J. Wüst, J. Zettel, Component-based Product Line Engineering with UML, Addison-Wesley Professional, 2002 ISBN 978–0201737912.
[5] P. Balabko, A. Wegmann, Context based reasoning in business process models, Proceedings of 4th IEEE International Conference on Information Reuse and Integration, IEEE Computer Society, Las Vegas, USA, 2003, pp. 120–128.
[6] E. Boiten, H. Bowman, J. Derrick, P. Linington, M. Steen, Viewpoint consistency in ODP, The International Journal of Computer and Telecommunications Networking 340 ((3):0) (September 2000) 503–537.
[7] P. Checkland, J. Scholes, Soft Systems Methodology in Action, Wiley, Chichester UK, 1999 ISBN 978–0471986058.
[8] O.-J. Dahl, E.-W. Dijkstra, C.-A. Hoare, Structured Programming, Academic Press, London, UK, 1972 ISBN 978–0122005503.
[9] D. Dan. *ERP Handbook, Outil d'organisation pour l'intégration et le développement de la solution ERP DOPG Prod.com répondant aux besoins actuels et futurs des clients de DOP Gestion SA*. Master's thesis, School of Computer and Communication Sciences, EPFL, 2008.
[10] J. Dietz, Enterprise Ontology: Theory and Methodology, Springer Verlag, New York, USA, 2006 ISBN 978-3642067150.
[11] R. Dijkman, D. Quartel, L.-F. Pires, M.-J. van Sinderen, A Rigorous Approach to Relate Enterprise and Computational Viewpoints, Proceedings of the 8th International Conference on Enterprise Distributed Object Computing, IEEE Computer Society, Monterey, USA, 2004, pp. 187–200.
[12] D. Dori, Object-Process Methodology: A Holistic Systems Paradigm, Springer Verlag, New York, USA, 2002 ISBN 978-3540654711.
[13] D. Dori, I. Reinhartz-Beger, A. Sturm, OPCAT – A Bimodal CASE Tool for Object-Process Based System Development, Proceedings of 5th International Conference on Enterprise Information Systems, 2003, Angers, France.
[14] Desmond F. D'Souza, Alan C. Wills, Object, Components and Frameworks with UML, The Catalysis Approach, Addison-Wesley Professional, 1999 ISBN 978–0201310122.
[15] ESPRIT Consortium AMICE (Ed.), CIMOSA: Open System Architecture for CIM, 2nd edition, Springer, 1993, ISBN 978–0387562568.
[16] G. Genilloud, A. Wegmann, A Foundation for the Concept of Role in Object Modelling, Proceedings of 4th International EDOC Conference, IEEE Computer Society, Makuhari, Japan, 2000, pp. 76–85.
[17] Guy Genilloud, Common Domain Objects in the RM-ODP Viewpoints, Computer Standards & Interfaces 190 ((7):0) (November 1998) 361–374.
[18] M. Glinz, S. Berner, S. Joos, Object-Oriented Modeling with Adora, Information Systems 270 ((6):0) (September 2002) 425–444.
[19] ISO/IEC, Information Technology – Open Distributed Processing – Reference Model – Enterprise Language | ISO/IEC 15414 | ITU-T Recommendation X.911. SC 7 and ITU, 2006.
[20] ISO/IEC, Information Technology - Open Distributed Processing - Use of UML for ODP system specifications | ITU-T Recommendation X.906 | ISO/IEC 19793, International standard, SC 7/WG19 and ITU-T, 2009.
[21] ISO/IEC, ITU-T X.902 | ISO/IEC 10746–2 Information Technology – Open Distributed Processing – Reference Model – Foundations, International standard, SC 7 and ITU, 2010.
[22] ISO/IEC, ITU-T X.903 | ISO/IEC 10746–3 Information Technology – Open Distributed Processing – Reference Model – Architecture, International standard, SC 7 and ITU, 2010.
[23] ISO/IEC 10746–1, 2, 3, 4, ITU-T Recommendation, X.901, X.902, X.903, X.904, Reference Model of Open Distributed Processing, International standard, OMG, 1995–1996.
[24] D. Jackson, Alloy: a Lightweight Object Modelling Notation, ACM Transactions on Software Engineering and Methodology 110 ((2):0) (2002) 256–290.
[25] M. Kloppmann, D. Koenig, F. Leymann, G. Pfau, A. Rickayzen, C. Riegen, P. Schmidt, I. Trickovic, WS-BPEL Extension for People – BPEL4People, White paper, IBM and SAP, June 2005.
[26] L. Kutvonen, Reflective Federation of Enterprises in Open Service Ecosystem, Proceedings of 6th International Workshop on ODP for Enterprise Computing, in conjunction with 14th EDOC, 2010, pp. 1–5, Vitória, Brazil.
[27] K. Langenberg. *Designing Enterprise Architectures with the SEAM Method - In-Depth Study, Application and Critical Analysis*. Master's thesis, School of Computer and Communication Sciences, EPFL, 2003.
[28] M. Lankhorst, Enterprise Architecture at Work – Modelling, Communication and Analysis, 2nd edition Springer Berlin Heidelberg, 2009 ISBN 978–3642013096.
[29] L-S. Lê. *SeamCAD: a Hierarchy-Oriented Modeling Language and a Computer-Aided Tool for Enterprise Architecture*. PhD thesis, School of Computer and Communication Sciences, EPFL, n° 4225, 2008.
[30] L.-S. Lê, A. Wegmann, Definition of an Object-Oriented Modeling Language for Enterprise Architecture, Proceedings of 38th Hawaii International Conference on System Sciences, Track 8, IEEE Computer Society, Hawaii, USA, 2005, p. 222a.
[31] L.-S. Lê, A. Wegmann, An RM-ODP Based Ontology and a CAD Tool for Modeling Hierarchical Systems in Enterprise Architecture, 2nd International Workshop on ODP for Enterprise Computing, in conjunction with 9th EDOC, 2005, pp. 7–15, Enschede, The Netherlands.
[32] L.-S. Lê, A. Wegmann, SeamCAD: Object-Oriented Modeling Tool for Hierarchical Systems in Enterprise Architecture, Proceedings of 39th Hawaii International Conference on System Sciences, Track 8, IEEE Computer Society, Hawaii, USA, 2006, p. 179c.
[33] L.-S. Lê, A. Ghose, E. Morrison, Definition of a Description Language for Business Service Decomposition, Proceedings of First International Conference on Exploring Services Sciences (IESS 1.0), 2010, Geneva, Switzerland.
[34] P. Linington, The Stereochemistry of Enterprise Objects, Proceedings of 6th International Workshop on ODP for Enterprise Computing, in conjunction with 14th EDOC, 2010, pp. 11–18, Vitória, Brazil.
[35] P. Linington, Z. Milosevic, K. Raymond, Policies in Communities: Extending the ODP Enterprise Viewpoint, Proceedings of the 2nd International Workshop on Enterprise Distributed Object Computing, 1998, pp. 14–24, La Jolla, USA.
[36] P. Linington, Z. Milosevic, J. Cole, S. Gibson, S. Kulkarni, S. Neal, A Unified Behavioural Model and a Contract Language for Extended Enterprise, Data & Knowledge Engineering 510 (1):0) (October 2004) 5–29.
[37] Ludwig Von Bertalanffy, General System Theory: Foundations, Development, Applications, George Braziller, New York, March 1969 ISBN 978–0807604533.
[38] E. Lupu, M. Sloman, A Policy Based Role Object Model, Proceedings of the 1st International Conference on Enterprise Distributed Object Computing, IEEE Computer Society, Gold Coast, Australia, 1997, pp. 36–47.
[39] E. Lupu, M. Sloman, N. Dulay, and N. Damianou. Ponder: Realising Enterprise Viewpoint Concepts. In *Proceedings of 4th International Conference on Enterprise Distributed Object Computing*, pages 66–75, Makuhari, Japan, 2000. IEEE Computer Society. *Living Systems*. New York: McGraw-Hill, 1978. ISBN 978–0070420151.
[40] Z. Milosevic, R.-G. Dromey, On Expressing and Monitoring Behaviour in Contracts, Proceedings of the 6th International Conference on Enterprise Distributed Object Computing, 2002, pp. 3–14, Lausanne, Switzerland.
[41] OCL, Object Constraint Language Specification Version 2.0, Object Management Group (OMG), May 2006 OMG Available Specification.
[42] Justin J. O'Sullivan. *Towards a Precise Understanding of Service Properties*. PhD thesis, Faculty of Information Technology, Queensland University of Technology, n° 16503, 2007.
[43] Michael E. Porter, Competitive Advantage: Creating and Sustaining Superior Performance, 1st edition Free Press, 1998 ISBN 978–0684841465.
[44] G. Regev, Donald C. Gause, A. Wegmann, Experiential Learning Approach for Requirements Engineering Education, Requirements Engineering 140 (4):0) (2009) 269–287.
[45] J.-R. Romero, A. Vallecillo, Requirements for ODP Enterprise Architecture Tools, Proceedings of 4th International Workshop on ODP for Enterprise Computing, in conjunction with 11th EDOC, 2007, pp. 224–230, Annapolis, USA.
[46] J.-R. Romero, A. Vallecillo, Well-formed Rules for Viewpoint Correspondences Specification, Proceedings of the 12th Enterprise Distributed Object Computing Conference, IEEE Computer Society, Munich, Germany, 2008, pp. 441–443.
[47] J. Schekkerman, How to Survive in the Jungle of Enterprise Architecture Framework: Creating or Choosing an Enterprise Architecture Framework, Trafford Publishing, 2004 ISBN 978–1412016070.
[48] Raymond M. Smullyan, First-Order Logic, Dover Publications, 1995 ISBN 978–0486683706.
[49] M.-W. Steen, J. Derrick, Formalising ODP enterprise policies, Proceedings of the 3rd International Conference on Enterprise Distributed Object Computing, IEEE Computer Society, Mannheim, Germany, 1999, pp. 84–93.
[50] M.-W. Steen, D.-H. Akehurst, H.-W. ter Doest, M.-M. Lankhorst, Supporting Viewpoint-Oriented Enterprise Architecture, Proceedings of the 8th International Conference on Enterprise Distributed Object Computing, IEEE Computer Society, Monterey, USA, 2004, pp. 201–211.
[51] E. Torlak, D. Jackson, Kodkod: A Relational Model Finder, Proceedings of the 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Wiley, Braga, Portugal, March 2007, pp. 632–647.
[52] A. Wegmann, On the Systemic Enterprise Architecture Methodology (SEAM), Proceedings of 5th International Conference on Enterprise Information Systems, 2003, pp. 483–490, Angers, France.
[53] A. Wegmann, A. Naumenko, Conceptual Modeling of Complex Systems Using an RM-ODP Based Ontology, Proceedings of 5th International EDOC Conference, IEEE Computer Society, Seattle, USA, 2001, pp. 200–211.
[54] A. Wegmann, P. Balabko, L.-S. Lê, G. Regev, I. Rychkova, A Method and Tool for Business-IT Alignment in Enterprise Architecture, Proceedings of the 17th Conference on Advanced Information Systems Engineering Forum, FEUP Edicões, Porto, Portugal, 2005, pp. 113–118.
[55] A. Wegmann, L.-S. Lê, L. Hussami, D. Beyer, A Tool for Verified Design using Alloy for Specification and CrocoPat for Verification, Proceedings of First Alloy Workshop, ACM SIGSOFT Sofware Engineering Notes, volume 31, 2006, pp. 58–62, Portland, USA.
[56] A. Wegmann, L.-S. Lê, J.-D. de la Cruz, I. Rychkova, G. Regev, An Example of a Hierarchical System Model using SEAM and its Formalization in Alloy, Proceedings of 4th International Workshop on ODP for Enterprise Computing, in conjunction with 11th EDOC, 2007, pp. 21–29, Annapolis, USA.

[57] A. Wegmann, L.-S. Lê, G. Regev, B. Wood, Enterprise Modeling Using the Foundation Concepts of the RM-ODP ISO/ITU Standard, Information Systems and e-Business Management 50 (4):0) (2007) 397–413.
[58] A. Wegmann, G. Regev, J.-D. de la Cruz, L.-S. Lê, I. Rychkova, Teaching Enterprise Architecture in Practice, Journal Enterprise Architecture ((3):0) (2007) 15–24.
[59] A. Wegmann, G. Regev, I. Rychkova, L.-S. Lê, J.-D. de la Cruz, P. Julia, Business-IT Alignment with SEAM for Enterprise Architecture, Proceedings of 11th IEEE International Conference on Information Reuse and Integration, IEEE Computer Society, Annapolis, USA, 2007, pp. 111–121.
[60] John A. Zachman, A Framework for Information Systems Architecture, IBM System Journal 260 ((3):0) (September 1987) 276–292.

Lam-Son Lê earned his Engineer's degree from the Faculty of Computer Science & Engineering, University of Technology of Ho-Chi-Minh City, Vietnam in 1998. He obtained his Ph.D. degree from the School of Computer and Communication Sciences, EPFL, Switzerland in 2008. Currently, he is a postdoctoral researcher at the School of Computer Science and Software Engineering, University of Wollongong, Australia. His research interests include enterprise architecture, service-oriented computing, business process management, requirements engineering, RM-ODP and formal methods in software engineering.

Alain Wegmann is a full professor at the School of Computer and Communication Sciences, EPFL, Switzerland. He worked 14 years for Logitech (Switzerland, Taiwan, US) in positions ranging from software developer, IS manager, manufacturing engineering to VP of engineering and OEM marketing before joining the EPFL in 1997. His research interests include enterprise architecture, requirements engineering and service engineering.