Towards Devising an Architectural Framework for Enterprise Operating Systems

Sérgio Guerreiro[1], Steven van Kervel[2] and Eduard Babkin[3]
[1]*Universidade Lusófona de Humanidades e Tecnologias, Escola de Comunicação, Artes, Arquitetura e Tecnologias da Informação, Campo Grande, 376, 1749-024, Lisbon, Portugal*
[2]*Formetis BV, Hemelrijk 12 C, 5281PS Boxtel, Netherlands*
[3]*National Research University Higher School of Economics, B. Pechorskaya 25/12, Nizhny Novgorod, 603155, Russia*

Keywords: Control, DEMO, Enterprise Engineering, Framework, Generation, Operating System.

Abstract: One often observes that despite the efforts involved on information systems development, many times the achieved solutions do not comply well with the needs of the organization and lack in change agility to face new emerging requirements. This paper conceptualizes about an architectural framework specifically designed for explaining and representing the working environment of enterprise operating systems (EOS) and non-EOS information systems (IS) that depend on it. With these concerns in mind, the goal of EOS is to capture and control all phenomena that occur in an organization and then to provide all the required data for all IS for that organization. Using a computer engineering metaphor, this paper defines the theoretical foundations, and the methodology to design and implement an operating system for organizations. To achieve this, EOS is a model-driven dynamically. The models are based on domain ontology with C4-ness qualities and expressed in ontological language. It will be shown that the theory of enterprise ontology and the DEMO methodology provide a high degree of ontological appropriateness for this domain. This paper, outline a framework with its foundations, concepts, principles and stratification of IS, which is a radical new approach, useful to discussion this solution among the practitioners. A methodology to apply this architectural framework is discussed using a professional production case management.

1 INTRODUCTION

Much similar to explosion of design space after introduction of reinforced concrete into the practice of civil engineering, deep penetration of Information and Communication Technologies (ICT) into our everyday activities leads to enormous inflation of design possibilities of software architects and engineers during design and implementation of complex software and hardware artifacts which are usually called information systems. This unwanted and inflated design space leads to unmanageable complexity in design and results in bad engineering. Now there is a great demand to produce mental techniques and tools, which will make design and development of information systems intellectually manageable.

Various proposals were offered to aim such goal, which are called architectural frameworks. The main purpose of architectural frameworks is to offer such architectural principles, which curb this unwanted design freedom as much as possible, but in such a way, that good engineering is supported as much as possible.

To the moment, there are plenty of architectural frameworks in use. Some of them like TOGAF, MODAF (Open Group, 2011) (Ministry Defense, 2010) encompass not only ICT aspects, and they are actively applied for engineering of enterprises as a whole. Despite a high level of conceptualization achieved in these frameworks researchers (Dietz, 2006) argued serious critical statements (Teka et al., 2012). For example, all of the mentioned frameworks lack consistent definition of the key concept of architecture, mixing in the same concept high-level design principles and reusable design constraints.

In our research, we reduce ambiguity in definitions of architecture following a strict definition of architecture (Hoogervorst, 2004), which defines it as *"a consistent set of design principles and standards that guide design".*

Guerreiro S., van Kervel S. and Babkin E..
Towards Devising an Architectural Framework for Enterprise Operating Systems.
DOI: 10.5220/0004597805780585
In *Proceedings of the 8th International Joint Conference on Software Technologies* (ICSOFT-PT-2013), pages 578-585
ISBN: 978-989-8565-68-6

However, even in that case, we claim, that there are other principal drawbacks in current theory and practice of architecture for information systems. Known approaches: (1) do not take into account human-centric foundations of enterprises for which the information systems are produced; (2) at the same time they insufficiently use generic and reusable formal methods and practices of engineering sciences (for example, clear distinction between function-construction perspectives, support for design- and life-cycles); (3) they do not take into account immanent stratification of abstraction levels of information systems, if we consider them in the global scope of the enterprise. The later drawback has great influence if we take maturity and strategy of particular enterprise during engineering of information systems seriously.

In this work, we propose several foundational concepts, which can be used for creating a radically new kind of architectural framework, which focuses on engineering of information systems, but simultaneously takes into account critical issues of Enterprise Engineering (EE) (Dietz and Hoogervorst, 2012).

Our conceptualization is based upon stratification of the various kinds of information systems. Traditional Information Technologies Systems (IT-S) occupy the bottom level of our classification. Above the notion of IT-S we put a stratification layer for the significant notion of Information Systems (IS) which refers to systems that capture some part or some aspect of the enterprise in a truthful and appropriate way. The third layer of stratification consists of Enterprise Information Systems (EIS), which specify systems capturing the enterprise as a group of individuals that cooperate to achieve some common goal, a product or service for a customer. It appears that Enterprise Information System gives a practical opportunity to integrate common IT concepts and intrinsic characteristics of human being (like responsibility, authority, sincerity and competencies). The fourth level is the new concept of Enterprise Operating system (EOS), an EIS, which controls the business transactions operation in an organization. Using a computer engineering metaphor, this paper defines the theoretical foundations, and the methodology to design and implement an operating system for organizations. Hence, the EOS controls the human enterprise in a prescriptive way (similar to workflow) and provides all atomic data to the information system that run on top of this EOS.

From a theoretical perspective, to specify some phenomenon in the world of phenomena using an ontology we need a conceptual language. Therefore, the proposed architectural framework is intended to design and implement EOS, considering the following aspects, which constrain the design space of EOS:

1.Epistemological foundations.

2.Ontological foundations.

3.Enterprise Modelling.

4.Software Development.

In order to prove relevance and consistency among all four aspects we propose to apply the concept of C4-ness (Dietz, 2006). In our case it means that, the proposed framework uses minimal, but complete, number of concepts, which cover principal aspects of EOS development.

In this framework, as any other engineering domain, we use models. We define a model here as a formal specification of some system that is being used to study some other system. For EE we study social systems (Dietz and Hoogervorst, 2012). This applies for the defined information systems, IS's, EIS's and EOS's that capture some domain of the world of phenomena. Within that domain, there is a specific phenomenon or aspect, which must be represented by a specific formal model. The term 'formal' refers to the representation of a model in a formal language and rules out any informal illustrations, such as boxes and arrows or ambiguous natural language.

In addition to the functional definition of a model, - a model represents a system to study some other independent system -, there are two commonly used meanings of the term `model` in general parlance. The first is that a model is a conceptualization of a phenomenon in some domain in the real world; a model references objects in the real world. The second meaning is the use of the term `model` also for a formal representation, a specification S, expressed using symbols and their relations in a formal language. If the modeling language is being used for software generation then the terms 'software primitive' and 'software construct' are also being used often. Though the use of the term 'model' for a conceptualization and for a symbolic specification S may be potentially confusing, it is clear from the context what is meant. To capture some domain in the world of phenomena in a truthful and appropriate way we need an ontology (Dietz and Hoogervorst, 2012). To specify models in a language we need conceptual languages. The notions of ontology, conceptual languages, models expressed in conceptual languages and the

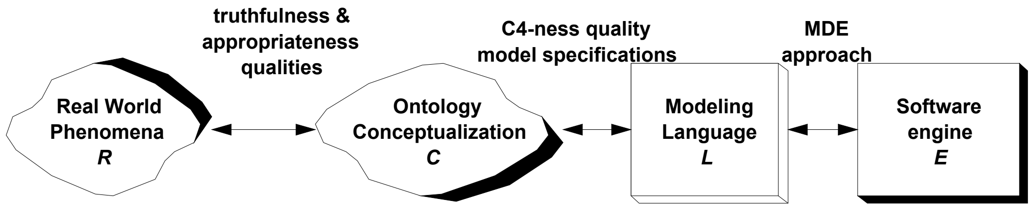
truthfulness & appropriateness qualities
C4-ness quality model specifications
MDE approach
Real World Phenomena R
Ontology Conceptualization C
Modeling Language L
Software engine E

Figure 1: The GSDP-MDE framework (simplified), derived from Guizzardi's framework.

quality of models, related to phenomena in the real world, are closely related.

The Generic Systems Development Process (GSDP) (Dietz, 2006) approach for engineering of systems of any kind is compliant with the technological theories for designing and implementing things (Dietz and Hoogervorst, 2012). The GSDP is a generic model for developing any kind of artifact, including software systems, and is followed here. In (Van Kervel,2012) is described how the GSDP applied to software engineering, using the Guizzardi framework, provides the Generic Systems Development Process for Model Driven Engineering (GSDP-MDE) framework for a model-instance driven software engine, a simplified version shown in **Fig. 1**.

The *Guizzardi* framework is extended with a software engine E, with systems ontology. The ontology C is the Using System (US) ontology of the GSDP framework, in our case Enterprise Ontology. The ontology C captures the real world in a truthful and appropriate way and provides models with C4-ness qualities. We design a modeling language L with a meta model (software primitives and software constructs) that is isomorphic to the ontology C. In this way any models expressed in L offer the advantages mentioned before. We may design a model executing software engine of which the systems ontology is isomorphic to the meta-model of L (and the ontology C).

We present the approach to devising the architectural framework for EOS as follows. In Section 2, we introduce the detailed conceptual structure of the proposed framework. In Section 3 describes the professional production case of application of the proposed architectural principles to development of complex information systems. Finally, we discuss the results of research and make a conclusion in Section 4.

2 DESCRIPTION OF THE PROPOSED FRAMEWORK

This section describes the framework for the enterprise operating system using textual explanations and the conceptual map depicted in **Fig. 2**. The definitions are divided into 3 clusters: *(i)* scientific solution grounding, *(ii)* solution enterprise environment and *(iii)* core solution. The aim of presenting these definitions is to ease the communication and discussion between the EE researchers and practitioners / engineers. A formal ontology using DEMO is expected to be further developed accordingly with the ongoing research activities of this proposal. At this point, it represents the knowledge acquired by the research team and configures the mandatory set of definitions to explain its scope and goals.

Regarding the scientific solution grounding the following definitions are, Architectural framework (Greefhorst and Proper, 2011), Epistemological foundations, Ontological foundations (Dietz, 2006), Enterprise Modeling methodology, Software Development of EIS and Model Driven Environment (Kent, 2002); (Schmidt, 2006).

In the proposed architectural framework we wish to explicitly introduce stratification among the different kinds of information systems of modern organizations. Such stratification facilitates systematic assessment of information assets of the organization, as well as helps in determining such strategy development, which correlates with a current level of IT-maturity. The definitions related with the solution enterprise environment aims at locating the scope and context where this proposal is applicable. The proposed definitions of recognized types of information systems are:

Information Technology System (IT) – is the whole body of systems to get computers doing something, not specifying some functional purpose in the "world of phenomena". It encompasses, for instance, disc drives, operating systems, languages, databases. It encompasses systems that fulfill a functional purpose within this scope.

Information System (IS) - refers to systems, built on IT systems, which capture in a truthful and appropriate way some part of the “world of phenomena”. In most cases, IS's are known in the form of Management information systems (MIS)

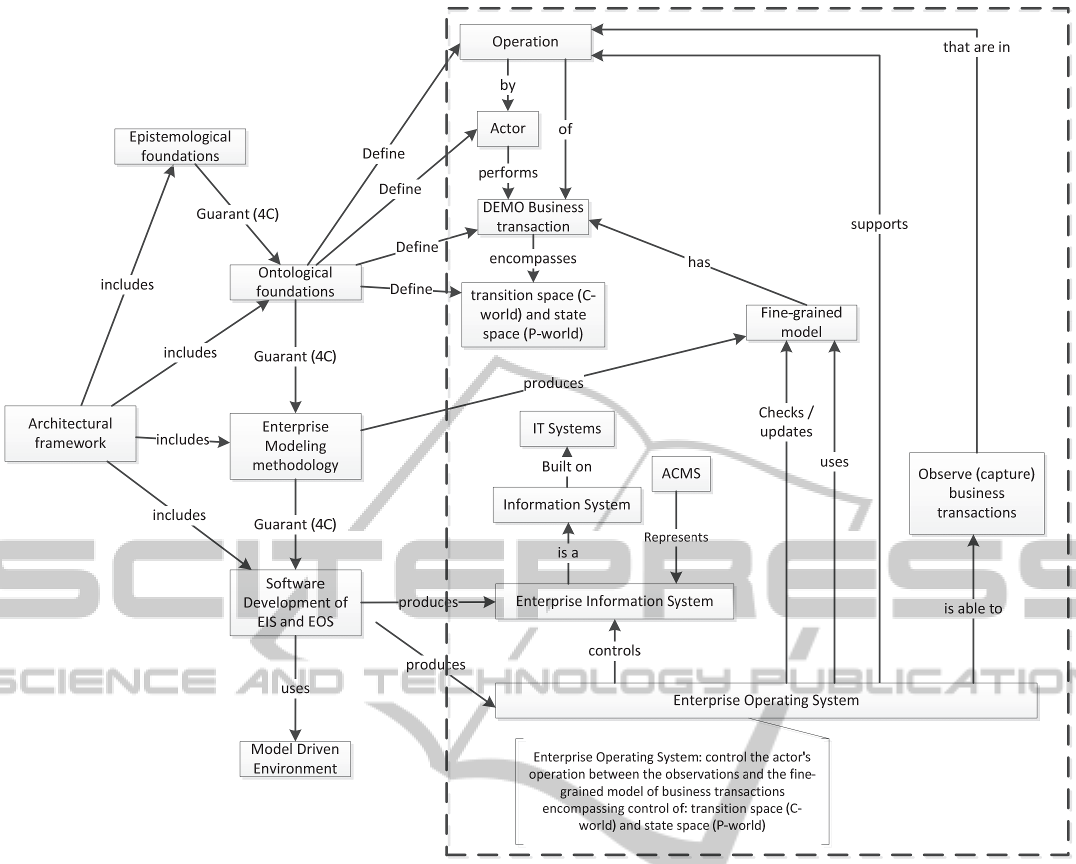
Operation
that are in
by
Actor
of
Epistemological foundations
Define
Define
performs
DEMO Business transaction
supports
Guarant (4C)
Define
encompasses
has
Ontological foundations
Define
transition space (C-world) and state space (P-world)
Fine-grained model
includes
includes
Guarant (4C)
produces
Checks / updates
IT Systems
Architectural framework
includes
Enterprise Modeling methodology
Built on
ACMS
uses
Observe (capture) business transactions
Information System
includes
Guarant (4C)
Represents
is a
Software Development of EIS and EOS
produces
Enterprise Information System
is able to
controls
produces
uses
Enterprise Operating System
Model Driven Environment
Enterprise Operating System: control the actor's operation between the observations and the fine-grained model of business transactions encompassing control of: transition space (C-world) and state space (P-world)

Figure 2: Framework overview. The definitions to operate EOS are inside the box in the right side. In the left side, are the concepts related with EOS solution grounding.

(Laudon and Laudon, 2012). In our case organizations, social systems that operate in a part of the world of phenomena are of specific interest. Since organizations, companies, legal entities etc., are complex entities with many different appropriate views (finance, personnel, inventory, etc), typically many different IS's are needed to capture an organization well. The term 'truthful' refers to the requirement that the representation provided by the IS is at any point in time, completely and in detail, true. The term 'appropriate' refers to a functional quality; the requirement that the IS should support the enterprise and its stakeholders well and in a useful way. IS's may capture also dynamic phenomena, phenomena that exhibit state transitions within a defined and allowed discrete state space must be represented in a truthful way. In practice, such dynamic phenomena represent agile enterprises; we observe in general that these phenomena evolve over time in a unpredictable way. To meet definitions, IS's should evolve also smoothly over time in such a way that the evolving phenomenon of agile enterprise is always captured in a truthful and appropriate way.

IT and IS relationship has significant importance. Actually, by the literature review it is consensual that enterprises require alignment between its IT systems and the business definition (Hoogervorst, 2009) whereas this alignment demands the consideration of aspects such as human, organizational, informational and technological (Mulder, 2007). In line, a well-known industrial standard: Archimate 2.0 (Open Group, 2013) uses three core aspects to model and align the enterprise: (i) the business architecture, (ii) the Information Systems (IS) architecture and (iii) Information Technology (IT) architecture. IS are defined by (Laudon and Laudon, 2012) as a set of interrelated components that collect (or retrieve), process, store, and distribute information to support decision-making and control in an organization. In addition, IS's may also help managers and workers analyze problems, visualize complex subjects, and create new products. On the other hand, IT systems are referred in (Laudon and Laudon, 2012) by the set of hardware, software, database, networking along with the tools and techniques for security and control, existing in the enterprise. Hence, a clear conceptual separation is understood in the literature regarding IS and IT. However, IS and IT are related because IS is built using IT and strictly depends on it.

Enterprise Information System (EIS) - refers to systems, built on IS's, that capture specifically an enterprise. An enterprise is defined as a social system, a group of human individuals – actors - that cooperate and communicate to achieve some common goal, a product or service for an external customer (Dietz and Hoogervorst, 2012). This definition is narrower than the notion of an organization, for example a company, with production facilities, bookkeeping etc. The notions of enterprise, actors, communication and production are precisely defined in the EE manifesto (Dietz and Hoogervorst, 2012). EIS's provide two distinct perspectives on an enterprise. The first is a descriptive perspective that provides a complete, detailed truthful representation of the operating enterprise. Since enterprises exhibit a discrete dynamic behavior, actors are communicating, there are acts and facts, the states and state transitions that have to be represented also. The second is a prescriptive perspective. An EIS is based on a model of the enterprise and prescribes the behavior of the enterprise to act according to the model (Dietz, 2006). The EIS is a software engine that executes a business transactions model instance; for each specific production instance there is a specific model instance under execution. An EIS prescribes, enforces all actors of the enterprise to act, the so-called communication acts (Dietz, 2006), within the allowed state space and state transition space of the model under execution. There is no possibility for any actor to deliver communication acts outside the allowed state space. This is called enforced compliance of an enterprise to a model. The allowed state space is defined by and calculated from the business transaction models under execution. This prescriptive capability is functionally equivalent to state of the art workflow systems; there is enforced compliance of all actors to a model. Unlike state of the art workflow systems, the workflow capability is here calculated from the enterprise model under execution, not specifically modelled. The purpose of EIS's is to capture working, implemented enterprises in full detail. To achieve this, the ontological model of the enterprise is – after validation and acceptance by the stakeholders – extended with all relevant info-logical and data-logical transactions. This modeling process is called fine-grained business transactions modeling and is an implementation of an ontological model. There are typically more possible implementations of an ontological business transaction model and finding the optimal implementation for a specific enterprise is an important process. The support of business transaction is DEMO (Dietz, 2006), which is a model under execution that provides a complete, detailed truthful representation, including the current state, of the operating enterprise.

Enterprise Operating System (EOS) - encompasses an EIS, a software engine that executes a (fine-grained) DEMO model and in addition (to an EIS) provides support for all non-EIS IS's in an organization, company, entity etc. In software engineering an operating system of a computer system monitors and controls all hardware subsystems and provides a generic, abstracted and hardware independent interface to the subsystems for all software applications. Similarly, an EOS captures and controls (Guerreiro et al., 2012) all phenomena in an enterprise and provides all required data for all IS's for that organization.

The theory of enterprise ontology, EO, the operation axiom (Dietz, 2006), specifies three worlds of an organization; the A-world of actors, the C-world of communication between actors, and the P-world of productions. An EIS provides – as described before – control of all communication acts, which is the C-world, for all actors, which is the A-world. The theory of EO captures also the P-world of productions; the composition axiom specifies the hierarchical structure and the aggregation of productions that are performed by specific actors. DEMO models define the productions and those actors with their communication acts about their productions. The actual production, the so-called production facts, is explicitly not executed by an EIS; only the sequence of the production facts is controlled. If an IS is required for specific production facts, for example the calculation of some specific data, or the measurement of quality, then an independent IS, specifically designed for that purpose, is needed. The operation of these production oriented IS's is however monitored and controlled by the EOS. An EOS controls and monitors all atomic elementary communication and production acts and facts (which is the EIS capability). An EOS has *“total factual knowledge”* about *“anything that is controllable, until the finest details, that happens in the enterprise”*. An EOS therefore provides the perfect bridge between the organization as phenomenon in the real world and all IS's that capture some aspect or view of that organization (finance, inventory control, personnel information etc).

Adaptive Case Management Systems (ACMS) – as one of many different kinds of IS's for production systems, ACMS's are a combination of an EOS and one or more (non-EIS) IS's, supported by the EOS, that capture together some parts of an operating organization. The EOS captures, as described already before, the operation of the enterprise. The operation of the enterprise involves the actors, their communication, all precisely specified, enforcing compliance to the enterprise model being executed. The IS's for ACMS's are typically document-oriented production environments, for the creation of documents, forms, decisions, datasets from legacy applications, etc. ACM's are typically aimed at the production of services, where the customer of the organization is an active co-producer. The services are tailor-made and adapted to the needs and requirements of the customer, as opposed to mass production. The production is subjected to so-called business rules derived from legal regulations, contracts etc.

Almost all defined kinds of systems in our framework represent complex socio-technical systems where involvement of human factors is highly influential. Many analysts and system thinkers argued that for achieving the design goals of such systems developers need much more higher levels of epistemological truth guarantors in comparison with traditional technical systems (rephrase of the words by Eric Maskin, Nobel Prize Laureate in Economics Design). Including into the proposed framework several hierarchically structured foundations reflects this aspect. In our particular case, we suggest to narrow a generic concept of guarantor towards specific definition of C4-guarantor. Further, in Section 3 we will describe in details the proposed principles.

Notion and interrelations of these foundations correspond to the core principles of inquiring systems design by (Churchman, 1971). Epistemological foundations are needed to align and verify IS, EIS and EOS solutions with expectations, values and norms of enterprise stakeholders. In the particular case of our framework, we offer to use the (Habermas, 1984) (Habermas, 1987) theory of communicative action. Ontological foundations determine cornerstone concepts of IS, EIS and EOS which can be treated as meta-models. As ontological foundations we suggest to exploit the PSI-theory of Enterprise Ontology by J. Dietz (Dietz, 2006) which is tightly connected with the chosen epistemological foundations of (Habermas, 1984) (Habermas, 1987). In own turn, Enterprise Modeling aspect determines concrete stages and outcomes which comprise concise and coherent description of all relevant parts of IS, EIS and EOS. On that stage, we suggest to use in our framework such modeling methodology as DEMO (Dietz, 2006) for which Enterprise Ontology naturally becomes a truth guarantor.

The major part in the framework is the architectural aspect of Software Development, which determines generic requirements and shapes how executable software artefacts can be produced on the basis of the enterprise models. To elaborate that aspect we propose to revise too generic OMG principles of model-driven design and development and replace them by the GSDP-MDE methodology (Van Kervel, 2012) (Van Kervel et al., 2012)supported by a number of specialized system components. The components include (1) run-time executor of DEMO models; (2) run-time rule executing engine; (3) Control component; (4) Policy enforcement component; (5) Argumentation and collective work component. For each component we propose to use the following architectural principles and solution core definitions.

Operation – "the collective activity of the elements in the composition and the environment is called the operation of the system. Thus, one could also say that the operation of a system is the manifestation of its construction in the course of time. It encompasses both the productions as performed by the elements in the composition and the interactions through the structural bonds. The operation of a system can be described by specifying (for every element in the composition) the action rules that guide or prescribe the activity of the element" (Dietz, 2006).

Actor - specify a role who is responsible for each part of the transaction, who initiates it and who executes it (Dietz, 2006).

DEMO Business Transaction - follows the definition given for a transaction in Enterprise Ontology (Dietz, 2006), where a transaction is the set of coordination acts that are performed as steps in a universal pattern, always involving two actor roles (the initiator and the executor) and are aimed at achieving a particular result. Here the enterprise models, by the mean of DEMO business transaction models may capture much more than the strictly ontological DEMO models that are fully implementation independent

State Space and State Transition Space (C-world) – The state space is the set of allowable states of a system. The transition space is the set of allowable sequences of transitions of a system. Every state transition is only dependent on the actual state

(Dietz, 2006).

Fine-grained Model – A model stands for the definition given by (Apostel, 1960), where: "Any subject using a system A that is neither directly nor indirectly interacting with a system B, to obtain information about the system B, is using A as a model for B". Fine-grained model stands for the concern of specifying all the minor details in respect to the stratification that is being considered at each time, but using as starting point the ontological models that have the advantage of being implementation independent. In practise, implementation is the extension of the implementation-independent ontological model with implementation-specific transactions and actors, also called fine-grained modeling.

Observe (capture) Business Transactions – the collection of steps and facts within DEMO business transactions that actors are performing during operation are observable. There are parts of a business transaction that are observable, while others are unobservable (Guerreiro *et al.*, 2012). Hence, not all the state space and transition space of the enterprise is observable directly in the operation of an enterprise.

3 ADAPTIVE CASE MANAGEMENT SYSTEM: PROFESSIONAL CASE STUDY

The first EOS, now in industrial production, supports an adaptive case management system (ACMS) for a Dutch semi-public company that delivers energy and utility services, such as water and electricity, for citizens. The complex tailor-made contract for a specific individual customer covers issues such as type of services provided, costs, costs calculation methods, conditions for payments, instructions for subcontractors, correspondence etc. The contract should comply with external legal regulations and internal business policies, conditions, procedures. This compliance should be enforced to the staff and its compliance has to be proven for each individual customer and contract for legal reasons. The quality of the contracts should be high; errors are unacceptable, incomplete transactions, deadlocks, deadline overflow, violation of procedures etc, must be eliminated. This is a very complex business process with 33 production steps / transactions. The EOS (Enterprise Operating System) executes fine-grained DEMO models, enforces compliance to business rules, and supports the ACMS. The ACMS is an IS for the production and management of documents, technical drawings, interfaces to legacy IT systems, including an ERP production control system. The contract contains documents originating from various sources and several subcontractors.

The project started with DEMO modeling with the knowledgeable staff, which delivered the so-called ontological or essential DEMO model of the enterprise. Extensive shared reasoning with the staff has been done to validate that this ontological DEMO model implements properly the service of the enterprise delivered to the customer. The second step involved a fine-grained DEMO modeling step, starting from the ontological DEMO model, to model the optimal detailed implementation of the ontological model. The result is a DEMO model with 33 ontological, 'infological' and 'datalogical' transactions that has been subjected to extensive simulations with the staff for acceptance. No software programming was done yet. The accepted DEMO model provides C4-ness quality specifications for the ACMS. Specifically it provided a constructional decomposition of the ACMS with a high degree of granularity. The ACMS consists of some 30 independent software components with a minimized complexity and a minimized interdependency. The C4-ness quality specifications resulted in low efforts of software implementation and good support of validation. The acceptance of the ACMS was straightforward and the functional quality, the business-IT alignment, was found to be high.

This first case, in line with the proposed framework, shows the technical feasibility of the EOS for ACMS, applied for complex governmental services, financial services etc. The design and implementation of these type of IS's is highly structured, systematic and provides good opportunity to validate the business-IT alignment before implementation of any software programming.

4 CONCLUSIONS

In this work, a new conceptual structure of software architectural framework is proposed. Such framework allows reducing complexity and ambiguity during engineering of several classes of information systems. In particular, the framework is expected for using during engineering of Enterprise Operating systems (EOS). Authors argue that the whole framework satisfies the C4 principles and follows the course of Enterprise Engineering. The EOS controls and monitors all atomic elementary communication and production acts and facts that are observable, in order to assess the *"world of phenomena"* occurring in the organization against the predefined fine-grained enterprise models that are consensual agreed between the stakeholders,

Using proposed architectural principles, we envision continuing in developing a specific kind of EOS, which has major characteristics of Adaptive Case Management systems supported by the requited architectural framework components. This particular set of systems will be used to boot the research of a Generic Systems Development for Model-driven engineering (GSDP-MDE) of Information systems. It is expected that the GSDP-MDE help in eliminating ambiguity, absence of anomalies, constructs overload and constructs excess. In addition, we will seek alternative sets of constraints for proposed aspects of our architectural framework, for instance, full observation of business transactions operation.

In our opinion, future results from this research could be applied to other Information Systems domains, since EOS conforms to a very consistent, harmonious and congruent architectural framework.

REFERENCES

Dietz J. L. G., 2006. Enterprise Ontology: Theory and Methodology / J. L. G. Dietz — B., Heidelberg, N. Y.: Springer, ISBN-10 3-540-29169-5.

Laudon K. and Laudon J., 2012. Management Information Systems, 12th edition, Prentice Hall.

D. Greefhorst and E. Proper, 2011. Architecture Principles, *The Enterprise Engineering Series*, DOI 10.1007/978-3-642-20279-7_2, Springer-Verlag Berlin Heidelberg.

Hoogervorst, J., 2004. Enterprise Architecture: Enabling Integration, Agility and Change. *IJCIS 13*(3), pp. 213–233.

The Open Group:, 2011. TOGAF Version 9.1; available at http://www.opengroup.org/architecture/togaf9--doc/ arch/; The Open Group.

J. Hoogervorst, 2009. Enterprise Governance and Enterprise Engineering, *The Enterprise Engineering Series*, Springer-Verlag Berlin Heidelberg.

Dietz, J. and Hoogervorst, J., 2012. The Principles of Enterprise Engineering, Springer-Verlag Berlin Heidelberg, Volume 110, Part 2, series Lecture Notes in *Business Information Processing, Enterprise Engineering Working Conference 2012 (EEWC 2012)*, DOI: 10.1007/978-3-642-29903-2, Delft, Netherlands, pp. 15-30.

Apostel, L., 1960. Towards the formal study of models in the non-formal sciences. Synthese 12.

Kent, 2002. S. Model-driven engineering. Proc. 3rd Int. Conf. on Integrated Formal Methods, pp. 286–298.

Schmidt, D. C., 2006. 'Model-Driven Engineering'. *IEEE Computer*, 39 (2), pp. 25–31

Mulder, J., 2007. Rapid Enterprise Design, ISBN-10: 9081048015

The Open Group, 2013. Archimate version 2. http://www.opengroup.org/subjectareas/enterprise/archimate.

Churchman, C. W., 1971. The Design of Inquiring systems: Basic concepts of Systems and Organization. *Basic Books*, Inc. Publishers.

Habermas, J., 1984. Reason and the Rationalization of Society, Volume 1 of The Theory of Communicative Action, English translation by Thomas McCarthy. Boston: Beacon Press (originally published in German in 1981).

Habermas, J., 1987. Lifeworld and System: A Critique of Functionalist Reason, Volume 2 of The Theory of Communicative Action, English translation by Thomas McCarthy. Boston: Beacon Press (originally published in German in 1981).

Kervel, S. J. H. van, Dietz, J. L. G., Hintzen, J., Meeuwen, T. van, Zijlstra, B., 2012. Enterprise Ontology driven Software Engineering. *Proceedings of ICsoft 2012 – 7th International Conference on Software Paradigm Trends*. SciTePress.

Kervel, S. J. H. van, 2012. Ontology driven enterprise information systems engineering. Ph.D. Thesis, ISBN: 978-90-9027133-0, Formetis.

Guerreiro, S., Vasconcelos, A., and Tribolet, J., 2012. Enterprise dynamic systems control enforcement of run-time business transactions, Springer-Verlag Berlin Heidelberg, Volume 110, Part 2, series Lecture Notes in *Business Information Processing, Enterprise Engineer-ing Working Conference 2012 (EEWC 2012)* Delft, Netherlands, pp. 46-60.

Ministry of Defence, 2010. Ministry of Defence Architecture Framework version 1.2.004.

Teka, A., Condori-Fernández, N., Kurtev, I., Quartel, D. and Engelsman, W. 2012. Change impact analysis of indirect goal relations: Comparison of NFR and TROPOS approaches based on industrial case study. *Procs. of the 2nd IEEE International Workshop on Model-Driven Requirements Engineering*, *MoDRE 2012*. pp. 58-67.