Effective use of MBSE

Sharon Shoshany Tavory, Technion IL.

1 Abstract

Model based systems engineering, had become, in recent years, the current buzz word in the systems engineering (SE) domain. While model based development is well established in specific engineering disciplines (electronics, mechanics, software), its role in SE is still evolving.

While models were used in SE from day one, their use was limited to confined development stages or deliveries. The latest interpretation of MBSE uses the same model over several stages of systems development and as a major artifact of the process.

The literature is filling-up with case studies and success stories, various vendors promise us full life cycle support yet the actual reality in project work seems to be different.

To define and analyze the gap, a research was set up surveying a range of organizations, projects and MBSE methods, in the Israeli industry. The idea being that by surveying a group of projects, we can find the factors holding MBSE from delivering its promises, and design ways to make its use more common and effective.

The paper covers the following topics

- Key performance parameters influencing the successful use of MBSE, as elicited from a literature review and as extracted from the surveyed projects
- Types of MBSE uses encountered in the local industry
- The primary findings of the research
- Suggested remedies

2 Introduction

The traditional SE process is textual by nature, where documents are being used at each phase as deliveries, and systems engineers translate or transform requirements from the user textual description, through the different stages of system analysis and design. The specialty engineers translate it yet again to their domain. Textual definitions tend to be inexact and very domain sensitive. The process is therefore over sensitive to interpretation and terminology issues. In large projects the problem is worsened by the volume of requirements and design items, and managing textual consistency turns into a nightmare. The discrepancies generated by such a process appear at the advanced stages of development – integration and testing where changes are costly and time consuming.

The traditional process was enhanced and annotated by models and simulations – used for specific problem solving. Examples of such uses are models used for operational requirements (OR), models used for component development and models for optimization of specific aspects of a system.

Current model based systems engineering tries to take modeling one step further and create a fully model based process. As phrased in [1] "elevating models in the engineering process to a central and governing role in the specification, design, integration, validation, and operation of a system". The vision is carried even further by vendors offering code generation (for software and hardware languages) hence adding implementation to the roles and making it model based engineering.

By doing so the SE community is trying to solve the following process problems:

- Communication and understandability – well structured models improves the ability to convey meaning to different stakeholders.
- Traceability – linked and repository based models allow for better traceability and consistent models.
- Early knowledge – early executable models allows eliciting knowledge earlier.
- Reduce time to market (TTM) – model based analysis and design takes less time than the textual process. Models that create deliveries automatically, improve TTM.
- Reuse – well structured model part can be reused in product lines or component based development, again shortening the development cycle and cost.
- Formal proofs – models can be validated early and fully, models that are turned into code are considered proof by construct. It is useful where system high reliability is required.
- Maintenance – a model of the system captures all the data needed for change thus allowing for easier maintenance.

The vision is accompanied by methods and tools and should be accompanied by a defined process.

This research set out to look at the methods and tools in their organization environment application to see whether the vision is becoming reality.

The underlying assumption seems to be that if a well defined method and good tool is offered, the SE community will turn MBSE[1] into a major contributor to the SE process. Sources such as [1] show that methods seem to be around and supporting tools are offered. Case studies such as [3, 4, 5, 6, 7] show that projects can turn it into success. Yet it seems that at the overall organization level MBSE is not embraced like it should [8]. The reasons might be

- It is just the start of deployment and it would soon turn into a larger impact (too early to tell, the research is premature).
- The methods and tools are not supporting enough to allow an effective MBSE process (technology problem)

[1] [1] The acronym MBSE will be used throughout the paper. In the literature, various other similar meaning acronyms are used - MBD (model based design) MBE (model based engineering) and other variations. More about it in Appendix A.

- The users have various issues that stop them from using it effectively (people/organization problem).
- There is no problem with the process or tools but the SE practitioners are resisting change (it was explored in the past as one of the key problems in SE community, but since then, most of organizations went through CMMI or CMMI like process improvement phases and are supposed to be in a "change enable" state of mind)

The goal of the research was to explore the state of affairs in the local industry and try to answer the following top level questions:

- What is the actual reality
- If there is a gap (like it seems), what explains the gap
- How can we do better – use MBSE more and more effectively
- When should we use MBSE

The following parts will outline the research, explore the key performance parameters elicited during it, and describe the primary findings.

3 Research method, process and logic

The selected research method was qualitative – since the problem seems to lie both in the technology domain and the social domain, and the sampling space was too sparse (few projects, many methods, many tools), applying this method to the problem seemed appropriate.

The research stages comprised of

1. Literature review (see appendix A)
2. Key performance parameters elicitation
3. Questionnaire developing (see appendix B)
4. Participant projects selection
5. Interviews and data gathering
6. Analysis and feedback from interviewees.

As speculated before, the problem might be technical or social, therefore the literature survey covered both the technology literature and the organization sociology one. Since using MBSE influences methods and tools, but might also influence process, all these domains were covered. It resulted in the survey covering MBSE case studies, exploration of technology transfer domain (specifically of tools and SW) and change management domain (specifically of process change).

From the literature survey a list of candidate key performance parameters, influencing the probable effective incorporation of MBSE was elicited. A questionnaire was developed based on these parameters. The questionnaire explicitly explored the conformance between reported success and the elicited parameters.

In the next stage, projects were selected. While not being a qualitative research, the participating projects were selected so to cover the extrema points of the assumed differentiators. The differentiators were selected so to eliminate the effect of specific organization/industry/project culture and way of conduct, and as a result make the conclusions more generally applicable. The following differentiators[2] were addressed:

- Different organizations – a single organization might have a specific process/method it might adhere to, or a local SE culture
- Different type of industry – the research tried to capture both defense industry (that might have its own culture) and non-defense, also governmentally owned and private.
- Different size – both of project and of organization
- Different modeling methods and tools – to eliminate the effect of a single method/tool maturity and performance on the research, multiple methods were selected. Of which the selected ones were uses of the model at system level, describing multiple aspects of the system and used for various SE phases. The final list of methods was UML/SysML, OPM, and Simulink. OPM, UML and SysML are methods/tools that address system level type of abstractions (while UML was originally designed for software, its stereotype extension ability allowed projects to use it for SE models). Simulink is usually considered as component modeling tool – since its power is in its established toolbox. Its vendor regards it as an SE modeling tool, and in certain applications, it allowed the developer to generate the system level model, therefore it was included in the research

Creating a semi quantifiable research that sets to compare different projects is by all means challenging. Since no two projects are similar it is quite meaningless to try for random statistic research. The method selected resembles LHS (Latin hypercube sampling) where each differentiator was sampled at least twice (low value and upper value) representing to best the variability of the problem.

Overall 10 of projects 6 companies participated, of which the sample sizes were (# in brackets):

1. Size of organization – big (4), small (2)
2. Industry type – defense (4), other (2)
3. Ownership - private (3), governmentally owned (3)
4. Size of project – very big (3), big (3), medium(2), small (2)
5. Modeling effort (part of the SE effort) – large (6), small (4)
6. Type of project – new (6), new generation(3), other (1)
7. No of disciplines in SE (2-8)
8. No of subcontractors (none-8)
9. Languages/Methods/Tools – UML/(Rhapsody, ROSE, EA) (7), Simulink (2[3]), OPM/OPCAT (1)

[2] While the differentiators covered most organizational differences, the one parameter that was not covered was the local culture - since all projects were locally run. The local SE culture does not believe in enterprise directive when addressing SE tools and methods. Even though most organizations went through CMMI at different levels, the influenced areas were RM/CM methods and tools, but MBSE is left to the practitioner interpretation. The one organization that regarded MBSE as part of the organization intended process was globally owned.

The projects representatives were then interviewed (the questionnaire was updated based on the early interviews findings).

4 Research questions

Following is the list of original questions listed in the research proposal:

1. what are the success factors of MBSE usage in a project
2. What are the criteria for projects that will benefit effectively from the use of MBSE and particularly do SW intensive systems benefit more from MBSE in SE.
3. What are the criteria for choosing the proper MBSE method/tool for a project
4. What are the criteria for deciding on the level of usage of MBSE
5. Does the SE process needs to be defined differently when we use MBSE methods/tools and by whom
6. Can and when MBSE be used for direct translation to component design

In the process of the research additional questions surfaced:

7. How to create a successful technology transfer within an organization[4]
8. How much modeling is valuable and sustainable (elaboration of question 4)[5]
9. When is code generation (SW or VHDL) cost/value/quality effective
10. What is the role of domain specific modeling languages (DSL refers to the level of abstraction offered by the given language)[6]

5 Key parameters from literature review

The following part lists and describes the suspected key performance parameters based on the literature survey. The list served as the base for the developed questionnaire and was updated by the first round of interviews:

The technical by nature factors are (based on [3, 4, 5, 6, 7]):

[3] Simulink was used in more of the projects but not at the system level, rather at subsystem – usually at the control subsystem level.

[4] By technology transfer I'm referring to the process in which a new technology (in this scope MBSE) is assimilated by an organization until it becomes of sustainable use. During the search for projects to be used for the research, a peculiar phenomenon was recognized: the different tool vendors reported many organizations as users. In reality, these users which were multi-project organizations, reported sporadic usage of multiple tools, in specific projects (not necessarily the most important ones – by risk or value). Some only reported performing a demo project. The phenomena were unrelated to the success (or failure) of the usage of MBSE in the experimenting projects.

[5] Modeling all the details is effort consuming both in the first phase and even more during change management, specifically when the change is made in a different, unconnected tool. On the other hand an exact model might allow better usage of automatic tool support (code generation and debug).

[6] The research set-out to address and compare different methods and tools, primarily to make the results more general. One of the differences between the tools lies in the way these methods/tools provide the users with ontology that matches their problem domain. While UML and OPM are general purpose modeling languages, Simulink is specific domain (with the proper toolbox). SysML is considered SE specific language.

1. **Specific profile/tailoring**. Most success stories describe the usage or definition of specific profiles or domain specific language (DSL) for the domain the project belongs to and that need modeling support. This also includes specific domain ontology. Too many options or abstractions inhibit the proper usage.
2. **Tools interoperability** – most papers describe the need for tool enhancement and the creation of framework and tools glue to achieve the project goal. Success has to do with the ability to properly create them.
3. **Legacy integration** – The organizations reviewed had prior investment in Legacy systems for process use. The main ones being the interface control (ICD) tools (unusually self-made), Requirements (RM) tools (mostly doors and requisite-Pro), SW modeling tools (mostly UML based), configuration (CM) tools, and testing platforms (mostly self-made). Aside from the interoperability parameter, some of the functionality of these legacy tools could be achieved in the SE modeling tool, so this becomes a superseding parameter.
4. All the reviewed papers described **code generation** or test bed code generation as the goal of their application
5. The papers addressed the **proper resource allocation** that was needed for technical support, modelers, tool license and to compensate for the associated learning curve ineffective time, as key success factor.
6. The reviewed case studies represent **specific types of systems**. Why success in using MBD is associated with these types and not with others, cannot be directly analyzed from the literature, and was explored in the research itself.

The soft factors list included the following:

7. **Management involvement and support** - the resource allocation was put in the technical section, here it implies interest, focus and moral support. This is a general parameter to appear in most change improvement forecasts (such as CMMI)
8. The **ontology distance** – how far was the method vocabulary, structures and process from the ongoing process. This parameter is related to the DSL/tailoring process in the technical section, that is supposed to lower the ontology distance.
9. **Key users early involvement** – early users serve as agents of change
10. **Tool/method lightweight**, easy to learn and intuitive (in the specific domain)
11. Key users can change the tool/process at early stage – this parameter, was merged in the questionnaire with the tailoring parameter.
12. Process improvement **based on current process analysis**

Additional soft factors, gathered from organization/sociology/communication literature, are based on ref [11, 12, 13, 14] and on the list from [15]. These references cover aspects of tools and methods, innovation, software and process change. The domains were selected based on their similarity to the problem at hand.

The following list comprises the relevant key factors, elicited from these references:

13. **Perceived success** of MBSE use - unlike measured success, perceived success is a "soft" evaluation criterion.
14. **Maturity** of tool/method (real and perceived)

15. **Organization incorporation process** - was it treated in a structured way with sustainability as a goal, was it recognized as innovation and treated as such.
16. **"packaging"** – this criteria covers all the aspects dealing with support, training, documentation, etc.
17. **Heterophily of group** – the more the user group is homogeneous the more the change is easier to achieve.
18. **Champions model** at place –champions are people who can lead the process and be the first to perform it (unlike early users who simply take part in the first trails).
19. **Perceived ROI** (return on investment) – it is interesting to note that none of the technical references addressed measured ROI, but the "soft" factor list had the perceived ROI as a major factor and to test it, the questionnaire addressed **ROI** as well.
20. **Clear benefit** to technical problem – is it clear what technical benefit will the project achieve by using MBSE.
21. **Compatibility with existing process** – while soft factor 12 has to do with how you design your change, this parameter looks at how far the new process is from your current one.

Grouping these factors the following families of parameters can be defined:

1. Ontology gap and its methods of closure
2. Value to project
3. Tool features
4. Change process
5. Management commitment and resources

In the following paragraph those key factors will be analyzed against projects performance.

6 Research analyses

6.1 MBSE interpretation

Before analyzing the research parameters, more should be said about the use cases of MBSE as met and selected in the initial participant search process. The following families of uses were encountered:

1. Models use for components development – an example for this use would be a control model of the control unit of a system, developed in Simulink. The reason that this application was not selected for the research (although it did a full cycle development, including code generation) was that it did not capture the full system scope.
2. Models used for optimizing or proofing certain aspects of the system – several examples for this use presented themselves – a model for optimization of certain product features (i.e. weight model), a heat transfer model (used for comparing design options and setting requirements for components), analysis of mechanical dynamics using finite element model (LS-Dyna). Since these examples did not present multiple aspects of the system they were not used for the research.

3. The system model approach for early validation – an example for this use is a full simulator built as an internal tool for a company, where the simulator environment allowed the project to develop their different components SW (i.e. control SW) and model the environment (i.e. wind model) and run them together in the simulator, using different scenarios. This tool allowed the project to early validate and early integrate its implementation (The same SW code was used for the end product). While this is certainly a system level model based approach, it was not use for the research since it could not be compared to the other applications.
4. The family of applications selected for this research presented uses were the model was a system level model, describing multiple aspects of the system and used for various SE process uses. OPM, UML and SysML are languages supported by methods/tools that address this type of abstractions. Simulink is usually considered as component modeling tool – since its power is in its established toolbox, but in certain applications, it allowed the developer to generate the system level model therefore it was included in the research. It is rather like comparing apples and bananas, yet since the goal of these tools is to provide a full system model, the result can be compared .

6.2 Organizations SE Change Culture

All the companies selected for the research, develop complex systems – thus need a good SE base. All but one of the companies, are long standing organizations with SE traditions and so called culture. During the past decade all of these companies went through SE process definition phase, particularly influenced by CMMI to various degrees. MBSE in itself does not belong to a specific key area in CMMI. It rather holds the potential to improve performance in different key areas such as requirement definition and management (RD, REQM), technical solution (TS), and IV&V (PI, VAL, VER) by improving their results. While not directly supporting a key area, the CMMI was supposed to leave these organizations in a gradual improvement state of mind and culture of process change management. Using MBSE in organization should have been part of that process improvement thus its deployment in an organization should have adhered to key area OPD (since most organizations stopped at CMMI level 3). The one organization that was small and new was formed by people with large company experience (so they had the lessons learnt) but also had the freedom to form their best fitted, up to date solution.

6.3 Findings

The following part describes the findings of the research. The first part describes the general findings and the second part correlates the findings and the key parameters.

Types of usage - The following usage types were reported in the research projects. In brackets is the number of reporting projects:

Scenario analysis – mostly use cases analysis (8).
Part of documentation (design & requirements) (8).
Stakeholder information – model used as a mean to communicate and convey

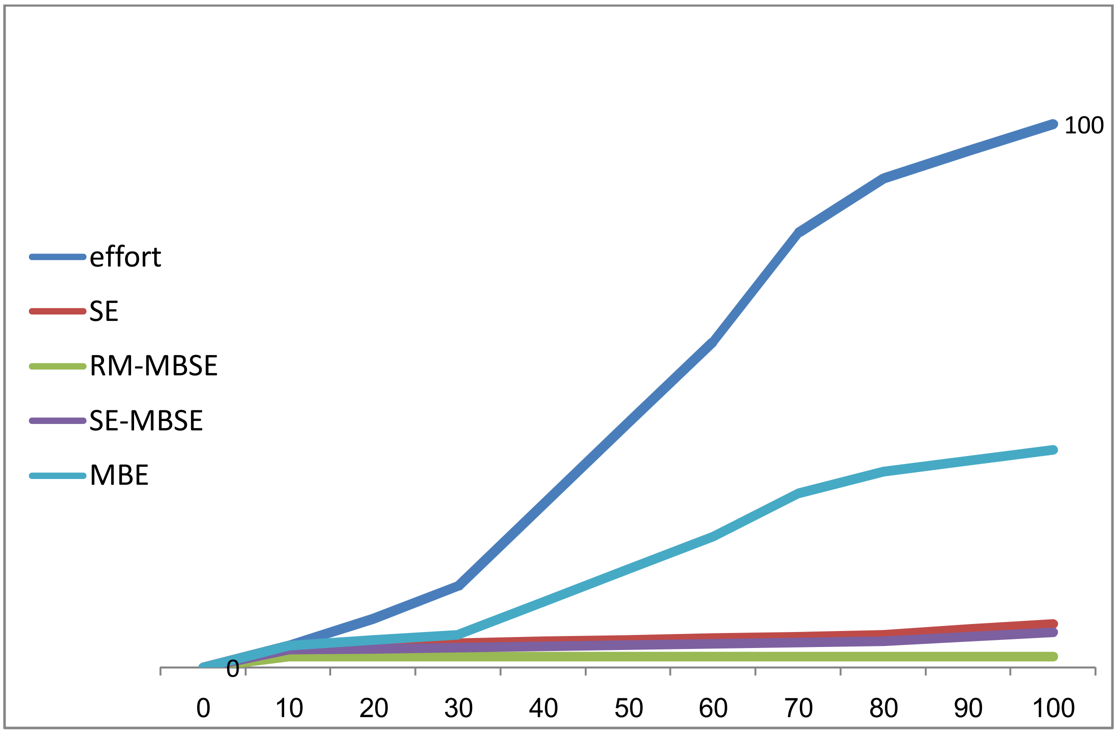
meaning to various stakeholders (6).
Requirement elicitation and Functional decomposition (6).
Part of maintained documentation (5).
Check Design issues (5).
Test scenario creation (4).
Trade studies (3).
Design completeness check (3).
Automatic code generation (3).
Automatic simulation generation (3).
Reverse engineering - of an existing project was used to create a model for several purposes: test generation, help in changes impact analysis and creating a new version (3).
Architecture trade studies (2).
Operation Research (2).
Prototype creation (2).
New team members training (1).
Exploring version change implications (1).
Base for Software modeling (1).
Environment simulation (1).
Human-machine interface prototype (1).

What is apparent from the usage analysis is that most projects still primarily use MBSE in the requirements phase, mostly using it for Use Case/Scenario analysis (either at the business level or systems level). Full cycle usage (either in automatic code generation or for system test scenario) is still emerging.

This use presents low commitment and low investment and limited impact versus full cycle use that presents higher commitment and investment, but at the same time higher benefit and impact.

In the low investment part there are two distinct groups –

- Moving towards a full SE model where most of the SE work is done directly on the model
- Limited use as document annotation and top level system understanding



1 : MBSE effort vs project time (%)

The graph above schematically represents the different MBSE investment scheme (percentage) used in the different projects – light MBSE (RM-MBSE), model based Systems engineering (SE-MBSE) and full model based engineering (MBE). The blue line represents the project effort and the red one represents the typical SE effort out of it (around 8%).

With the selected definition of MBSE (elevated to central role) 4 of 10 projects went all the way when regarding the process stages influenced by the model. When analyzing resource investment 5 of the 10 invested high resources into the model.

Details and structure – Most projects mentioned level of detailing as a major issue in the MBSE effectiveness. The following contradicting considerations were mentioned:

- Going into details allows you to create an executable model and go into direct code generation
- Detailing for code generation had a down side that needed controlling, not in all the cases was it less expensive (in life cycle cost) to model then to write the code by hand or use it from previous projects.
- Going into details is labor consuming – when modeling in a tool it means you have to tie all the loose ends.
- Updating a detailed model requires a lot of work of configuration management, some of its value questionable.
- Some systems engineer don't care about the finishing touches required by the model
- Some (or most) of the effort might be wasted if the data flow is not defined and maintained – for example if the details or updates are made in different, unlinked tools (i.e. SW tools) the basic investment in the model is soon lost.

- Detailing exception scenarios (previously done at later stages of SE process or by implementing disciplines) improved the system specification maturity, but seemed in cases like a waste of energy.

The structure issue was more straightforward – In order to be workable and presentable a model needs to be decomposed into views that allow the different users to view their concerns without being flooded by information.

UML vs. SysML – An interesting finding had to do with projects using UML rather than SysML, as a SE tool. Some of the projects were set-up before SysML was offered by vendors. But some of the projects made a conscious decision. The following reasons were given to such a decision:

1. UML is supported by the organization IT services – being the choice of the SW engineers, therefore by using UML, support and practitioners' knowledge was guaranteed and also a large investment in the "packaging" was saved.
2. The additional features provided by SysML over UML did not seem important enough to change from the known UML – by choosing the right diagrams and making the proper extensions supported by the language, the advantage of SysML became marginal in practitioners view. While SysML offered solutions for requirements and ICD, most projects decided to use their legacy systems for that purpose.
3. Since tailoring was perceived as needed both for SysML and for UML the organization did not see an edge in using SysML over the additional tailoring needed if UML was used.
4. Most organization thought using the same tool for SE and SW will be beneficial (mostly the SE community point of view) in the transfer process.

Technology transfer – Only one organization used a fully concise technology insertion process – and that organization is the only one that had shown sustainable performance in MBSE. However that same organization was slow to use (in the researched project) the more advanced features of code generation or test scenarios generation. The other organizations had performed at least one example of MBSE in a project, but while regarding it as a success, had an issue with making the success sustainable. The usage of Simulink was more easily spread and sustained. All the projects were based on a champion model, that needed to be repeated for each additional project instance. The champions being either

- Someone with hands-on relevant experience from previous project,
- Someone senior enough in the project, with a clear vision or
- Someone senior enough from the support group

The various organizations had different knowledge sharing methods:

1. SE forums
2. SE support group – these groups served this purpose only if they dealt with methodology issues and had senior enough support people.
3. Adapt and require – in this method, following a SEPG adaptation process the MBSE becomes mandatory in the organization.

Only one organization chose the third method (and created sustainable results).

Packaging – support and training were essential in the success of the method incorporation. Every project and organization had its own mode of "packaging". The most important feature of it was the adaptation process – since most methods explored in the research could not be taken out of the box, someone needed to supply that service. It was achieved either by acquiring outside help, or by assigning someone internal to the project for that purpose. While it is true that without proper packaging the MBSE process had failed, the existence of it did not guarantee sustainability. Most of the projects explored had to manage the packaging by themselves (because the organization was not set-up to do that).

Management support –while some kind of management support is essential (if any, most tools simply require upfront investment that needs to be approved), the sponsor identity varied highly between projects and organizations. The lack of sustainability might be attributed to the lack of management directive in their support.

Success and ROI – Measuring ROI in process improvement (and MBSE is mostly process improvement) is known to be a problem. Little had changed since the statements regarding ROI in process improvement [9]:

" 1) There are no “hard numbers.” 2) There will be no hard numbers in the foreseeable future. 3) If there were hard numbers, there wouldn’t be a way to apply them to your situation, and 4) if you did use such numbers, no one would believe you anyway".

This is again supported by CMMI latest results [10] showing large variance in organizations results. MBSE is even more problematic – part of the modeling time is strictly SE thinking time that should have been carried one way or the other, making it difficult to analyze costs. To make ROI really irrelevant the MBSE in some cases had given projects an edge that was the show-stopper – not achievable any other way[7]. So instead of trying to measure quantitatively let us look at qualitative result:

1. All projects interviewed in the research perceived their use as being successful and holding positive ROI (them and their organization)
2. Time saving - All projects described qualitative saving in time, no matter how far they had gone with MBSE. In all projects the savings were noticed at the analysis/requirement development phase. In the code generating projects, this encompasses the savings of the implementation phases, in the different iterations.
3. Productivity – When doing direct code generation, the engineering productivity is improved. However the use of MBSE does not necessarily improves the productivity of the SE process. Some systems engineers even found it holding their productivity back by requiring them to tie loose ends. At the long range it is supposed to improve productivity by lowering the # of iterations needed to get to the same level of quality but that is a measurement that is hard to get.

[7] Such examples in the reviewed projects were either because expressing the project complexity was too hard without a repository based model, or code generation was a method to sidestep the problem of missing SW resources that couldn't be allocated.

4. Quality – the quality of a system can be measured by # of defects and # of required iterations. A system described by models tended to be more explicitly defined and communicated, therefore achieving earlier maturity as expressed by the project and customers.
5. Investment – while all reviewed projects representatives felt that the value in their MBSE effort superseded the investment, several cases of previous less successful use of MBSE came-up during the interviews, where the related investment seemed to be partially wasted. These projects had the following common failure reasons:
    a. Wrong tool for the problem (too complex, too much tailoring needed, too little support, immaturity)
    b. Wasted detailing effort – either because due to lack of tool interoperability the investment was lost, or because the detailing did not improve the SE result (see more in the following paragraph)

Having said that, only 3 projects had predefined ROI in mind, at project kickoff. These projects clearly planned the modeling effort and decided in advance what part and how deeply will they model, and what is the most cost effective way around it. Of these three, one did it because it was the company way, one as lessons learnt from prior less successful trial, and the third was simply very cost conscious.

Syndrome of non-sustainability – two unrelated phenomena were observed. One had to do with sustaining the models in the specific project. Of the projects that did not generate code or testing directly from the models, most projects reported not continuing to update the SE models. If any, only SW models got updated. While the reason seemed to be related to a tool interoperability issue (when interoperability does not exist, you need to update both the SE models and the subsequent tools separately, and benefit little from the effort), there was penalty involved with abandoning the SE models – for example some features that were captured in SE models but were not a functional SW feature disappeared all together from the testing.

The other non sustainability issue had to do with MBSE not becoming part of the organization process, even after a successful use in a project within the organization, with perceived ROI. This non-sustainability issue certainly does not conform well with the supposedly change state of mind that should have been expected based on first paragraph of this section.

The exception to these observations was the starting of Simulink to be organizationally applied to projects in one of the bigger organizations. Observing this exception by the key performance parameter shows that the following parameters foresee this –

- Simulink ontology gap is lower than the rest of the methods (building blocks are less abstart)
- The systems that use the MBSE were of certain type (real type control systems)
- The SE of these systems were of more homogeneous background (control and SW engineers)

- There was a primarily champion model application and the process was adapted to the rest of organization
- For most of the practitioners the difference from the way they worked before was not great (low process gap)
- The value achieved (code generation) was considered big

Influencing the existing SE process – The research question tried to explore the changes (if any) to the typical SE process (the typical would be V based incremental in most organizations). While not necessarily defining it as such, the projects that achieved automatic code generation, managed as a result to start integration earlier than the typical process – thus shortening the whole development process and making it parallel rather than sequential. The other projects improved their requirement validation process thus making that phase shorter, but did not change the process itself.

Process improvement - In 5 out of 6 organizations this is a valid question (since they are multi-project organizations and have explored process improvement techniques such as CMMI). The other organization is new and is still a one project organization. Of these 5 organizations, only one moved towards MBSE in a structured process improvement manner.

Key parameters findings and correlation - The research findings are grouped in the following table based on the literature review key parameters, with the additional parameters found in the research itself. Each finding is described.

Key Parameters	Findings
Tools interoperability	Several major interoperability issues were raised – to RM tools, ICD tool and SW tools. These tools, in most organizations, were selected prior to the MBSE modeling tool and were considered Legacy. The lack of interoperability and the need to find solutions for data transfer was mentioned in most projects as an issue. While not influencing the perceived success of the MBSE process, it did influence the likelihood of sustainability of the models in the project that tended to abandon changing the requirements in the models and support only change in the RM, ICD and SW tools.
Specific profile/tailoring (including ontology distance)	The general purpose tools and methods (UML and OPM) required adaptation and modeling decisions such as which diagrams to use, how to represent different entities. The proper process (and the knowledgeable people being able to perform it) influenced the success of the modeling heavily. SysML as an SE profile was not found supportive enough to be taken as is. It was easier for organizations to make sustainable use of more specific domain tools (Simulink). While the general abstract languages provided a better solution for general complex problem, the adaptation process needed to be repeated for each instance – requiring the proper know-how people vision and availability.
Process improvement	In most organizations (4 out of 5) the decision to use MBSE was a local project decision (driven by either need that could

Key Parameters	Findings
	not be met otherwise, or local vision), or a fashion based decision. As a result most projects did not have upfront vision for the process and its organization follow-up.
code generation or test bed code generation	Only 5 out of 10 of the projects used it. All of them defined it as a major issue influencing their decision to move to MBSE. The likelihood of sustainable use of the models in the project itself was much higher (80%) in these projects than in the other projects (20%)
Proper resource allocation	As analyzed before, this presented a key factor – rather in the failure avoidance. Different MBSE methods and tools required different size and types of resources.
Type of application	As noted is the literature review the systems that benefited most, by MBSE, were Control systems, SW intensive, and systems with horizontal Complexity. Control systems effective use is due to existing toolboxes and dynamic behavior property. SW intensive and horizontal complexity (meaning complexity that is a feature spread across the system rather than residing in its components) are there sharing the features of abstract complexity that defies a one person ability to grasp the system without a model.
Management involvement and support	Three sub organizations should be addressed regarding support needed for the process – Higher management support was needed to support funding, backwind, its lack of vision tended to be the driving reason for non-sustainability. SE infrastructure –the SE support group. That group structure, goals and abilities varied highly between the different organizations. From central groups that could drive the vision, to IT support that could only solve tool problems. Most projects had to solve their modeling problems and make MBSE decisions pretty much alone, or with outside consultant help. That in itself matches the current state of emerging technology use. Only in one organization, the SE group had the mandate to make a sustainable change through-out the organization. The lack of proper support burdened the projects with additional work unrelated to project direct goals. When the need and vision in the project were high enough, that did not create a usage barrier. It does create a barrier in projects that exhibit less internal insight or need.
Key users early involvement	Most projects (9 out of 10) ended up being the first of a kind, therefore their key users were involved from start. However other projects in the same organization were exposed to the process only when status was established. In one prior failure case the lack of key user involvement created a problem that stopped the MBSE effort (decision was made centrally outside the project without involving key figures).
Tool/method lightweight	Good fit was demonstrated between the success and the perceived lightweight of the tool/method – however this

Key Parameters	Findings
	cannot be judged by objective criteria - proper support and packaging made the same tool more users friendly.
Perceived success and ROI	All projects regarded their use of MBSE as successful and investment returned. Less than third actually measured the investment and only 2 could discuss it in actual ROI terms. 2 related problems in measuring repeated themselves 1. Separating the system analysis process from the modeling time, 2. Attributing the benefit to MBSE use (shortening the time to the next design review was easy, but measuring the quality or maturity of the product is less easy).
Maturity of tool/method	Good fit was shown in all projects.
Organization incorporation process	Only 1 out of 5 organizations had shown a consistent SE change process regarding tools/method – this is the only organization that made a sustainable use.
"Packaging" (support, training, manuals..)	Packaging proved important in all project, but different means were used to achieve it – from in-house support to tool vendor or outside consultants.
Heterophily of group of users	SE IPT group is always heterogeneous. The background discipline of each representative influences his/her ability to use MBSE at a system level. SW/control/EE engineers tend to adapt better. The other disciplines tended to adapt late – if at all (drawing models by hand or Visio and letting other members do the model update). Another factor is the age gap – older engineers tend to be less tool-oriented, creating the same result.
Champions in place	All projects reported champions involved in the project or in the project support, in that aspect there is a good fit. However for the more general modeling languages, most organizations needed to reproduce a champion model in other projects in order to repeat the success. Two champion roles were observed – the visionist and the know-how. Sometimes the same person served at both roles.
Clear benefit to technical problem	Of the projects explored 4 projects could simply not do without – either because expressing the project complexity was too hard without a model, or code generation was a method to sidestep the problem of lack of certain resources (SW people). The other 5 projects had a perceived benefit rather than clear one.
Compatibility with existing process	This did not present a key factor
Legacy	The current legacy in most organizations is that requirements management is done in one tool, ICD in another, and SW models in their own tool, etc. In new MBSE tools all these activities can be performed in the same tool. While presenting an opportunity in reality it created friction with the surrounding organizations – QA, SW, T&E

7 Discussion

Based on the research findings let us explore the different aspects influencing the use of MBSE.

People aspect

The practice of MBSE in the newer fashion moves the models from specific phase, process support tool, to a SE backbone artifact and a production tool, which is directly associated with the performance of the product. While systems engineers usually embrace technology change when it comes to product performance, this shift of paradigm is not trivial for all of them – to some, it is not gradual and requires a new way of thinking, and therefore most of them resist the change. The ones that do embrace the change belong to the following groups:

1. Abstract Oriented - Tool oriented and modeling oriented people belong to that group (most SW engineers belong to that group).
2. Previous knowledge – in some domains the new approach is just one step in a gradual change over the years. For control engineers using toolboxes and generating code was just one more step (which happened to also remove the need to recruit SW engineers).
3. Champions – the first adopter in each organization needed to be a champion. Most organizations did not provide proper internal support for a new process. Some of the organizations were conscious enough to hire outside help – some not. Still the champion of the project role remained essential to the definition of the use methodology of MBSE for the project scope. The project champion sometimes works in a vacuum with no organization support at all.

Project aspects:

1. Specific types of systems have an obvious need that cannot be satisfied in a different way (customer requirement, standard push, complexity handling). In these types of systems the change is driven from the outside[8].
2. Specific projects had a crisis that could only be addressed using MBSE (such as lack of resources of specific type).
3. SW intensive systems use MBSE more than others (need for complexity handling and tool/method oriented practitioners available) – while this being the case, the SE models even if similar to the SW models in modeling language could not replace the SW models.

Tools/method aspect:

[8] You would expect high reliability systems and critical components, to be such examples. However since the different regulator authorities do not mandate it, and on the other hand the tool vendors do not offer qualified building blocks, this direction is still left unexplored.

Tools and methods are successfully used when they become "users friendly". In the MBSE example it encompasses tool lightweight, maturity and support (general for tools) and closure of ontology gap.

1. In most general purpose methods there is a need (and usually support in the tool) for tailoring to specific ontology (also known as DSL – domain specific language). SE profile (such as SysMl) does not appear to be enough to be used as is and a specific domain tailoring is needed so the user (the Systems Engineer) will feel comfortable enough to use it.
2. In specific fields the profiles are good enough to allow for models becoming part of the production rather than process enhancers. In these cases the change, regardless of the way it started, manages to become a sustainable achievement. Control system use of Simulink seems to be such an example. Both the toolbox offering (building blocks, low ontology gap, maturity of tool and provided code), and the rather limited hetrophility of the SE users made the ROI better understood and the technology transfer easy to achieve.
3. The other sustainable application seemed to be about a successful specific field tailoring by qualified person or group, constant support for the process, and organization directive.

Organization aspects:

The following types of organization involvement were found in the research

1. Organizations with a well defined SEPG that is driven by management support and provides proper "packaging". In these organizations (only one in this research) change is well structured and ROI is well defined. Even in this organization, different process was needed for different types of applications. It is interesting to note that the change rate, in this organization, did not follow the change in market offers – thus not enabling the full benefit of the latest achievements.
2. Organizations with proactive SE support structure – the SE support group tried to foresee the projects demands, explored method and tools and was able the advise and lead the structuring of the MBSE process in the specific project. However the use of MBSE remained the project leaders' decision.
3. Organizations with reactive SE support structure – after the success of a project in incorporating MBSE, the organization SE support group was involved in generating the "method for all" procedure and approving the need for IT support. In these organizations the champion was more or less left to face his/her decisions. The organization involvement was minimal during the process.

Another aspect of organization behavior had to do with SE knowledge sharing. The organizations used the following methods

- Knowledge forums
- Through SE support group activity
- None or sporadic

8 Summary

The paper had explored the various factors influencing the success of giving MBSE a central role in the SE process and making it a worthwhile investment. 10 projects of 6 organizations were interviewed to support it.

The Following table summarizes the degree to which the key performance parameter supports the research findings. "+" denotes high correlation. "~" denotes medium correlation.

Key Parameters	Support Success	Lack Influence Sustainability
Tools interoperability	~	+
Specific profile/tailoring (including ontology distance)	++ (SysML not enough)	
Based on Process improvement	~	
code generation or test bed code generation	+	+
Proper resource allocation	+	
Type of application	+	
Management involvement and support	+	+
Key users early involvement	~	
Tool/method lightweight	++	
Perceived success/ROI	+	~
Maturity of tool/method	++	
Organization incorporation process	~	+
"Packaging" (support, training, manuals..)	++	+
Heterophily of group of users	-	
Champions in place	++	+
Clear benefit to technical problem	+	
Compatibility with existing process	~	
Legacy Adherence	+	+

This list of key parameters can be used to design for MBSE successful use, and for improving sustainability of its use within a project and organizations. Vendors and organizations can use it.

In the following table I summarized specific actions that a project can carry-out to improve its MBSE use. It is assumed that the project is the first of its kind to use MBSE in that specific way, in the organization environment. The goal is to enhance the performance in these key factors thus creating a better chance of success:

Key Parameters	Activities
Specific profile/tailoring (including ontology distance)	1. Congratulations you are the visionist. Don't worry about the rest of the organization. 2. Decide why your system would benefit
Tools interoperability	
code generation or test bed code	

Key Parameters	Activities
generation	from MBSE (complexity, SW intensive, control). Decide what parts of the SE process would the model support. Don't hesitate to select only partial implementation 3. Define in advance which MBSE achievement you want (see list), who are the modeling stakeholder and how are they going to use the model 4. Assign a champion for the process (from now-on it becomes his task) 5. Select your tool based on the problem you are facing. Comparing the various tools for your example is a good idea. Maturity must be a criterion. Get a support package (including training) either from external sources or internal. The common user should get to feel that it is light weight enough 6. Define your specific domain ontology (based on the stakeholders of the model) and tailor your tool to match 7. Decide where to draw the line – what details are important and what should be carried out in other tools 8. Decide how to manage the information in your multi-tool and legacy rich environment, and how to pass information around. Think of the change process. Try to minimize the number of tools 9. Design tools interfaces and if needed implement adaptors between the tools you are using 10. Decide if your work is on the model or on documents annotated by models (if the later, try for automatic production) 11. Buy-in your customer, and if needed your manager 12. Decide which part not to model (based on value). You don't have to model all parts to the same detail level. 13. Observe and solve problems – if your model is intended as a long lasting one, new problems would rise, when new users come along.
Management involvement and support	
Tool/method lightweight	
Maturity of tool/method	
Organization incorporation process	
"Packaging" (support, training, manuals..)	
Champions in place	
Legacy Adherence	

A similar analysis can be carried-out for other stakeholders – tool vendors, support groups and organization management. By methodically doing so, the effectiveness of MBSE use in the project and organization can be raised.

The projects participating in the research had shown that MBSE is here to stay. They also had shown that different successful uses can be pursued. To make it more sustainable a list of key performance parameters was defined and influence level was analyzed. By influencing these parameters we can improve use rate and success.

Acknowledgement:

Setting out to find enough cooperating projects, from different companies had required activation of all my past links. My work at INCOSE_IL provided me with helpful and knowledgeable colleagues, but still getting the approval of the companies was not easy and some of them did not even approve being listed. Once approved, the project representatives were very open and helpful. I would like to thank the participating companies and their representatives – Paul Friedman, Eran Peleg, Irit Peled, Asaf Shalev from Rafael, Mimi Timnat, Yaniv Rosen, from Elbit and Elbit/Elop, Mike Turgeman from Urbanauro, Koby Grudka (and Haim Reichman) from Elta/IAI, Dalia Pongratz from Motorola IL, and Sergey – thanks for sharing your knowledge with me, and I hope you'll approve the result.

R

[1] Jeff A. Estefan,Survey of Model-Based Systems Engineering (MBSE) Methodologies, INCOSE Survey of MBSE Methodologies INCOSE-TD-2007-003-02 5/23/2008.
[3] Harwell, Richard; Redell, Ola; Malvius, Diana; Ritzén, Sofia, Introducing Structured Information Handling in Automotive EE Development, Proceedings of the 16th Annual International Symposium of INCOSE
[4] Piggott, Stephen; Melanson, Philip; Hartman, Leo, Vision for Super-Model Driven Systems Engineering, IS INCOSE 2007
[5] www.c-lab.de/uml-soc
[6] Torngren et al., Component-based vs. Model-based Development: A Comparison in the Context of Vehicular Embedded Systems, Proceedings of the 2005 31st EUROMICRO Conference on Software Engineering and Advanced Applications
[7] S. Ramesh, P. Vignesh, V. Ganesan, Gurulingesh Raravit, A Formal Framework for the Correct-by-construction and Verification of Distributed Time Triggered Systems, India Science Lab, General Motors Research, IEEE xplore
[8] S. R. Piggott, P. Melanson, L. Hartman, C. Adourian, Advancing Model-Based Systems Engineering at a Small Space Agency, IS INCOSE 2008.
[9] Sarah A. Sheard and Christopher L. Miller, The Shangri-La of ROI, IS INCOSE 2001.
[10] Diane L. Gibson, Dennis R. Goldenson, Keith Kost, Performance Results of CMMI®-Based Process Improvement, TECHNICAL REPORT, CMU/SEI-2006-TR-004 ESC-TR-2006-004
[11] Desouza et al., Crafting organizational innovation processes, Innovation: management, policy & practice (2009) 11: 6–33
[12] Leonardi, Why Do People Reject New Technologies and Stymie Organizational Changes of Which They Are in Favor? Exploring Misalignments Between Social Interactions and Materiality, Human Communication Research 35 (2009) 407–441
[13] T. Punter et al., Software engineering technology innovation – Turning research results into industrial success , The Journal of Systems and Software 82 (2009) 993–1003
[14] Punter et al., Sustainable Technology Transfer, TT'06, May 22, 2006, Shanghai, China, ACM
[15] Pfleeger, Understanding and improving technology transfer in software engineering, The Journal of Systems and Software 47 (1999) 111-124

Appendix A – Ref Literature survey summeryver4.doc

Appendix B – Research Questionnaire

General Project Parameters
Project
Organization
Organization Size
Is it a multi-project organization
Project Size
No of disciplines
No of subcontractors & partners
Type of project
Modeling used for
stakeholder information
Requirement elicitation and Functional decomposition
Scenario analysis
Check Design issues
Trade studies
Design completeness check
Part of documentation (design & requirements)
Automatic code generation
Automatic simulation generation
Test scenario creation
Part of maintained documentation
reverse engineering
HMI prototype
Architecture trade studies
OR
Prototype creation
New team members training
Exploring block change implications
SW modeling
environment simulation
Process use part
Project Process
MBSE method
MDB tool
Development stages for which the tool where used
Deliveries using modeling output
is MBSE used and maintained
If not why
Who chose method/tool
Who sponsored
Main SE discipline
Tools
name
Was there a need for adaptation
how was Configuration Control performed

Training for modeling team
interorganization support
Technical support by vendor
Tool performance influenced success?
If multimodel for same development stage - how connected
other modeling tools - how connected
How many people modeled
How many hours invested in modeling
How many extra people used the modeling output
Was the model used for interoraganization cooperation and information sharing
was the model used for outside organization cooperation and information sharing
Effective model detail level
Coding
was it planned
conclusions
Will you use in future and when
Do you regard as success
Does the organization regard as success
What would you do more effective
is it the first project in organization to use MBSE in SE
What kind of knowledge sharing does your organization have
was it considered innovation
was it part of organization decision
Who checked the tool and what were the criteria
Was it clear at the beginning what you were trying to solve
Were there legacy systems that the model had to link with
issues