Tesperanto – A Model-Based System Specification Methodology and Language

Alex Blekhman

Technion, Israel Institute of Technology
Haifa 32000, Israel

blekhman@tx.technion.ac.il

Dov Dori

Technion, Israel Institute of Technology
Haifa 32000, Israel
Massachusetts Institute of Technology,
Cambridge, Massachusetts, USA

dori@ie.technion.ac.il

Abstract. Technical requirements, specifications, and standards are fundamental documents that serve all the stages of the systems engineering management process. This paper presents Tesperanto – "Technical Esperanto" – a model-based methodology and language for authoring, analysis, design, and testing of system specifications, and its integrated development environment. Tesperanto is an English text that is generated automatically from the evolving graphical model of the system being specified, creating corresponding text-model representation of technical information. Tesperanto was evaluated in a case study over a period of more than two years by a multi-national group of systems engineering and enterprise management experts as a means to enhance text in ISO international standards. The evaluation showed that Tesperanto improves the consistency and quality of standards and can potentially be of similar value for other kinds of technical documents.

1. Introduction

The rapid development of system complexities has accelerated the pace of searching for an intuitive yet formal way of analyzing requirements and designs of new systems, as well as specifying technical information over a wide spectrum, from description of systems and system components, conditions and processes, through structure and process definitions, to testing and execution instructions, and operation guidelines to be followed.

An illustration for the increasing complexity of current technical texts can be seen in Figure 1, which shows a model of functional enterprise-control interface and relevant functions, as specified in the IEC 62264 standard (paragraph 6.3, figure 5).

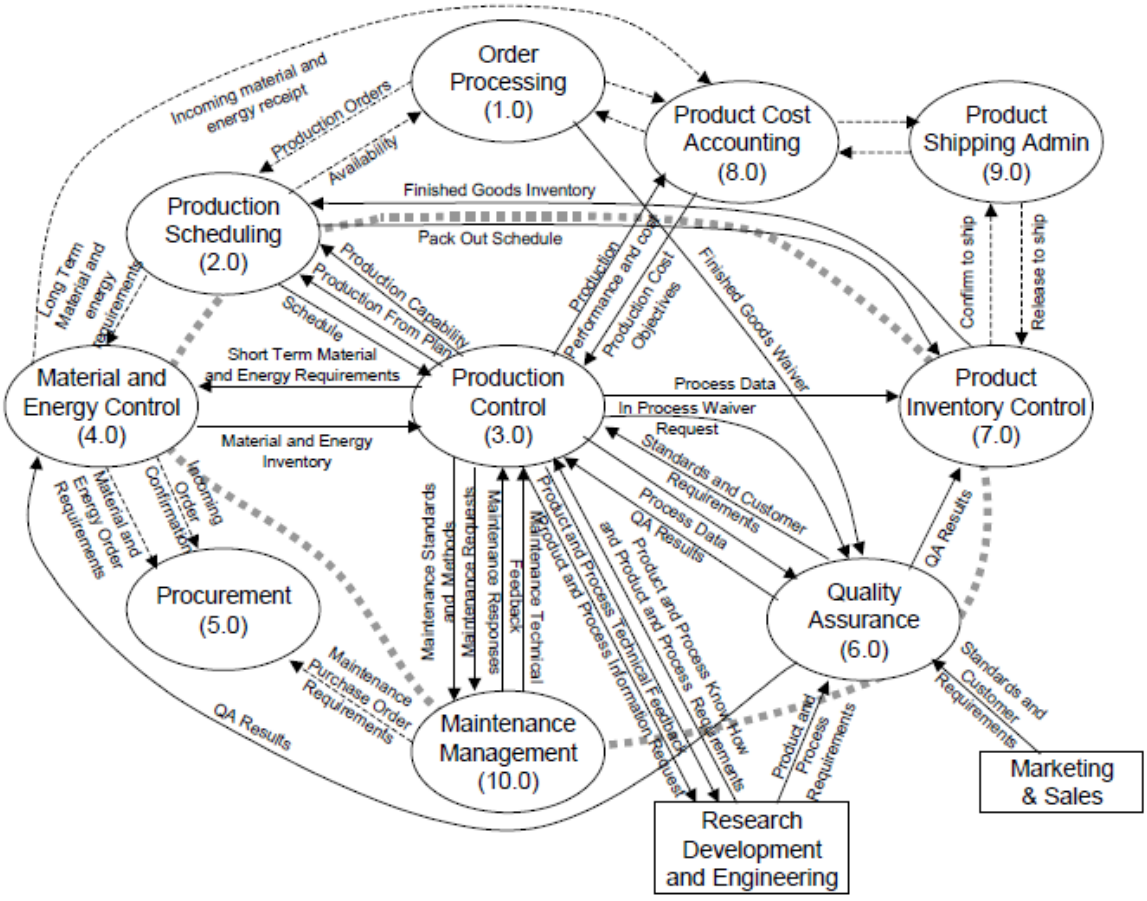

Figure 1: IEC 62264-1 – Functional enterprise-control model

The abundance of elements, listed in the informal, free-form diagram of Figure 1, renders the task of maintaining consistency of the standard nearly impossible. Indeed, manual analysis we performed in the early stages of Tesperanto development revealed numerous inconsistencies in the current free-text IEC 62264 standard specification. Some of these deficiencies were presented in (Blekhman, Howes & Dori 2010). Other examples of the level of inconsistencies found in a real published standard through applying early Model-Based Systems Specifying techniques are found in (Dori et al., 2010).

In general, technical specifications aim to convey structured, exact, and supposedly well-defined information. Although scientific, technical, and industrial specifications are regularly expressed in free text, they commonly describe an underlying scheme, be it a block diagram, a flow chart, a state machine, or some informal, ad-hoc schema. In many cases, this hidden structure bears the true value of sometimes wordy and confusing specifications. Indeed, nearly all technical documents contain accompanying figures, diagrams, schemes, or drawings that illustrate and complement the text, providing a visual reference for the textual content.

With this in mind, we wish to uncover, rely on, and work with the actual scheme underlying the technical document text, rather than interpret the free text with its accompanying disparate figures and mostly informal diagrams. The combination of underlying model and text that relies on that model allows for the document content to be presented in an unequivocal manner, providing an analytical representation of the authors' intent. Building on a formal model can potentially render structured document authoring and analysis a much easier process. The model can serve as a solid, consistent and unambiguous basis for analyzing the system, comparing it to models of other systems, and even generate code for

the relevant software-intensive portions of the system.

Following the model-based systems engineering (MBSE) paradigm and the ideas presented above, we have developed a prototype version of Tesperanto ("Technical Esperanto") – a model-based system specification methodology and language with a supporting integrated development environment (IDE).

The rest of the paper is organized as follows: Section 2 presents model-based system specification principles underlying Tesperanto. Section 3 demonstrates the capabilities and functions of the Tesperanto integrated development environment (IDE) that supports model-based system specification. Section 4 describes the evaluation of the methodology by a group of ISO experts in the course of authoring technical standards case study. Section 5 concludes the presentation and discusses future work.

2. Model-Based System Specification principles

2.1 Conceptual modeling and model-based systems engineering

The process of representing system-related knowledge in a model is referred to as conceptual modeling, and the outcome of this activity is a conceptual model 0(Thalheim 2011). The vision of the Massachusetts Institute of Technology Engineering Systems Division (MIT ESD 2012) states clearly what can be taken as the motivation for conceptual modeling – that "the fundamental principles and properties of engineering systems—the complex socio-technical constructs that are the foundations of modern society—are well-understood, so that these systems can be modeled, designed, and managed effectively."

Modeling is key to model-based systems engineering (MBSE). INCOSE systems engineering vision 2020 defines MBSE as "…the formalized application of modeling to support system requirements, design, analysis, verification and validation activities, beginning in the conceptual design phase and continuing throughout development and later life cycle phases" (INCOSE 2007).

Understanding systems of all kinds requires a well-founded, formal, yet intuitive methodology that is capable of modeling their complexities in a coherent, straightforward manner. The same modeling paradigm can serve for both designing new systems and for studying and improving existing ones.

2.2 Model-Based Systems Specifying Goals and Underlying Formalism

The research that inspired the development of Tesperanto was initiated by ISO – International Organization for Standardization. ISO standards, as well as those of other standards organizations, and other technical system specifications, are often criticized as being difficult to use for a variety of reasons, including inter- and intra-standard inconsistency, low accessibility, poor traceability, and ambiguity (Dori et al, 2010). Given the variety of authors and relationships to other domains on top of the usual hurdles renders managing the quality of a technical document such as an ISO standard or a system specification a daunting task.

The main goal of the Tesperanto approach is to improve the quality of technical documents through human-readable textual representation that is synchronized with an underlying conceptual formal model of the document's technical content. This approach ensures that the

technical text is significantly less likely to suffer from undesirable attributes that include ambiguities and contradictions that are common in a text-based specifications (with or without possible accompanying graphics), as exemplified by Dori et al. (2010). Specifically, Tesperanto Model-Based System Specification methodology is intended to increase the internal and external consistency of the documents that is defined as follows:

1. Internal consistency – the extent to which facts stated within the technical specification are aligned and do not contradict each other.

2. External consistency – the extent to which facts stated in different related technical specifications pertaining to the same domain and/or system are aligned and do not contradict each other.

The underlying formalism for Tesperanto should be a conceptual modeling framework that would be general enough to fit for a wide set of systems and domains. Further requirements include expressiveness of the notation for defining common conceptual models, formalism, and clear semantics. Since the potential users come from a wide range of disciplines, the notation should also be simple and intuitive.

Following the survey of knowledge representation approaches in (Dori 2004), which included concept maps, semantic networks, conceptual graphs (CGs), Knowledge Interchange Format (KIF), the Common Logic (CL) Standard initiative, Resource Description Framework (RDF), Unified Modeling Language (UML) and Object-Process Methodology (OPM), we stick to the observations made there in favor of OPM (Dori 2002) as the underlying conceptual modeling language for Tesperanto. It is possible, though, to use Tesperanto with other modeling frameworks, assuming a mapping exists from OPM to that framework. Although OPM was found most suitable for overall text-model representation, aspects of a Model-Based Systems Specification methodology can be translated to other modeling languages and notations, such as UML, BPMN, and SysML. Indeed, Grobshtein and Dori (2011) have devised a mapping of conceptual entities from OPM to various SysML diagrams.

2.3 Tesperanto Model-Based System Specification methodology

The Tesperanto Model-Based System Specification methodology follows MBSE principles and is based on corresponding and simultaneous text-model representation that evolves the bimodal representation of OPM (Blekhman, Dori & Martin 2011). Focusing on the system's overall function and the value it is designed to deliver, function-structure-behavior unification of OPM provides a coherent single frame of reference for representing the technical content of the system and tracking objects, processes, states, and static and dynamic relations among these entities. The model, expressed in corresponding graphics and text, holds facts—information about what the system does and how it does it—in terms of these entities and links among them. The text and the graphical model coexist as two complementary and interlinked modalities; changing the model in one modality triggers an immediate and automatic change in the other, ensuring that both are maintained equivalent at any point in time. This specifically addresses ambiguities, contradictions, internal and external consistency of specifications and documents, as the underlying model of technical content supports only uniform and coherent definitions, and visualizes inconsistencies and missing or inaccurate details by comparison to accompanying text.

A cornerstone of the Tesperanto methodology is the Tesperanto language. Tesperanto is an enhancement of Object-Process Language (OPL), a subset of English, constrained by a context-free grammar that is auto-generated from OPM diagrams. OPL was not explicitly meant to become a text to be read by a general audience as it consists of short, often

disconnected sentences. OPL may be good for lower-level machine-oriented tasks, such as code generation, but being mechanical and repetitive, with no text fluency, it is not natural for human reading. Thus, while each OPL sentence is a syntactically and semantically correct English sentence, lack of fluency from one OPL sentence to the next prevents OPL from becoming a descent substitute for the free text that dominates real-life specifications. This has motivated the development of Tesperanto as the next level of automatic model-based text-from-graphics generation on top and instead of OPL.

The basic Tesperanto language rule is that each Tesperanto phrase has its graphic counterpart as a model fact. Therefore, each sentence, phrase or token can be traced down to its representation in model. By default, Tesperanto follows Breadth-First Search (BFS) presentation style. Informally, this results in specifying by text all the links that are closest to a thing (object or process), and only then moving to farther links and lower-level things.

A hierarchy of object types (agents, instruments, resultees etc.) is defined for automatic model-to-text and text-to-model translation, but text order can be changed by the user, as long as the link to the relevant model fact remains. Furthermore, the user has the ability to change the automatic wording for links, and define synonyms for any model fact, since a common limited set of links and entities was found too generic for the establishment of exact translation algorithm that would be appropriate for diverse systems of different kinds, processes and domains. Instead, the best capture of the information need is through guiding the user to specify or edit the auto-generated Tesperanto text based on model facts until the text is clear and well-formed in concurrence with diagram facts.

The Tesperanto text generation module follows OPM's gradual presentation principles, which cater to humans' cognitive limited capacity (Dori 2002). Following OPM guidelines of complexity management and context maintaining, Tesperanto is capable of text-from-model and model-from-text generation and includes heuristics for sentence length adjustments, synonyms, word ordering, phrase recurrence control, and other algorithms aimed at making the Tesperanto text look less mechanistic and more human readable. As examples below demonstrate, the Tesperanto text generation module faithfully reflects the model facts in formal and verified OPM graphical model while being humanly readable, without being too boring, repetitive, and mechanical. Conversely, the OPM graphical model represents all the facts indicated in the Tesperanto text.

Figure 2 (the graphical OPM model) and Figure 3 (the corresponding Tesperanto text) formally define the Tesperanto methodology at a high level. This specification is reflective in the sense that it applies on itself the rules of dual text-model specification it prescribes. As the model in Figure 2 expresses, a model-based system specifying process that follows this methodology relies on a domain ontology reference base, which is continuously updated and extended. Accordingly, Figure 3 is an automatically-generated Tesperanto paragraph that corresponds to the model in Figure 2.

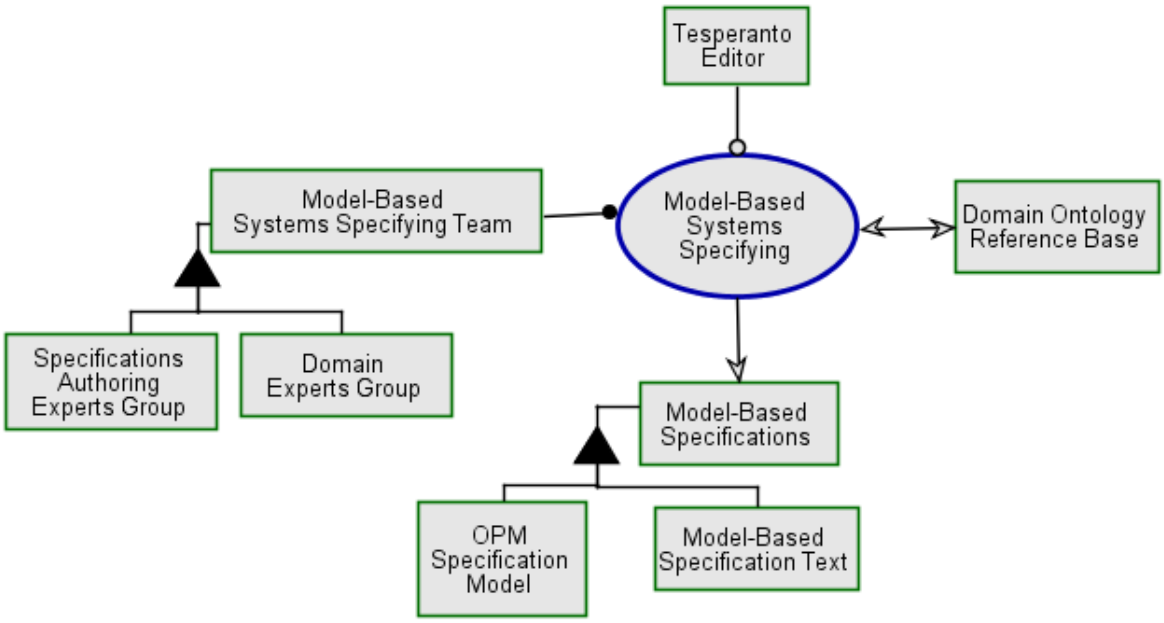

Figure 2: Object-Process Diagram (OPD) of the Model-Based Systems Specifying process

Model-Based Systems Specifying is the process of creating **Model-Based Specifications**, which consist of an **OPM Specification Model** and a **Model-Based Specification Text**. This process affects a **Domain Ontology Reference Base**, with the aid of **Tesperanto Editor**. It is performed by a **Model-Based Systems Specifying Team**, which consists of a **Specifications Authoring Experts Group** and a **Domain Experts Group**.

Figure 3: The Tesperanto paragraph that was auto-generated from the Model-Based Systems Specifying process model in Figure 2

Multi-level Tesperanto generation is possible by joining the corresponding paragraphs for upper-level and in-zoomed diagrams. Further elaboration by zooming into the model-based systems specifying process is provided in Figure 4 (model) and Figure 5 (text). For the sake of comparison, Figure 5 lists on the left the OPL text and on the right the Tesperanto text, both generated automatically from the OPD in Figure 4. Obviously, the Tesperanto version is much more human-readable, but it is also longer (129 vs. 154 words).

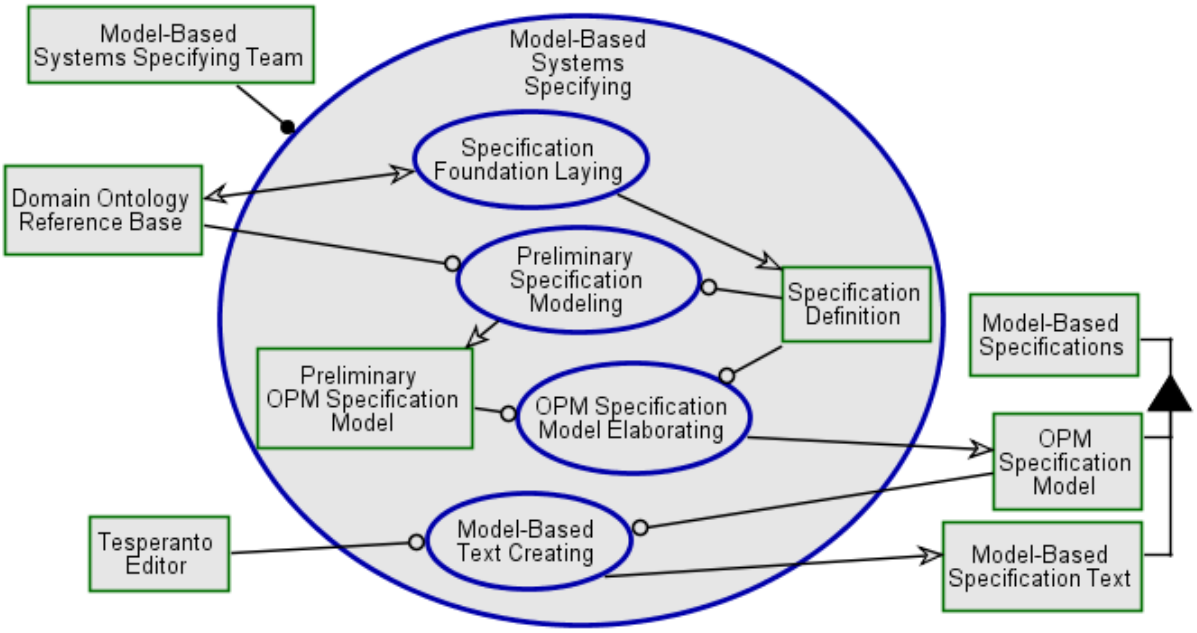

Figure 4: A detailed OPD obtained by zooming into Model-Based Systems Specifying

OPL	Tesperanto
Model-Based Specifications consists of **OPM Specification Model** and **Model-Based Specification Text. Model-Based Systems Specifying Team** handles **Model-Based Systems Specifying. Model-Based Systems Specifying** exhibits **Specification Definition** and **Preliminary OPM Specification Model. Model-Based Systems Specifying** zooms into **Specification Foundation Laying, Preliminary Specification Modeling, OPM Specification Model Elaborating,** and **Model-Based Text Creating,** as well as **Preliminary OPM Specification Model** and **Specification Definition. Specification Foundation Laying** affects **Domain Ontology Reference Base. Specification Foundation Laying** yields **Specification Definition. Preliminary Specification Modeling** requires **Specification Definition** and **Domain Ontology Reference Base. Preliminary Specification Modeling** yields **Preliminary OPM Specification Model. OPM Specification Model Elaborating** requires **Preliminary OPM Specification Model** and **Specification Definition. OPM Specification Model Elaborating** yields **OPM Specification Model. Model-Based Text Creating** requires **Tesperanto Editor** and **OPM Specification Model. Model-Based Text Creating** yields **Model-Based Specification Text.**	**The Model-Based Systems Specifying** process is comprised of the following four subprocesses: (1) **Preliminary Specification Modeling**, (2) **OPM Specification Model Elaborating**, (3) **Specification Foundation Laying**, and (4) **Model-Based Text Creating**. This process is performed by a **Model-Based Systems Specifying Team**. **Specification Foundation Laying** is the process of creating a **Specification Definition**, also affecting a **Domain Ontology Reference Base**. **Preliminary Specification Modeling** is the process of creating a **Preliminary OPM Specification Model**, with the aid of a **Domain Ontology Reference Base** and a **Specification Definition**. **OPM Specification Model Elaborating** is the process of creating an **OPM Specification Model**, which is a part of a **Model-Based Specifications**. This process is performed according to a **Specification Definition** and a **Preliminary OPM Specification Model**. **Model-Based Text Creating** is the process of creating a **Model-Based Specification Text**, which is a part of a **Model-Based Specifications**. This process is performed according to an **OPM Specification Model** and a **Tesperanto Editor**.

Figure 5: Text generated from the OPD in Figure 2. Left – OPL. Right – Tesperanto

The reflective description of the **Model-Based Systems Specifying** process in Figure 2

through Figure 5 illustrates both the concept and the principles of text-model correspondence. Reading the Tesperanto text carefully can still expose the fact that it was generated automatically, and that its quality depends in part on the human modeler expertise. For example, consider the sentence "**OPM Specification Model Elaborating** is the process of creating an **OPM Specification Model**, which is a part of a **Model-Based Specifications**." It is grammatically wrong because the modeler used plural as a name of an object, which is not expected in OPM object naming.

Probable solution for this situation is changing the above Tesperanto text to singular, that is, **Model-Based Specification**; this change will automatically be reflected in the model. While the modeler works on the specification, the model keeps evolving, and the association between the text and the graphical model provides for inherent consistency and automatic update on changes.

2.4 OPM model construction

The purpose of the specification shall guide the scope and level of detail of an OPM model. A detailed specification may involve many stakeholders, including the beneficiary, owner, users, and regulators, as well as hardware and software components, exposing different aspects relevant to each stakeholder. A fundamental OPM guideline is that the function of the system is its top-level value-providing process, as perceived by the system's main beneficiary or beneficiaries group. The function is expected to provide value for the system's beneficiary, and this prescribes the system's purpose. The construction of the system model is based on functional decomposition of the system's function, which is its main process. This stepwise decomposition creates a hierarchy of interconnected object-process diagrams—the OPD process tree.

The system's function, which is the starting point of the OPM model, is expressed as the top-level process in the first object-process diagram (OPD), called the system diagram (SD). Such a deliberation, which precedes the definition of the system's function, is extremely useful, as it exposes different approaches, and often even misconceptions, among the various system stakeholders regarding the primary objective of, and value provided by, the system under design, and subsequently the content to be specified, modeled, and designed. Moreover, the value of the function to the beneficiary is often implicit, expressed in terms of a process, which emphasizes *what* happens rather than *for what*—the purpose for which this top-level process happens. Distinguishing between function and behavior is critical for the creation of a clear and unambiguous specification. This distinction is essential because in many situations a system's function is achievable by different concepts, each implementing a different architecture—combination of objects and processes.

Consider for example a simple process specified by the following text: "Welding two steel parts together creates the combined part."

This simple sentence can be easily presented as the Object-Process Diagram in Figure 6.

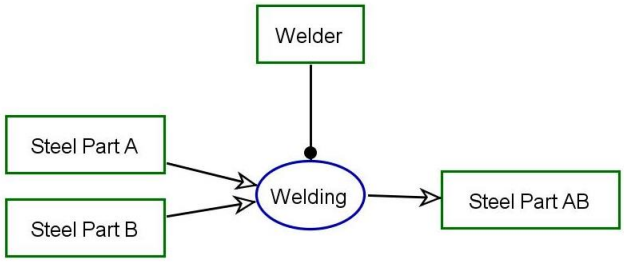

Figure 6: OPD for a basic Welding process

Obviously, the conversion of text to model introduced interpretation that is use-dependent. Converting the model in Figure 6 back to text provides the Tesperanto paragraph in Figure 7.

Welding is the process of creating a **Steel Part AB**, by a **Welder**. This process consumes a **Steel Part A** and a **Steel Part B**.

Figure 7: Tesperanto paragraph for a basic **Welding** process

If it is important to specify the fact that a gas metal arc is needed for this process, the model will change accordingly, as shown in Figure 8.

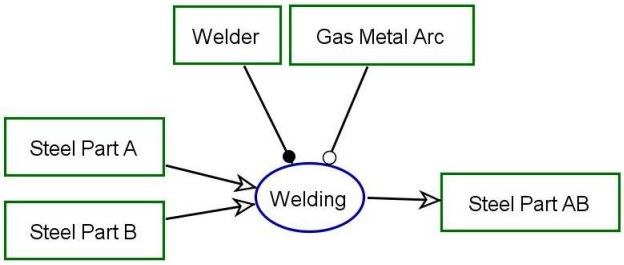

Figure 8: OPM diagram for Welding with a Gas Metal Arc process

The text corresponding to Figure 8 will automatically reflect the introduced change, as expressed in the Tesperanto text in Figure 9.

Welding is the process of creating a **Steel Part AB**, with the aid of a **Gas Metal Arc**. This process is performed by a **Welder**, consuming a **Steel Part A** and a **Steel Part B**.

Figure 9: Tesperanto paragraph for Welding with a Gas Metal Arc process

3. Model-Based System Specification IDE

3.1 OPM Modeling Tool

The Tesperanto modeling layer is based on OPCAT (Dori 2010), a modeling environment that supports OPM. Figure 10 is a screenshot of an OPCAT main screen with an open model of a **Machining** process. OPCAT is capable of creating and editing of OPDs, Object-Process Language (OPL) sentences, and running simulations of the model.

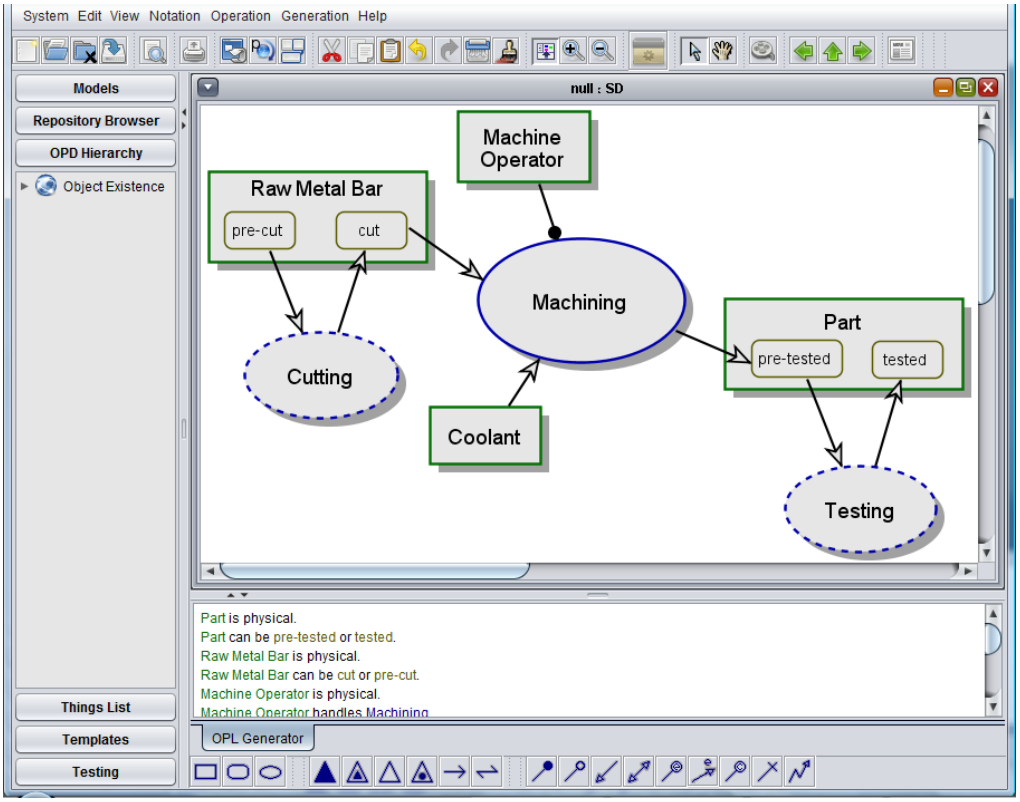

Figure 10: OPCAT modeling environment

3.2 Graphics-to-text model transformations with the Tesperanto editor

The Tesperanto editor is a graphics-to-text editor that supports Tesperanto methodology. The current version is a of Microsoft Word text editor that has been customized using Visual Studio Tools for Office (VSTO) and C#.

The decision to use MS-Word as a basis for the Tesperanto editor rather than continuing the development of a dedicated editor (as in earlier versions) was motivated by the benefit of adhering to current customary environments and tools. Imposing the least amount of changes on prospective users increases the likelihood of the tool's adoption. Wide spread of the tool is instrumental for achieving the overall goal of retaining technical documents humanly readable and writeable without compromising their formality and consistency.

The Tesperanto editor can be used for reviewing and improving an existing specification. This is achieved by converting the specification from its current text-based form to an OPM model. The source for a new Tesperanto document is this underlying OPM model, as the editor is capable of generating Tesperanto text automatically from the model. The Tesperanto editor can also serve for authoring a new bimodal (graphics-and-text) model-based specification.

If needed, Tesperanto text can be edited manually, and the graphical modality of the model will be updated accordingly. The same is also true in the reverse direction; when the graphical OPM model is edited, the Tesperanto text will be updated in response to reflect the changes in the model.

The example in Figure 11 shows an MS-Word window with Tesperanto text that was generated after synchronization with the model in Figure 8. Processes and objects are color emphasized. The internal entity numbering (shown as a subscript following the entity name)

helps to indicate multiple expressions that refer to the same entity. For example, "Welding$_1$" and "This Process$_1$" in Figure 11 are equivalent. Figure 12, which is the text of the Model-Based System Specifying process shown earlier in Figure 3, is another example for the use of such internal entity numbering.

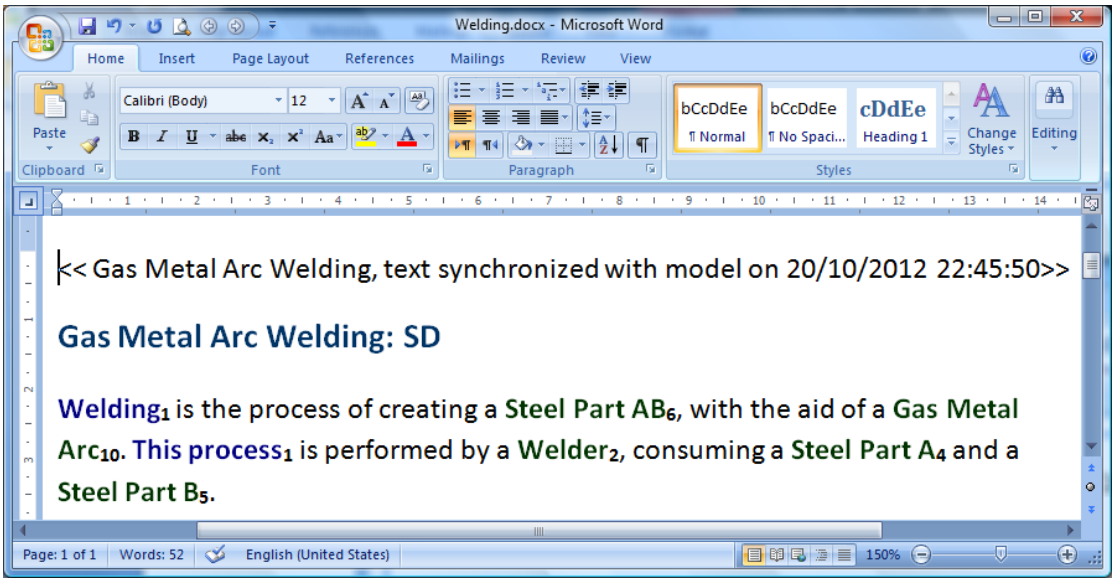

Figure 11: Gas Metal Arc Welding process Tesperanto text in the Tesperanto editor

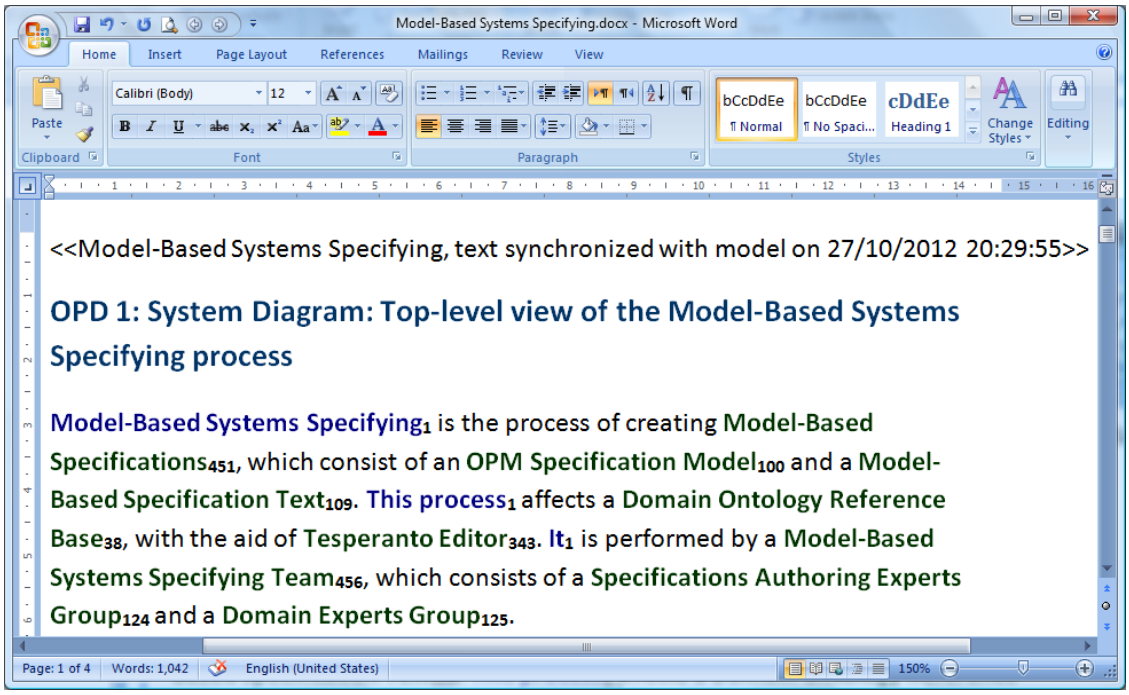

Figure 12: Tesperanto specification for Model-Based System Specifying process in the Tesperanto editor

The Tesperanto command screen, shown in Figure 13, contains three main modules: model-to-text (top), text-to-model (middle), and model matching (bottom). Features of the Tesperanto editor include extraction of structure and keywords from document tables of contents, indices and glossaries, pre-processing, and data import and export. A Natural Language Processing (NLP) layer includes sentence simplification, parts-of-speech tagging, and semantic similarity analysis. A pattern analysis module includes object-process-link heuristics and phrase repository, as well as text-to-model consistency checks. Advanced

editing tools on top of the MS-Word interface include syntax highlighting, phrase completion, smart tips, and snippets.

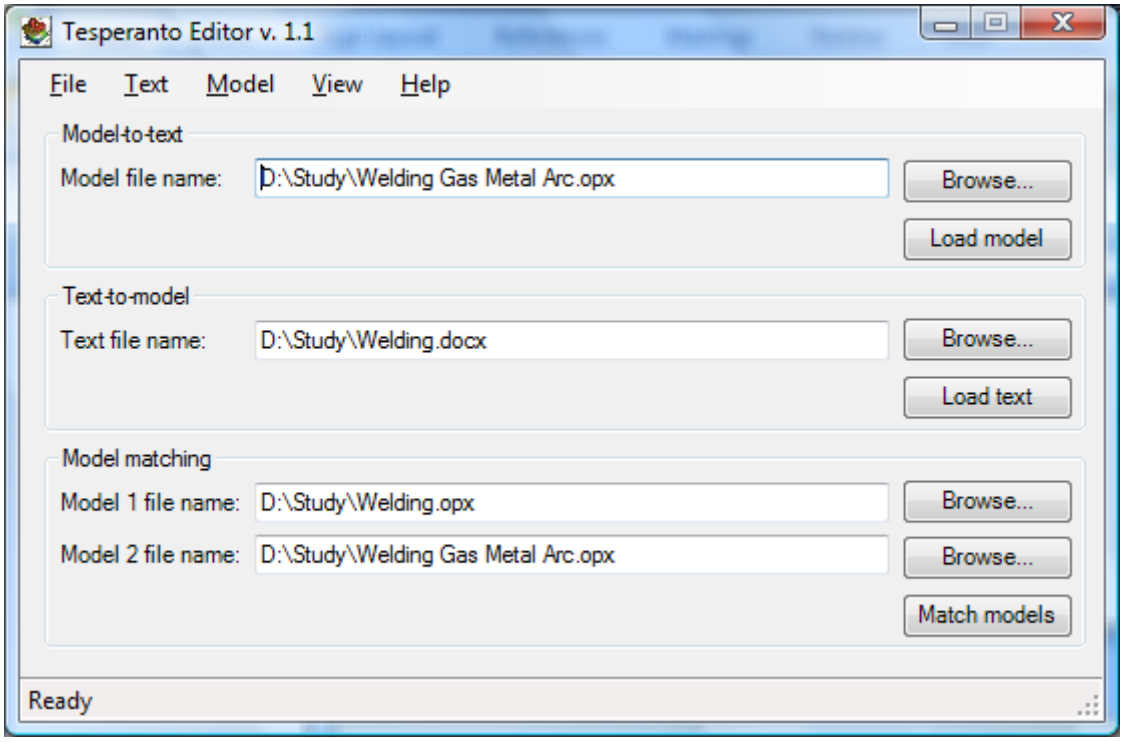

Figure 13: Tesperanto editor command screen

4. Evaluating Tesperanto for ISO standards

The assessment of the contribution of the developed Tesperanto method is a major methodological challenge. Partly, this is because it is hard to isolate the influence of various aspects of the Model-Based System Specification methodology, Tesperanto language, and the IDE in use.

Another part of the problem is that assessing Model-Based System Specification through an experiment involving human subjects requires reference to some source system or document specification. Under such conditions, parameter isolation is nearly impossible, as any significant experiment has to deal with a whole set of phenomena. Moreover, this can be a source of distortion and strong bias towards existing practices, as the participants are expected to exhibit tendency toward what they are familiar with, and therefore tend to reject new methods.

We believe that a thorough introduction to and acquaintance with the method is necessary both to achieve the desired results and be able to evaluate it critically and to suggest improvement directions. A case study is therefore the main evaluation avenue that is followed in this research. According to Yin (2003), this path allows for the best evaluation of Model-Based System Specification method, without compromise on the scientific exactness and generalization potential.

The work on the Model-Based System Specification methodology and the Tesperanto development environment has started following a presentation of model-based standard

authoring with OPM at the ISO TC184/SC5 annual meeting in April 2009 (Dori 2009). In Resolution 611 (Paris 21), ISO TC184/SC5 established Object Process Methodology Study Group (OPM SG) in order to explore the usefulness of Object Process Methodology for creating, designing, analyzing, and simulating models of its standards to improve the development, communication and understanding of these standards. OPM SG has been tasked with the goal of investigating the viability of using OPM as a methodology and modeling language for the purpose of streamlining, formalizing, and explicating the standard ontology and glossary, and making enterprise-related standards more comprehensive, accessible, usable, and consistent both internally and across standards (Dori 2009).

The common course of action for analyzing an ISO standard as a part of OPM SG mission is as follows.

1. Marking up objects, processes, and links in the document and creating an OPM model of the system specified in the document.
2. Analyzing and refining the model with respect to reference models in the same document or in other sources.
3. Automatically generating Tesperanto text.
4. Examining the Tesperanto text and adjusting it manually, if needed. The update of the corresponding OPM model is automatic.

A group of 27 experts from 12 countries, who had expressed interest in participating in this effort within the OPM SG, took part in online sessions and electronic exchange of documents and models over a period of more than two years. During this period, this methodology was applied to several manufacturing, control and enterprise standards, including IEC 62264, ISO 19440, and ISO/IEC 15416 (Dori, Martin & Blekhman, 2010).

Analyzing existing standards with this Model-Based System Specification methodology has exposed inconsistencies, mismatches, missing and confusing definitions, and contradictions between figures and text, as shown in (Dori et al. 2010). Some of the documents examined were approved technical standards that are in current use, emphasizing the importance of streamlining technical specifications in a consistent, *standard* manner.

Based on experience gathered from the work indicated above, the following conclusions were agreed upon and submitted to ISO TC184/SC5 plenary (Dori et al. 2010, Martin & Dori 2011). The reference throughout the evaluation report is to OPM, though the usage of OPM for analyzing standards followed the Model-Based System Specification methodology and Tesperanto as presented in this paper.

1. OPM offers a modeling methodology that can be applied to a wide variety of ISO TC184/SC5 and other ISO and IEC standards.
2. OPM advocates top-down refinement beginning at the System Diagram and aligns well with the way standards are structured beginning with a statement of scope.
3. OPM models of specific clauses assist in identifying inconsistencies within or between standards.
4. Using a modeling language in general and OPM in particular significantly improves the quality and the value of standards.

The following limitations of the proposed method were reported:

1. Learning OPM is not difficult: there is a minimal set of concepts and just one type of diagram that integrates function, structure, and behavior. However, the power of the

set of simple concepts means using them well in a wide variety of situations, and this is not a trivial task.

2. While OPM models of specific clauses may assist in identifying inconsistencies within or between standards, there seems to be little benefit from the extraordinary amount of effort required to produce OPM models for the entire standard.
3. Domain expertise is critical to the preparation of a satisfactory OPM model.
4. Domain-specific ontology and practice are implied qualities of a standard that cannot always be overcome without extensive revision.

5. *Conclusion and future work*

In this paper, we presented the Tesperanto methodology for model-based system specification and an evaluation of its application in the area of enterprise standards. The evaluation has demonstrated the value of this methodology as an aid for authoring technical specifications that are more consistent and more complete than those produced by a traditional text-centered approach.

Limitations of the proposed method include the extensive amount of work required for using existing tools in a realistic scenario and the need for domain-specific support.

Following these identified issues, we plan to continue the work at both the conceptual and the technical levels. Technically, we plan to enhance the Tesperanto editor with relevant commands and features based on its evaluation reports. At the conceptual level, we plan to incorporate domain ontologies into the tool.

In parallel, together with people from ISO TC/184 SC/5 WG/1 we have been developing OPM as an ISO Publically Available Specification (PAS), which is in its finalizing stages, and the ISO Draft International Standard for Model-Based Standards Authoring. We expect Tesperanto and its IDE implementation to play an important role in this emerging standard.

Acknowledgement

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 262044 - VISIONAIR.

References

Blekhman, A., Howes, D. B. and Dori, D. 2010. Model-Based Verification and Validation of a Manufacturing and Control Standard, *MSOM (Manufacturing and Service Operations Management Society) International Conference*, Haifa, Israel.

———, Dori, D. and Martin, R. 2011. Model-Based Standards Authoring. *21st INCOSE International Symposium*, Denver, CO, USA.

Dori, D. 2002. *Object-Process Methodology - A Holistic Systems Paradigm*. Berlin: Springer Verlag.

_____. 2004. 'ViSWeb – The Visual Semantic Web: Unifying Human and Machine Knowledge Representations with Object-Process Methodology', *The International Journal on Very Large Data Bases (VLDB)*, 13, 2, pp. 120-147.

———. 2009. Adopting Object-Process Methodology as a standard modeling language for modeling enterprise-related standards. *Plenary Meeting of ISO Technical Committee 184 WG5*, Paris.

———, Blekhman, A., Howes, D., and Martin, R. 2010. Object Process Methodology Study Group — Interim Report 2010. *Working document ISO/TC 184/SC 5 N1070*.

———, Martin, R., and Blekhman, A. 2010. Model-Based Meta-Standardization: Modeling Enterprise Standards with OPM. *IEEE International Systems Conference*, San Diego, CA, USA.

———. 2010. OPCAT – An Object-Process CASE Tool for OPM-Based Conceptual Modelling. *Design Society – First International Conference on Modelling and Management of Engineering Processes (MMEP 2010)*, Cambridge, United Kingdom.

Grobshtein Y. and Dori, D. 2011. Generating SysML Views from an OPM Model: Design and Evaluation. Systems Engineering, 14 (3), pp. 327-340.

INCOSE 2007. INCOSE Systems Engineering Vision 2020, INCOSE-TP-2004-004-02 Version/Revision: 2.03.

Martin, R., and Dori, D. 2011. Report on Activities of ISO/TC 184/SC 5/OPM Study Group Final Report. Working document ISO/TC 184/SC 5 N 1113.

MIT ESD 2012. The vision of the Massachusetts Institute of Technology Engineering Systems Division. Accessed 19/10/2012. http://esd.mit.edu/about/vmv.html.

Thalheim, B. 2011. The Theory of Conceptual Models, the Theory of Conceptual Modeling and Foundations of Conceptual Modeling. In *Handbook of Conceptual Modeling edited by D. W. Embley, B. Thalheim*. 543--577, Berlin, Springer.

Yin, R. K. 2003. 'Case Study Research: Design and Methods. *Third Edition. SAGE Publications.* California.

Biography

Alex Blekhman is a Ph. D. student at William Davidson Faculty of Industrial Engineering and Management at the Technion, Israel Institute of Technology, Haifa, Israel under the supervision of prof. Dov Dori.

Dov Dori is a professor in the William Davidson Faculty of Industrial Engineering and Management at the Technion, Israel Institute of Technology, Haifa, Israel, and a research affiliate at MIT, Cambridge, MA.