Is There a Complete Project Plan?
A Model-Based Project Planning Approach

Amira Sharon
Faculty of Industrial Engineering
and Management
Technion
Israel Institute of Technology
Haifa 32000, Israel
amirash@techunix.technion.ac.il
tel. +972503662466

Olivier L. de Weck
Engineering Systems Division
Massachusetts Institute of
Technology
77 Massachusetts Avenue,
Cambridge, MA 02139
deweck@mit.edu

Dov Dori
Technion, Israel Institute of
Technology
Haifa 32000, Israel, and
Engineering Systems Division
Massachusetts Institute of
Technology, Cambridge, MA
dori@mit.edu

Abstract

Project planning and control are the essentials of project management. The first steps of the project planning require some knowledge of the concept of the product that the project is expected to deliver. The initial project plan, scope of work, Work Breakdown Structure, and allocation of resources rely on understanding of at least the top-level product functionality, architecture, and concept of operation. The project planning and control is an iterative process of derivation, refinement, and simulation of the product model, while maintaining traceability and coherence between the product model and the project plan at all levels. Nevertheless, existing representations or views for project planning, such as Gantt, PERT, and DSM, contain only a portion of the entire set of relationships among project entities. This paper outlines a model-based approach to project planning, in which a joint project-product OPM model is the basis for the various project management views.

Introduction

Suppose you have been assigned to be the project manager of a new product development type project at Printers LTD. You will be leading the project of developing a new printer, which was identified as a need by the marketing department. As stated by that department's representatives, in addition to the standard printing functions, the printer is required to have two new capabilities: (1) wireless communication with a single computer or in a network, and (2) MS Vista OS compatibility. After some initial exploration, you determine the project scope and stakeholders views and you start thinking about the best way to devise the project plan, which will best serve as the basis for developing this wireless Vista-compatible printer. You know from your past experience that the initial project plan – the definition of tasks, their dependencies and duration, and the allocation of resources – rely on understanding of at least the top level product functionality, architecture, and concept of operation. The project planning and control is an iterative process of derivation, refinement and simulation of the product model, while thriving to maintain consistency and coherence between the product model and the project plan at all levels through the hierarchical structure of both the product and the project, as illustrated in Figure 1.

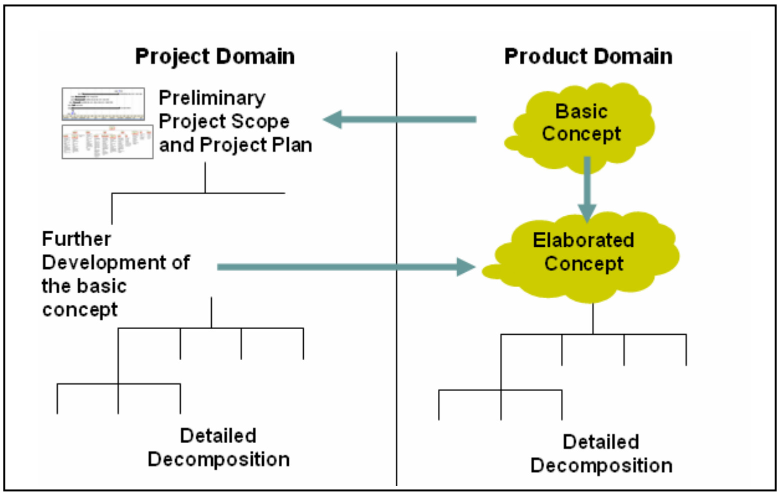

Figure 1. Building the project plan iteratively while creating the product model

With this realization in mind, you start by writing down the major tasks you think are necessary to deliver the printer, trying to determine what tasks must start before others, and what tasks must be completed in order for others to begin. You create the project Activities Network to depict the tasks, duration, and dependency information. The resulting network may have multiple parallel or interconnecting tasks. You plan scheduled milestones, checkpoints, or review points, where all tasks up to each one of those project control points must terminate at that review node.
As you are interested in viewing the information you have modeled in a manner that presents the duration, you will create and inspect the Gantt chart representation of the data. This will provide you with the information of what needs to be done and when. You define the top-level hammock tasks, followed by an iterative drill-down definition of the next level activities under each hammock. Usually, the last data line under each hammock is a milestone for indicating the end event of the specific hammock. Milestones are also used to indicate specific outcomes, associated with end of specific activity. The dependencies among the tasks are defined implicitly, based in the rationale of a "technological order" [Levy et. al. 1963] as understood from the product model.
Basically, the Gantt chart model contains activities (usually in a hierarchical manner), their duration, milestones (which are assigned zero duration), and dependencies among the activities. The dependencies are typically Finish-Start (FS), and each activity can start at its earliest start (ES). Task durations are deterministic, and no split of activities is modeled.
To gain better understanding of the human resource allocation, you use organizational charts and wish you had a way to automatically connect these chart nodes to specific tasks. Figure 2 presents a Gantt chart for our wireless printer. The first level hammock tasks are Scope, Requirements Analysis, Design Development, Testing, Documentation Preparation, Pilot, and Deployment. These top-level processes are typical and rather generic. The lower-level activities reflect increasingly deeper details and typify the specific product to be delivered by the project. A Gantt chart does not show the importance and inter-dependence of related parallel activities, neither does it show the prerequisite to complete one task before another can begin, as a critical path analysis does. To determine the critical path, you use the Critical Path Method (CPM), where you obtain, in addition to the tasks' duration and dependency information, also the slack time for each activity, and the critical path. This will provide you with the time required to complete the project, showing which activities are critical to maintaining the schedule and which are not. You know that a delay in the critical path causes a delay of the entire project, and if you are required to accelerate the project, it is necessary to reduce the total time required for the activities along the critical path. However, since the critical path of the activities does not account for the information flow of product parts or required artifacts as they are being designed, you will need to

use Design Structure Matrix (DSM, also known as N-square matrix) in order to analyze the interdependencies among the product components information and minimize the design process iterations based on information extracted from the DSM.
Regardless of the way you model and analyze the project plan, you must constantly and closely consult the product model, which is the ultimate project goal. Some of the task names contain product components names, some of the milestones represent completion of product-related issues, and the order of activities is the "technological order" in which the product evolves throughout the project activities. The product model is written all over the project plan, albeit not explicitly.

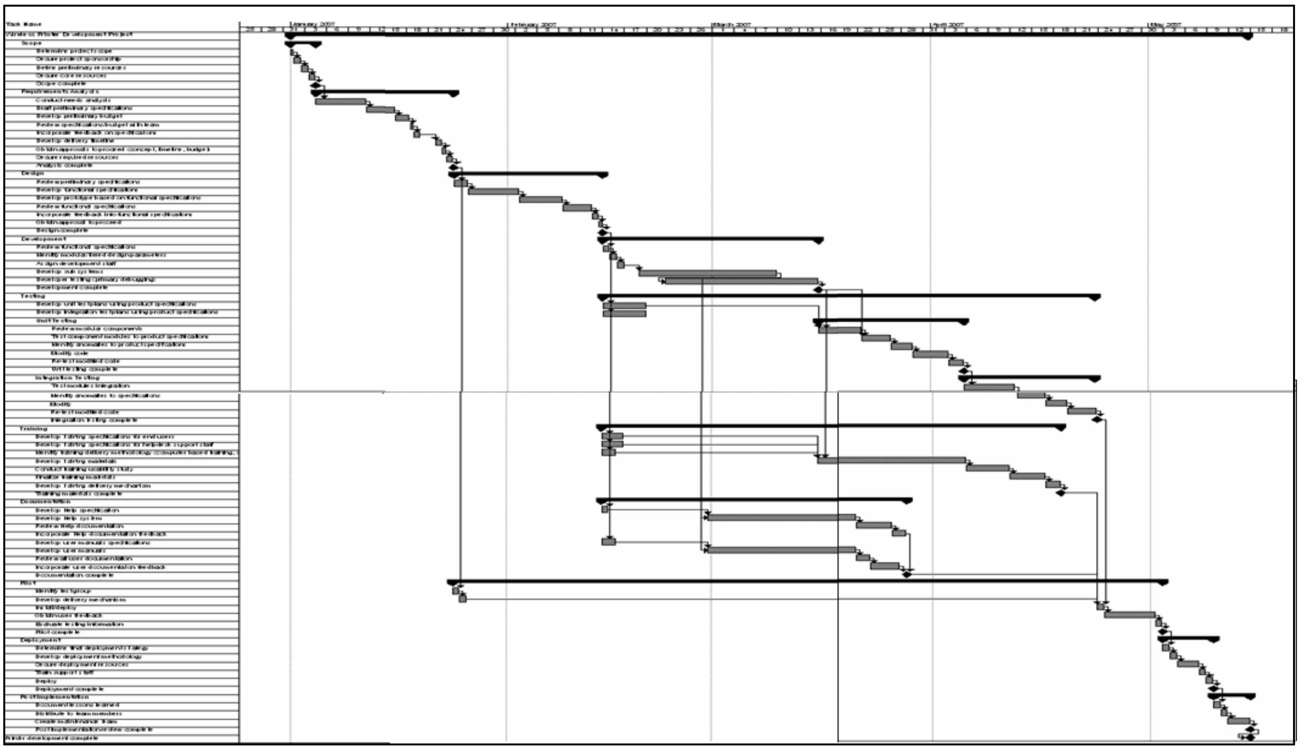

Figure 2. Wireless Printer project plan Gantt chart model

Having studied the different models and representation you had created in order to come up with a good project plan, you might realize that each of these is some derivative of a yet-to-be-determined overall project model, exposing a relatively narrow point of view of the project and even less of the product it is expected to ultimately deliver. How, then, is it possible to evaluate how "good" is your plan? How certain are you that the plan is complete and internally consistent? How can you ensure that the project plan is consistent with the product model to be designed?
A step towards enhancing your confidence in this project plan can be achieved if there was a way in which you could simultaneously model the project processes (activities and tasks), including their duration and dependencies, together with the project artifacts and their relationships to these processes, as well as resources allocation, in a single coherent model.

The Model-Based Project Plan Framework

Our Model-Based Project Planning (MBPP) approach calls for construction and utilization of a comprehensive product-integrated project model. MBPP is a derivative of the Project-Product Lifecycle Management (PPLM) approach [Sharon et. al. 2008, Perelman et. al. 2008], which integrates the project domain with the product domain via a shared ontology that explicitly relates project entities to product entities within a unifying frame of reference. PPLM is a combined research effort aimed at developing a methodology and a software environment for fusing the *product* to be developed with the *project* as they are executed by the *enterprise*. More specifically, the research concerns the development of an underlying holistic conceptual model, based on a shared ontology and supported by a software environment, for an integrated project and product lifecycle management. The shared ontological foundations facilitate associating concepts from the three domains, such that the resulting comprehensive model and supported software yield significant cuts in time-to-market, project risks, and product malfunctions.

To make the framework useful and beneficial, analysis mechanisms to evaluate the integrated model are defined, including the provision of execution capabilities for evaluation of project constraints and assertions related to the cost, time limits, resource allocation, and "what-if" scenario analyses.

A major challenge is to unify data from different sources through a systematic model-based approach. Hence, the supporting model notation for our MBPP framework must be both formal and intuitive. Desired characteristics of the modeling language include (1) ability to represent large amounts of data in simple, hierarchically organized diagrams, (2) expressiveness that allows definition of a common conceptual metamodel, and (3) formalism and clear semantics that can serve as a basis for model simulation and execution. Since the potential users of our contemplated framework would come from a wide range of disciplines and user profiles, such as project management and enterprise management, while formal, the notation must also be simple and intuitive so all the potential users and stakeholders can relate to the model as it evolves.

The candidate modeling languages were the same as for PPLM, namely UML, xUML, SysML, and OPM. UML and xUML are software oriented and hence less appropriate for modeling of general systems. SysML is designed for general systems and its notation modifies a subset of UML diagrams, while adding two new diagrams: Requirements Diagram and Parametric Diagram. Like UML, SysML handles complexity management primarily via aspect decomposition, while hierarchical breakdown is supported for a subset its diagrams. Unfortunately, the multiple-views model of SysML makes it difficult to comprehend the system as a whole. Unlike xUML, the notation does not use the formal activities specifications, and it allows free text for equation descriptions in the Parametric Diagram. This lack of formality hinders the definition of SysML model-based execution without significantly changing the notation. OPM notation supports conceptual modeling for general systems. Its top-down approach includes refinement mechanisms of in-zooming and unfolding. OPM [Dori. 2002] uses a single type of diagram to describe the functional, structural and behavioral aspects of the system. OPCAT [Dori et. al. 2003], an OPM-based conceptual modeling software environment features an accessible API, a basic animation module, and integration with files of various formats e.g., XML and CSV), reducing the development effort.

OPM is a holistic, integrated approach to the design and development of systems in general and complex dynamic systems in particular. OPM comprises entities and links. The three entity types are objects, processes (both referred to as "things"), and states. Objects are things that exist and can be stateful (i.e., have states). Processes transform objects: they generate and consume objects, or affect stateful objects by changing their states. Objects and processes are of equal importance, as they complement each other in the single-model specification of the system. Links, which are the OPM elements that connect entities, are of two types: structural and procedural. OPM objects relate statically to each other via structural relations, graphically expressed as structural links. The four fundamental structural relations are aggregation-participation, generalization-specialization, exhibition-characterization, and classification-instantiation. Objects can also be structurally related to each other by unidirectional or bidirectional tagged relations, similar to association links in UML class diagrams. Structural relations specify relations between any two objects. Due to the object-process symmetry, they can also specify relations between any two processes. Conversely, procedural links connect a process with an object or an object's state to specify the dynamics of the system. Procedural links include (1) transforming links: effect link, consumption link, result link, and the pair of input-output links, (2) enabling links, which are the agent and instrument links, and (3) control links: event, condition, invocation, and time exception links.

An OPM model consists of a set of hierarchically organized Object-Process Diagrams (OPDs) that alleviate systems' complexity. Each OPD is obtained by in-zooming or unfolding of a thing (object or process) in its ancestor OPD. One or more new things (objects and/or processes) can be specified within a thing in an OPD that was refined from a higher-level OPD. Copies of an existing thing can be placed in any diagram, where some or all the details, such as object states or links to other things, which are unimportant in the context of the diagram, can be hidden. It is sufficient for some detail to appear once in some OPD for it to be true for the system in general even though it is not shown in any other OPD.

The MBPP framework and platform is developed as an extension of Object Process Methodology (OPM), using the infrastructure provided by OPCAT and taking advantage of its following features:

- OPM is a visual methodology that incorporates the static-structural and dynamic-procedural aspects of a system into a unifying model, which is presented in its entirety using a single diagram type. This is achieved by treating both objects and processes as equally important things (entities). By using a single model at varying levels of detail, clutter and incompatibilities are likely to be avoided even in highly complex systems.
- OPM is designed to express triggering events, guarding conditions, timing constraints, timing exceptions, and flow-of-control constructs. These features are the basic elements required for exceptional behavior design, which is typical of real-life projects as they execute and therefore need to be simulated.
- OPM has proven to be an efficient methodology for modeling complex dynamic behaviors in general and temporal exceptions in particular. OPM was shown to be significantly better in specification quality, compared with OMT, UML's main predecessor [Peleg and Dori. 2000].
- Through its recursive seamless complexity management (scaling, or abstraction/refinement) mechanisms, OPM is highly appropriate for managing systems' complexities. There are three complexity management mechanisms in OPM: (1) unfolding/folding, which is used for refining/abstracting the structural hierarchy of a thing; (2) in-zooming/out-zooming, which exposes/hides the inner details of things within its frame; and (3) expressing/suppressing, which exposes/hides the states of an object. These complexity management mechanisms enable OPM to represent complex systems gradually.

OPM consists of two semantically equivalent modalities of the same model: graphical and textual. A set of interrelated Object-Process-Diagrams (OPDs) constitute the graphical model, and a set of automatically-generated sentences in a subset of English constitute the Object-Process Language (OPL). In the graphical-visual model, each OPD consists of OPM elements depicted as graphic symbols, while the OPD syntax specifies the consistent and correct ways by which those elements can be managed. Since the corresponding textual model is generated in a subset of English, it is immediately understood by domain experts, who need not learn any special language nor decipher cryptic code.

The Model-Based Project Plan

We now construct and examine the Model-Based Project Plan for the new wireless printer project, prepared in the Model-Based Project Plan Framework of OPM, using OPCAT. Figure 3 shows the OPM top level model for the project. The automatic Object Process Language (OPL), generated in OPCAT while creating the OPM model, is listed in Figure 4. The top level process at the first System Diagram (SD) of the model, **New Printer Project**, represents the entire project, which (like any project) is a process, and it is analogous to the first hammock activity in the Gantt chart, which encompasses all the project activities. The duration of this process is the duration of the entire project, and it is either determined a-priori or obtained by recursively aggregating the durations of all the subprocesses. The process is triggered by the **New Marketing Requirement** for a wireless printer, exhibited by the **Marketing Department**, which is a part of **Printers Ltd**. **Printers Ltd.** consists also of the **Project Team**, **Facilities Set**, and **Budget**. The process is handled by the **Project Team** (the agent) and consumes the **Budget** Resources, using the **Facilities Set** (the instrument). **Facilities Set** is instrument of the **New Printer** Project—it is required for executing the process but is not affected by it. The output of the process is the product – the **New Printer** in its final state which is **complete** – exhibiting the **Wireless Functionality** and the **Vista Compatibility**.

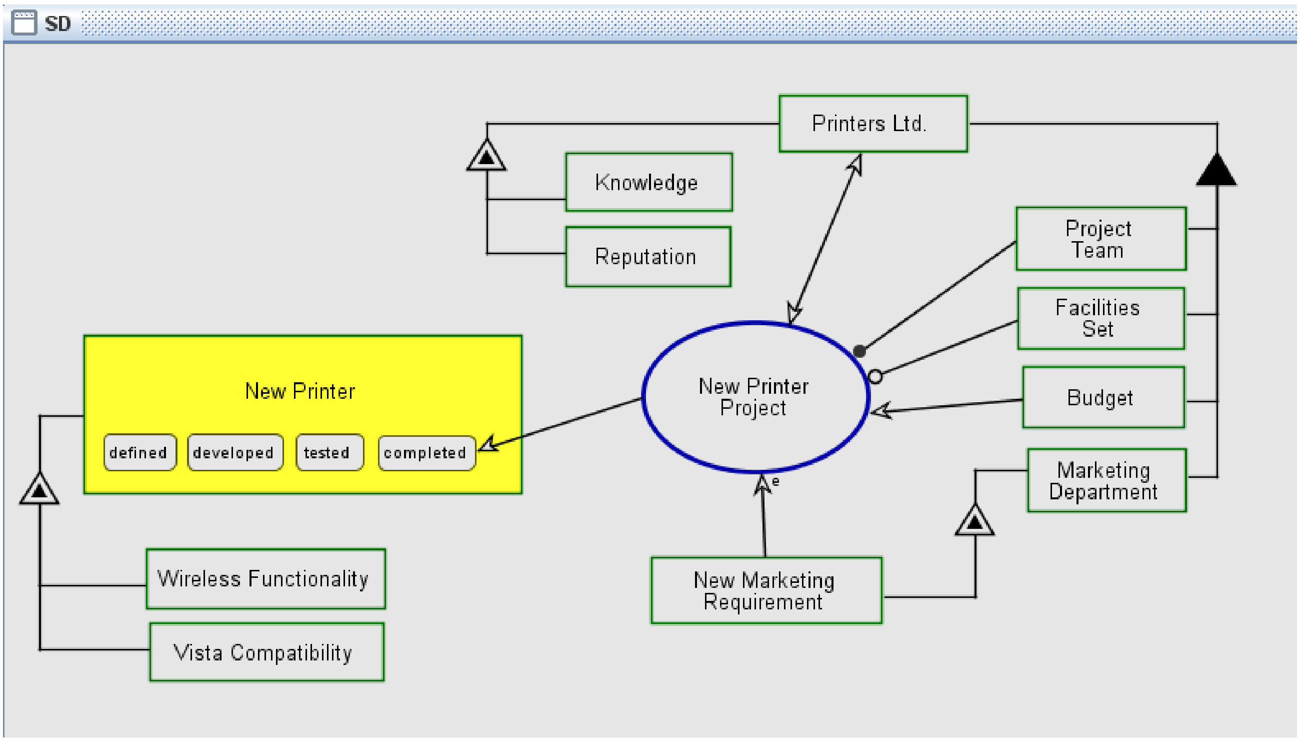

Figure 3. OPD of the top-level System Diagram (SD) of **New Printer Project**

Printers Ltd. exhibits Knowledge and Reputation.
Printers Ltd. consists of Project Team, Facilities Set, Marketing Department, and Budget.
Project Team handles New Printer Project.
Marketing Department exhibits New Marketing Requirement.
New Marketing Requirement triggers New Printer Project.
New Printer can be defined, developed, tested, or completed.
New Printer exhibits Wireless Functionality and Vista Compatibility.
New Printer Project requires Facilities Set.
New Printer Project affects Printers Ltd..
New Printer Project consumes Budget and New Marketing Requirement.
New Printer Project yields completed New Printer.

Figure 4. The automatically-generated top-level OPL paragraph of the OPD in Figure 3

Zooming into the project process **New Printer Project** (Figure 5) reveals the next-level processes, which are identical to the hammocks of the Gantt chart for the wireless printer, shown in Figure 2: Scope Defining, Requirements Analysis, Design Development, Testing, Documentation Preparation, Pilot, and Deployment, only this time these processes are presented along with the objects flow among them, explicitly addressing the processes order rationale and required outcomes design. Each one of these sub-processes can be further decomposed, using the in-zooming mechanism, maintaining downwards and upwards consistency at each level of decomposition. The automatically generated Object Process Language (OPL) text is listed in Figure 6.

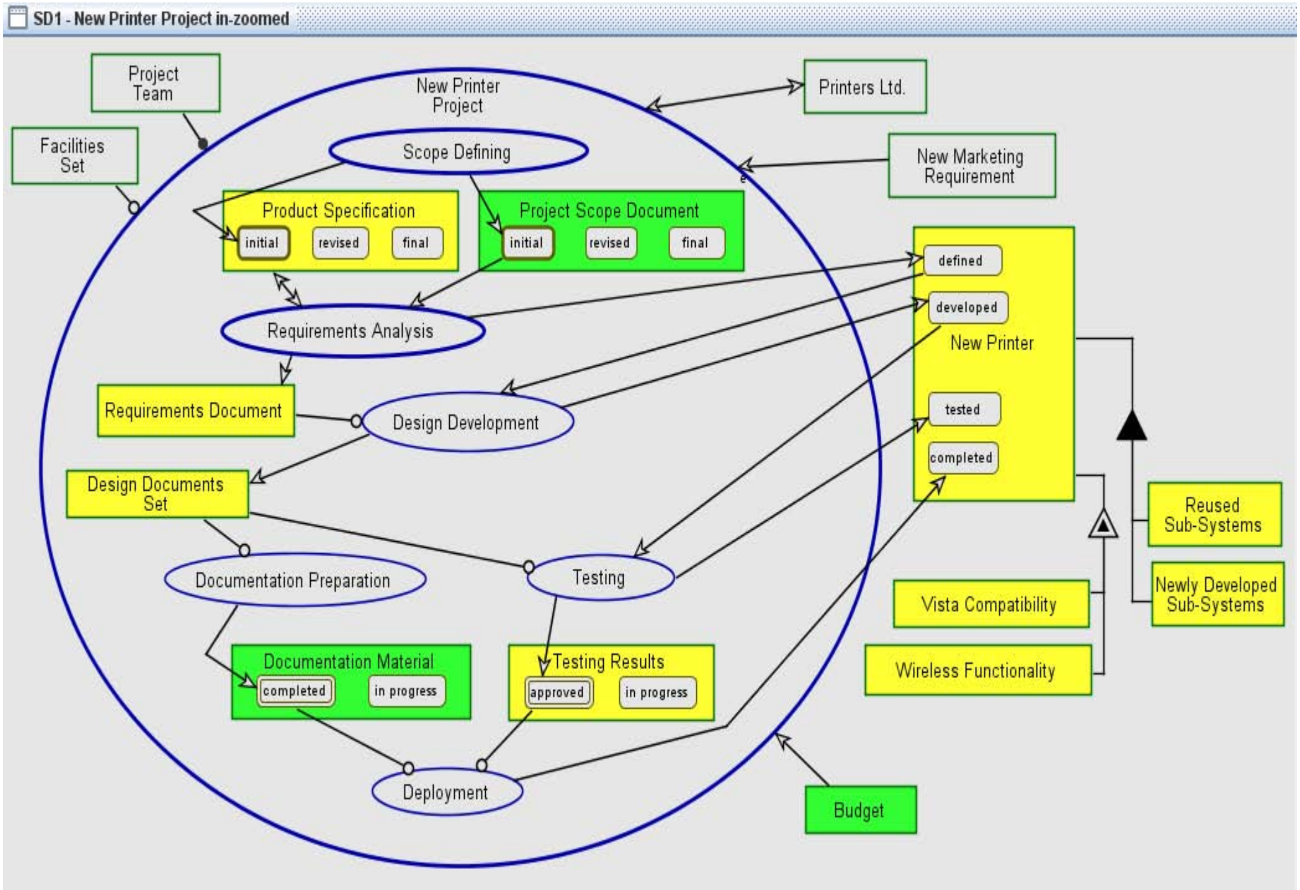

Figure 5. OPD of **New Printer Project** in-zoomed

The **Scope Defining** process accomplishes two outputs: The **Project Scope Document** and the **Product Specification**, both in their **initial** state. These two outputs are required for the **Requirements Analysis** process, which also affects **Product Specification**, and it accomplishes the **Requirements Document**. The **Requirements Analysis** process yields the **New Printer** at its **defined** state. This **New Printer**'s state together with the **Requirements Document**, are required for the **Design Development** process to begin. The **Design Development** process changes the **New Printer**'s state from state **defined** to state **developed**. It accomplishes the **Design Document Set** required for the **Documentation Preparation** process and **Testing** process, both of which enabled simultaneously. The **Testing** process yields **Testing Results** at their **approved** final state, while the **Documentation Preparation** process yields the **Documentation Material** at its **completed** final state. The **Deployment** process requires the completed **Documentation Material** and the approved **Testing Results**. The **Deployment** process yields the completed **New Printer**, which is constructed from **Reused Sub-Systems Set**, as well as **Newly Developed Sub-Systems Set**.

Project Team handles New Printer Project.
New Marketing Requirement triggers New Printer Project.
New Printer can be defined, developed, tested, or completed.
New Printer exhibits Wireless Functionality and Vista Compatibility.
New Printer consists of Reused Sub-Systems and Newly Developed Sub-Systems.
New Printer Project requires Facilities Set.
New Printer Project affects Printers Ltd.
New Printer Project consumes Budget and New Marketing Requirement.
New Printer Project zooms into Scope Defining, Requirements Analysis, Design Development, Documentation Preparation, Testing, and Deployment, as well as Documentation Material, Testing Results, Requirements Document, Design Documents Set, Product Specification, and Project Scope Document.

Documentation Material can be completed or in progress.
completed is final.
Testing Results can be approved or in progress.
approved is final.
Product Specification can be initial, revised, or final.
initial is initial.
Project Scope Document can be initial, revised, or final.
initial is initial.
Scope Defining yields initial Project Scope Document and initial Product Specification.
Requirements Analysis affects Product Specification.
Requirements Analysis consumes initial Project Scope Document.
Requirements Analysis yields Requirements Document and defined New Printer.
Design Development requires Requirements Document.
Design Development changes New Printer from defined to developed.
Design Development yields Design Documents Set.
Documentation Preparation requires Design Documents Set.
Documentation Preparation yields completed Documentation Material.
Testing requires Design Documents Set.
Testing changes New Printer from developed to tested.
Testing yields approved Testing Results.
Deployment requires approved Testing Results and completed Documentation Material.
Deployment yields completed New Printer.

Figure 6. The automatically-generated OPL paragraph of the OPD in Figure 5

Modeling each one of the **New Printer Project** subprocesses further is conducted in the same manner. In-zooming into the **Scope Defining** process (Figure 7) shows again the required two basic essential inputs – the **Budget** resources (consumed by the process) as well as the **Project Team** (to handle the process). Examining the objects in Figure 7, we can clearly see the things that reside in the product domain, which are colored in yellow: The objects—**Product Specification**, and **New Printer**—the final product, as well as specific attributes of the **Selected Alternative.** Things in the project domain, identified by their green color, include **Project Scope Document**, the **Cost** attribute of each **Alternative** under the **Alternatives Set**, and the **Cost** and **Schedule** attributes of the **Selected Alternative**. The things that are common to both the project and the product domains are colored in light blue (which is the color obtained by mixing green with yellow): The **Alternatives Set**, which obviously contains both product- and project-related data and issues, including the associated **Risk** for each **Alternative**, which is also constructed based on data from both domains. The automatically generated Object Process Language (OPL) text is listed in Figure 8.

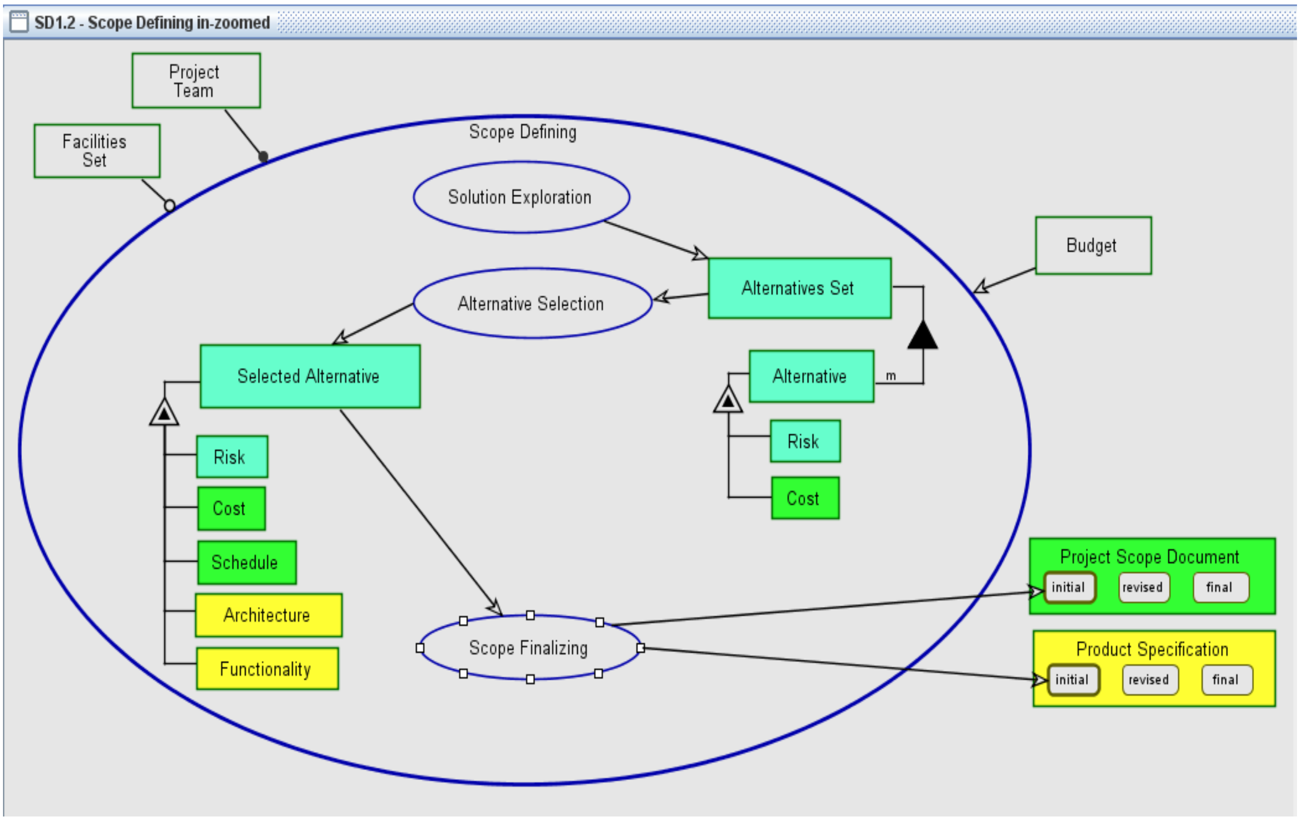

Figure 7. OPD of **Scope Defining** in-zoomed

Project Team handles Scope Defining.
Project Scope Document can be initial, revised, or final.
initial is initial.
Product Specification can be initial, revised, or final.
initial is initial.
Scope Defining requires Facilities Set.
Scope Defining consumes Budget.
Scope Defining zooms into Solution Exploration, Alternative Selection, and Scope Finalizing, as well as Selected Alternative, Schedule, Cost, Risk, Cost, Risk, Alternatives Set, Alternative, Functionality, and Architecture.
Selected Alternative exhibits Risk, Cost, Schedule, Architecture, and Functionality.
Alternatives Set consists of many Alternatives.
Alternative exhibits Risk and Cost.
Solution Exploration yields Alternatives Set.
Alternative Selection consumes Alternatives Set.
Alternative Selection yields Selected Alternative.
Scope Finalizing consumes Selected Alternative.
Scope Finalizing yields initial Product Specification and initial Project Scope Document.

Figure 8. The automatically generated OPL paragraph of the **Scope Defining** process

The entire project is further modeled through concurrent hierarchical decomposition of processes and objects. Using this approach the model ultimately contains the activities and tasks (which are simply processes at various detail levels) required for completing the product, the artifacts (mainly model-based documents) created during the project process, and the different resources, both agents and budget. Having all this information embedded consistently in the same model eventually yields all the specific structural relations (among objects) and procedural relations (between objects and processes). The project planner can model the rationale for the processes order by identifying the objects flow in and out of each process. Understanding the objects hierarchy hand-in-hand with the project progress is the rationale for the technological process order and timing.

Another advantage of this model-based project planning approach is the capability of zigzagging between the product domain and the project domain in the same model. In fact, the planner must perform this back-and-forth thinking process while planning in order to come up with a project plan that best addresses the product to be accomplished by the project.

The ability to simultaneously express the required information from both domains within a single integrated model-based framework can potentially lead to a more reliable project plan, which is less prone to the need for repeated changes. The increased robustness of the resulting project plan results from having to make the decisions about the project's process order and logic while explicitly addressing the associated product model. The planning process carried out following this approach clarifies the intricate relationships between the project and product entities. This model-based approach enables the simultaneous expression of the function, structure and behavior of both the project and the product via the same ontological and methodological foundations, maintaining full traceability between the project and product data.

The Generic Project Construct

Investigating the complete project model throughout its levels reveals a basic Generic Project Construct (GPC), which is presented in Figure 9. This generic construct contains one *Process* representing a single activity at any level, with an associated *Duration*, related to at least three generic object-sets (one or many): **Budget** (the input), **Human Resources Set** (the agent), **Facilities Set** (the instrument) and **Deliverables Set** (the output). The logic behind this generic construct is that a process is aimed at achieving a specific **Deliverables Set** and it can be accomplished by the **Human Resources Set**, **Budget**, and **Facilities Set**. The **Deliverables Set** may include artifact such as documents, approvals, simulations, analysis, specifications, and reports. Each one of these artifacts results explicitly from a specific process and used (usually as instruments, since they are informatical objects and hence are not consumed) in a subsequent process.

Using the GPC facilitates complete modeling of the entire project, as it provides a means for explicitly addressing the artifacts that are generated and required along the project process, together with the time aspects for these artifacts, as derived from the relevant processes duration. The resulting project plan integrally incorporates the product to be designed, manufactured, delivered, used, serviced, and disposed of, with the project, including all its significant artifacts, such as resources and their allocation, timetable, limits and milestones, evaluation of project constraints, and assertions related to its cost and duration. Since all the entities and their relations are represented in the model, automatic procedures can be devised to generate other project view representations, such as Activities Network Plan, Gantt chart, PERT, Organizational charts, DSM, and CPM analysis.

The Activities Network Plan view is generated based on the dependencies between processes in the OPM model-based Project Plan. Each process turns into an activity node of the network, and all its predecessors are identified and automatically converted into adequately linked nodes while maintaining their names. Since the OPM Project Plan is hierarchical, the user can choose the level of Activities Network Plan she whishes to look at. The coherence of the network is valuable, since although an

Activities chart might be created for the entire project, size limitations of real-life projects require breaking the top-level chart into smaller, manageable parts, which are not necessarily coherent with each other, or with the entire project network. The automatically-generated nested Activities Network Chart maintains the required coherence and enables zooming in or out as necessary, in order to gain full comprehension of the network.

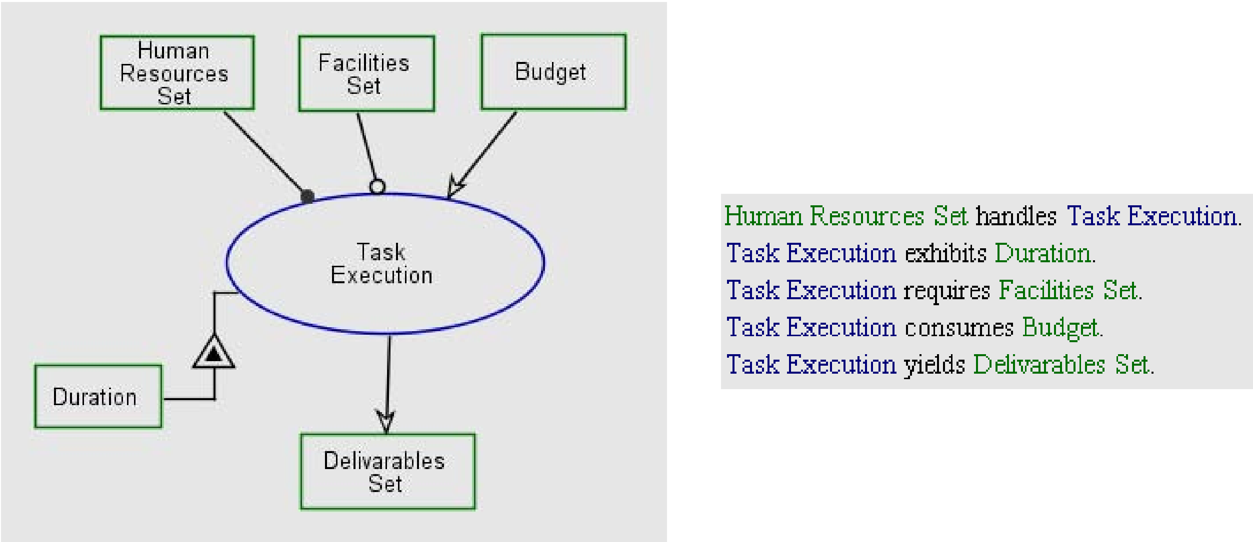

Figure 9. The Generic Project Construct (GPC) of the Model-Based Project Plan

The classical Activities Network contains all the processes without dataflow, but the automatic extraction from the OPM model-based Project Plan enables viewing an Enhanced Activities Network Plan, in which the processes and the objects are simultaneously displayed, along with their input, output, agent, and instrument relationships. A simple mechanism can show or hide the dataflow of objects among the processes, at each level of detail in the Enhanced Activities Network (EAN). Assigning probabilistic durations to the processes in the OPM model-based project plan, followed by applying the OPCAT simulation mechanism to the project model will provide statistical estimations of the project duration processes and their corresponding budget allocations.

If Early Start (ES), Late Start (LS), Early Finish (EF), and Late Finish (LF) constraints have been determined for the processes in the OPM model-based Project Plan, then the critical path can be obtained in the OPM model. Furthermore, this information can be extracted from the OPM model, copying the data to the corresponding activity nodes, and presented in its common CPM view. Like the Activities Network, the classical CPM view lacks dataflow. Automatic extraction of the CPM view from the OPM model-based Project Plan enables the view of an Enhanced Critical Path (ECP) showing the processes and objects simultaneously. This enables the project manager to detect not just the critical path, but also the critical artifacts (human resources, budget…) and risks—those that are related to the processes along the critical path—and monitor them closely. Knowing what these artifacts and risks are, enabling to focus on managing them with high priority, adds value to the project management process. Rather than inquiring about the processes designed to achieve and produce these different artifacts, the project manager can monitor the status of the different artifacts themselves, including documents, approvals, simulation outcomes, analyses, specifications, and reports. This, in turn provides the project manager with the flexibility of treating the processes not as end in itself but rather as a means, vehicles for obtaining the required artifacts and optimizing the processes accordingly to *achieve the critical **artifacts** rather than complete the critical **tasks***. In other words, the ability to describe the critical path in terms of products (artifacts) rather than the processes that yield them, can be extremely valuable to the project manager, since one can easily spend the budget and time on completing less critical processes, or "spin the wheels in neutral" on work activities, and produce outputs that do not advance the project goals and deliverables.

The Gantt chart is the view that is simplest to automatically generate from the OPM model-based Project Plan, since its structure contains all the processes in their exact hierarchical structure. Some project planning software packages (e.g., MS-Project) treat a milestone the same as a process (activity) only with zero length. However, semantically, a milestone is an event—a point in time at which some important happening occurs in the project system. To remedy this, a special transformation rule has been devised within the model-based OPM project plan, for the last process at each level, along with its related objects, enabling the smooth transformation of theses final constructs in the OPM project plan to corresponding milestones in the Gantt chart, and visa versa.

The three different types of DSM presented in Table 1 can also be automatically generated from the OPM model. The Process-based DSM is obtained from the relationships defined in the OPM model. Examining the generated DSM, identifying blocks of coupled activities, iterations and re-sequencing product development tasks to minimize budget and duration can be integrated into OPM. Since the OPM model includes the project and product objects in addition to the processes, the automatic generation of the component-based DSM can also be extracted based on the relationships established in the OPM product model. As a result of the inclusion of all agent human resources in the OPM model-based project plan, it is also possible to automatically produce the Organizational-based DSM or social network, as revealed from the assignment of the humans in the model.

Table 1. Different types of Dependency Structure Matrix (DSM)

DSM Data Types	Representation	Application	Analysis Method
Process-based (Task-based, Activity-based)	Activity input/output relationships	Project planning	Sequencing & Partitioning
Organizational-based	Multi-team interface characteristics	Organizational design, interface management, team integration	Clustering
Component-based	Multi-component relationships	System architecting, engineering and design	Clustering

Summary and Future Research

The model-based project management (MBPM) paradigm proposed in this paper can be considered a subset of model-based systems engineering (MBSE). Indeed, some general benefits of MBSE are especially notable and valuable in MBPN. These include the ability to gain deep comprehension of the project-product super-system and to derive the various project management tools and views from the unifying project-product model. We have focused on the model-based construction of the project plan. The Model-Based Project Plan (MBPP) is the basis for MBPM. Starting with construction of an OPM model of the project and the product it is expected to deliver, we obtain a comprehensive model that serves as a basis for deriving the customary repertoire of project management tools. However, rather than being constructed separately and independently, which is a certain source of mismatches and incompatibilities with potential detrimental consequences to the project success, these project views are merely reflections of various aspects of a comprehensive underlying model and are therefore consistent and coherent. Any change in the project model is automatically reflected in the various project management tools—Activities Network, CPM, DSM, WBS, and Gantt. Moreover, any change in the product to be delivered—the ultimate output of the project—can be modeled in the joint project-product model and its implications on the project parameters can be directly inspected and assessed.

The fact that the model integrates the process view of project activities and tasks with the object view of the product deliverables with its components at all levels and their associated artifacts, enables the project manager to focus on advancing the completion of the output deliverables rather than on just performing processes without direct reference to their anticipated outcomes. Knowing the critical path and the critical artifacts—those generated by processes along the critical path—enables to closely manage and monitor them. Future work includes the following research items:

- Specifying **algorithms for creating the various project management tools** from the OPM model and automatically updating them in response to changes in the project plan or product specification.
- Enabling real-time instance and data-based **execution of the project model** for simulation, exploration, and forecasting purposes. Using the enhanced execution capabilities of OPCAT, it will be possible to simulate and adapt the project and product parameters for feasibility or even optimality of the project's designed timeline, resource allocation, and tradeoff of product functionality and deliverables.
- Developing mechanisms for effectively **assessing and managing risks** that relate to processes and their resulting objects along the critical path by monitoring them more closely than other, less critical risks and mitigating them by allocating more resources to reduce the probability of their occurrence.
- Concurrently monitoring and controlling both the project and product evolution, the product manager will be able to **take corrective actions** in real time as needed if potential deviations are detected via the execution-based simulation or if actual deviations are detected via close monitoring of the interacting project-product lifecycles within the enterprise or in relation to subcontractors' deviations from their schedules, and
- Developing mechanisms for answering "what-if" questions that enable executives to **assess the impact of changes** requested by the customer or are about to be introduced to the product during its design due to technological, legislative, or economical considerations, in terms of impacting cost, risk, and time-to-market. What-if analysis based on possible architecture models can be conducted to help achieve objective and rationalized decisions regarding the appropriate architecture from the set of potential candidates. Such analysis can reveal contradicting requirements and find out corrective actions in order to improve the analyzed architecture. Finally, the system model analysis reduces the risk of developing a system with architecture that does not meet the requirements and the technical, legal, and/or environmental constraints.

References

Sharon, A., Perelman V., and Dori, D. 2008. A Project-Product Lifecycle Management Approach For Improved Systems Engineering Practices. In Proceedings of the Eighteenth Annual International Symposium of the International Council on Systems Engineering (Utrecht): INCOSE.

Perelman, V., Sharon, A., and Dori, D. 2008. Towards a Unified Product and Project Lifecycle Model (PPLM) for Systems Engineering. In Proceedings of ESDA 2008 – 9th ASME Engineering Systems Design and Analysis Conference, Haifa, Israel, July 7-9, 2008.

Dori, D. 2002. Object-Process Methodology – A Holistic Systems Paradigm, Springer Verlag, Berlin, Heidelberg, New York.

Dori, D. Reinhartz-Berger, I., and Sturm, A. 2003. Developing Complex Systems with Object-Process Methodology using OPCAT. Conceptual Modeling – ER 2003. Lecture Notes in

Computer Science (2813), pp. 570-572, 2003.

Levy, K.F., Thompson, G. L and Wiest, J.D. 1963. The ABCs of the Critical Path Method, Harvard Business Review, #63508, Harvard Business School Publishing.

Peleg, M. and Dori, D. 2000. The Model Multiplicity Problem: Experimenting with Real-Time Specification Methods," IEEE Transactions on Software Engineering, vol. 26, pp. 742-759.

Biographies

Amira Sharon is Senior Systems Engineer at IAI – Israel Aerospace Industries' ELTA Division and a Ph.D. candidate of Information Management Engineering at the Technion. She holds a B.Sc. in Mechanical Engineering (1992) and M.Sc. in Industrial Design (1997) from Technion. Her professional career spans the areas of programming, systems engineering, and technical management of large scale projects. During the past seven years, she has been leading systems and software engineering methodologies development and implementation across IAI.

Olivier L. de Weck is Associate Professor of Aeronautics and Astronautics and Engineering. He holds an industrial engineering degree from ETH Zurich (1993) and Ph.D. in aerospace systems engineering from MIT (2001). Before joining MIT he was a liaison engineer and later engineering program manager on the F/A-18 aircraft program at McDonnell Douglas (1993-1997). His research interests, teaching emphasis and professional experience is mainly in two areas: Systems Engineering for Changeability and Commonality and Space Exploration Logistics. Prof. de Weck is an Associate Fellow of AIAA, winner of the 2006 Frank E. Perkins award for excellence in graduate advising and recipient of the 2007 AIAA MDO TC outstanding service award. He won two best paper awards at the 2004 INCOSE Systems Engineering Conference, held the Robert Noyce Career Development Professorship from 2002-2005, and co-advised the best MIT System Design and Management thesis in 2005. He has over 100 journal and conference publications in the area of systems engineering and space systems design for exploration and communications. His research has been funded by GM, NASA, BP, JPL, Arvin Meritor, DARPA/AFRL and the Alfred P. Sloan Foundation. Prof. de Weck is a member of the International Council on Systems Engineering (INCOSE) and the American Society of Engineering Education (ASEE). He served as the General Chair for the 2nd AIAA MDO Specialist Conference in May 2006. He serves as a faculty mentor to a number of student teams.

Dov Dori is Visiting Professor at MIT's Engineering Systems Division (ESD). Between 2001 and 2008 he was Head of Technion's Area of Information Systems Engineering at the Faculty of Industrial Engineering and Management, and Research Affiliate at MIT. Between 1999 and 2001 he was Visiting Faculty at MIT's Sloan and ESD. Professor Dori received his B.Sc. in Industrial Engineering and Management from the Technion in 1975, M.Sc. in Operations Research from Tel Aviv University in 1981, and Ph.D. in Computer Science from Weizmann Institute of Science, Israel, in 1988. Between 1978 and 1984 he was Chief Industrial Engineer of the MERKAVA Tank Production Plant. His research interests include Model-Based Systems Engineering, Systems Development and Lifecycle Methodologies, and Information Systems Engineering. Between 1999 and 2001 he was Associate Editor of IEEE T-PAMI and is currently Associate Editor of Systems Engineering. He is author/co-editor of four books and author of over 150 publications. Prof. Dori is Fellow of the International Association for Pattern Recognition (IAPR), Senior Member of IEEE and ACM, and member of INCOSE.