Abstract—A lot of systems developed in MBT Missiles Division of the Israeli Aerospace Industries are derivations of former deliveries; sometimes they were developed over decades, and were redesigned for different customers or were upgraded as technology was improved. In addition, the requirements management (RM) process in MBT has to comply with an extensive system of systems, a matrix organizational structure and many sub contractors in order to develop the fully operational system. It was obvious that generic sub-systems will be reused in projects of the same product family, which started in reuse of their specification. The only question was in what level of RM the reuse will start between contract requirement to the software and hardware components specifications. Eventually, reuse occurred in all levels of requirements in different ways for different purposes with great improvement of time to market. This paper presents real industrial experience of requirements reuse along with results and analysis that demonstrate the impact of requirements reuse on time to market.

Keywords- requirements; specifications; management; tools; process; reusable software

I. INTRODUCTION

A. Scope

MBT Missiles Division of the Israeli Aerospace Industries (IAI) is developing large scale defense systems [18] incorporating thousands of person-years projects. A defense system is usually a software intensive system and takes 5 to 10 years to accomplish full delivery. During that time, many experiments both in laboratories and in the customer environment take place in order to test partial deliveries of the system.

The requirements management (RM) process in MBT has to comply with an extensive system of systems, a matrix organizational structure and many sub contractors in order to develop the fully operational system. The full scale project development involves different parties, including: Other IAI plants, different enterprises with joint venture contracts, outsourcing consultants and off the shelf products.

The organizational structure is built around the principle of product families. New developments are always customer-driven, while there is great similarity between customer needs and resulting projects within the same product family. In fact, each new project is based on another project in the same product family. Thus, the reuse of generic software and hardware components is highly desired.

By 2006 it was decided to use DOORS IBM/Telelogic tool [13] as the requirements management infrastructure.

B. Motivation

Requirements management [16][17] is particularly important to MBT for a number of business reasons:

- The main goal of all quality improvement processes in the company relate to improving customer satisfaction.
- CMMI maturity level 5 is a key business goal, thus requirements management (REQM) and requirements development (RD) is mandatory to all projects stakeholders [1].

MBT sought to implement a requirements management process, far beyond “producing documents.” The process would include a change in language, reviews, workflows, and in fact the whole project culture. The benefits of such a process would be to:

- Resell existing solutions to new customers, with minimum additional development.
- Develop generic solutions for multiple customers.
- Reduce development and delivery time.
- Use requirements as the backbone of the development.
- Reuse requirements of generic components as a basis for reusability in different projects of a product family.

Reuse is a major component of many software productivity improvement efforts, because reuse can result in higher quality software system at a lower cost and delivered within a shorter time period [2]. One of the critical reuse
success factors is the adoption of the product line approach of Boehm [3][4].

The granularity of reuse can vary from the narrowest range of reusing a single artifact, such as a component, document, or test case, to the widest possible range, namely, a whole product. Reuse can take place during any phase of the life cycle, including marketing and proposals, requirements, analysis, design, coding, and testing. Performing reuse at a certain level of reuse (life-cycle phase) usually carries with it reuse at all subsequent levels. Thus, reusing requirements is most beneficial in terms of development effort and time to market.

The purpose of the paper is to summarize the implementation experience at MBT with requirements management in order to demonstrate a real industrial experience of requirements reuse.

C. Background

When requirements are elicited and managed correctly, they facilitate effective analysis, design, development and testing of the system. On the other hand, “Bad” requirements are proliferated through multiple branches of the development process, leading to low quality system.

Seventy-one percent of all software intensive systems development projects result in complete failure (i.e. premature cancellation or shelf-ware upon completion) [5]. Poor requirements management is generally considered one of the major causes for product failure [6], [7]. After all, if we do a poor job of understanding our customers’ needs, if we do a poor job of deciding the right specifications to build and if we do a poor job of writing down what we think we want out of a system, how can we possibly expect a successful project? [8]

Implementing a formal requirements management process, such as that defined in maturity level 2 of CMMI [1], enables an organization to find defects in requirements earlier in the development process, keep requirements specifications up to date, and communicate requirements change to the development teams.

It is widely accepted that the organizational challenges of software reuse outweigh the technical ones, as summarized by the “STARS” program report, for example [9]. As a result, metrics are needed in order to make “business decisions possible by quantifying and justifying the investment necessary to make reuse happen” [10][11].

In fact, there are many software reuse success stories in which the economic benefits are evident in practice. For example, substantial costs were saved due to implementation of software reuse in the “STARS” demonstration project. The first system that was developed cost 43% of a reference baseline and a second cost only 10% [9]. Another example is the experience at Hewlett-Packard described in [12] where, by applying software reuse, a defect reduction of 15% was achieved and productivity increased by 57%.

D. Outline

In the previous sections, we have presented MBT, and the company’s motivations for launching a requirements management process and infrastructure for reuse. The rest of the paper describes the requirements management process that was chosen and the analyses experience gained from implementing both the process and underlying infrastructure. The paper also describes the requirements management process.

Section 2 describes the project management and system engineering stakeholders and how they affected the requirements management process. Section 3 describes the requirements management process, and Section 4 describes the requirements reusability. Section 5 describes the measurements taken, in order to evaluate the progress of the requirements management process and reuse. Section 7 summarizes both past and future work, in terms of the benefits that MBT has already reap, and those that it expects to reap in the future.

II. PROJECT MANAGEMENT AND SYSTEM ENGINEERING STAKEHOLDERS

A lot of systems developed in MBT Missiles Division are derivations of former deliveries; sometimes they were developed over decades and were redesigned for different customers or were upgraded as technology was improved. Therefore the requirements environment includes multi projects, multi documents, and multi stakeholders.

The projects stakeholders in MBT include: project manager, program manager, system engineer, software and hardware developers, and testers both in labs and simulations. Usually, the same system engineers and the same hardware and software engineers develop projects in the same product family.

Project Managers are responsible to review the contract (annex) and estimate project cost, effort and schedule. System Engineers are responsible for defining the system and subsystems specifications and keep track to their development. Software engineers and hardware engineers are each responsible for his components specifications.

III. THE REQUIREMENTS MANAGEMENT PROCESS

The System Engineering group has taken the lead and started managing all project specifications in RM tool DOORS [13], including contract requirements, system specifications (SSS), subsystem specifications (SSDD), algorithms, and software specifications (SRS). Since MBT as part of IAI was already committed to the DoD-STD-498 [14], requirements templates exist, their equivalence was defined in DOORS and requirements content was available in MS Word documents as part of normal project deliverables. Once existing specifications documents were captured into the RM tool, it became obvious that they need to be converted into well-defined specifications.

A. Requirements Engineering Initiative

Motivated by CMMI maturity level 3 MBT decided to implement a RM tool. Two programs were ongoing with 2 different customers to provide two large scale defense systems with minor requirements differences between them. In addition, another large scale contract was negotiated, which was a derivative of the same defense system but due to operate in a different operational environment.
Thus, it was obvious that a requirements management infrastructure is required, and it was clear to the system engineering group that a solution to reuse contract requirements, requirements specifications and requirements documentations as well as the actual components, is mandatory in order to achieve contract goals economizing time, money and manpower.

The expectations of the RM tool and process were: Easy capture of the requirements into the tool database: Enable many employees working simultaneously (it was already planned that the same leading engineering group would develop all the projects family accounted) : And most importantly, that the requirements could be reused and serve all the projects without duplicities.

The initial plan was to manage only the contract and system requirements levels. However, as more users were introduced to the RM tool, especially the software team (which in the beginning were opposing it), it was clear that all requirements levels need to be managed in the RM infrastructure.

The RM infrastructure deployment happened in 3 phases:
1) DOORS was introduced as the RM tool under the responsibility of the second author.
2) Defining the RM process and customizing the RM tool by the second author and the consultation of the first author, and implementing it in the project managed by the third author
3) Promoting the second author to be the requirements engineering responsible in MBT, aiming to improve the requirements processes for CMMI maturity level 5.

B. Requirements Levels

A major issue in defining the requirements management process in MBT was to define the requirements management levels in accordance to system architecture, rather than refer to the standard documents names SSS, SSDD. In a system of systems there exist a number of SSSs and SSDDs on different levels of system architecture. Even the word 'system' had to be better identified at the different requirements management leveling, since often a subsystem is managed as a project by itself and referred to as a 'system'. The objective was to determine the coverage from contract requirements, to systems specification, to R&D and testing, and to identify inconsistencies between them.

The main requirements levels in MBT projects are R1 to R5 described in Figure 1; R1 refers to the customer requirements containing the contract documents, including detailed technical annexes, R2 refers to the system specifications document (SSS), R3 and R4 refer to system and subsystems design specifications (SSDD) and R5 refers to the software and hardware components specifications (SRS).

C. Requirements Traceability

Requirements levels and traceability, is typical in a product-based company. The requirements traceability allows systematic analysis of requirements compliance, discovering inconsistencies between the different levels of the requirements and requirement gaps (missing requirements). These traces are based on a many to many relationship between the levels.

The RM traceability scheme described in Figure 1 is the backbone of the product family projects development process, starting from customer contract requirements allocated and traced, to system technical specification needed by software engineers for the system design and architecture, through subsystems specifications to R&D software and hardware components specifications.

Using advanced RM tools based on database enabled us to create and manage the traditional product management documentation (SSS, SSDD, SRS). In addition, the RM process schema served as a vehicle to produce different project management deliverables that did not exist beforehand such as:

- Difference between projects in terms of contract requirements and impact analysis,
- Simulation specifications traceable to contract,
- Automatic generation of documentation,
- Experiments documentation derived from and traceable to whole project software according to the experiments phases, enabling actual reuse of the software components.

IV. REUSE OF REQUIREMENTS

Multi-projects RM process was the biggest challenge in implementing DOORS in MBT. The project that we started with already had other derivatives to additional customers. It was obvious that generic sub-systems will be reused in this project, which meant reuse of their specification (with some evolution of requirements). The only question was in what level of RM the reuse will occur.

The contract requirements at R1 level had to be specific per each customer, and they were captured into DOORS, see Figure 2 Level R1. The contract managers quickly learned the benefit of accessible requirements, and preferred to use a copy of a similar contract and then rewrite it to the specific customer. This enabled the generation of a difference report between the 2 customers, which was valuable for the project
managers in their schedule and budget estimation in early stage of project lifecycle.

A. Common Terminology

The same system engineering group usually supports all projects in the same product family. Therefore, the knowledge sharing of system engineers of projects in the same product family is obvious and valuable towards reuse. By having the same people with the same terminology, it was easy to identify that subsystems in levels R4 and R5 are technically generic building blocks that are traditionally reused from project to project in an opportunistic way rather than in a systematic way of reuse [15].

At the beginning of a project, an analysis is done with the senior system engineer to determine the product tree of the new project based on the planned architecture. During the analysis of a third project in the same product family, we realized that there is a difference in terms between the projects as a result of different problem domains. In the existing projects of the family a major subsystem name in level R3 was SSA while in the new project it was called SSAB. Fortunately it was early in the project lifecycle, and the senior system engineer (the third author) decided to use the existing terminology SSA as a basic step toward reusability, see Figure 2 Level R3. This project had very tight deadlines, and any chance for reuse was appreciated very much. Subsequently, a major reuse of algorithms of this subsystem actually happened, basically because the system engineers served in the 2 projects and proactively looked into requirements reuse options in all components specifications relevant to SSA.

The fact that two projects are from different domain problems is reflected in the contract that should be specific to each customer (R1), and in the system specifications (R2) that provides the solution to that specific problem domain. However, any subsystem that technically is generic enough to be incorporated into all projects in a family, can carry the same internal terminology in order to be identified as a generic building block and be reused. Once this generic building block is recognized, the reuse of requirements is straight forward leading to actual reuse of components.

B. Generic Components

Once the system engineers decided to look into requirements reuse of projects in the same product family, it was obvious that the SSA major subsystem, which is a system by itself, will be reused. Thus, systematic workflow of reusing all the SSA development artifacts SSDDs was implemented. Eventually most of the subsystems were reused, incorporating requirements evolution while keeping compatibility between projects in the same product family.

Noticeably, in one project, 50% of algorithms were reused, enabling 100% reuse of software SRS, incorporation requirements evolution while keeping compatibility between projects in the same product family, see Figure 2 Levels R4 and R5.

Figure 2. Requirements reuse in product family projects

V. REQUIREMENTS MANAGEMENT EVALUATION

In the following we provide an evaluation of the efficacy of the requirements management process that supported the requirements reuse, showing requirements metrics taken from the RM tool database and analyzing them.

A. Metrics

At the end of 2007 the requirements management process at MBT was evaluated and the results are listed below:

- DOORS was used by 3 projects in a product family, 2 projects in production phase and one before contract signing.
- 30 engineers were using the RM tool for assessing differences of projects in production, aiming for reuse.
- More than 70% common requirements found in 5 of 8 major subsystems (62% of project), making reuse option apparent.

At the end of 2009, a further evaluation was performed. Due to the large amount of data, only projects that had been updated in 2009 were reviewed, focusing on projects of a product family. In addition, DOORS became an operational tool by the projects engineers used for creating requirements, reviewing requirements changes, generating documents, etc., thus resulting in a different set of metrics. The results are listed below: first are presented the results referring to the requirements use in DOORS, followed by those referring to requirements reuse in DOORS.

- Approximately 30% of all specifications documents released during the year were directly related to the requirements process via DOORS.
- 120 employees were using DOORS for their tasks related to requirements.
- 8 projects in different stage of development were actively using DOORS.
- More than 120 modules of specifications documents were written via DOORS during 2008-9.
- 4 projects in a family product are reusing requirements.
• Between 60% to 90% of requirements were reused in the major subsystem of a product family.
• Between 30% to 55% of algorithms SSDD reused in a major subsystem.
• 100% reuse of software components SRS, implying actual reuse of existing software components.
• About 70% of requirements were used as-is per reused SRS.

B. Analysis

There has been a marked improvement in requirements management process over time. In 2007, just 3 projects were using DOORS, mainly for assessing requirements differences between projects, whereas in 2009 projects were following the RM process using DOORS as a desktop operational RM tool. This is an increase in the number of projects using the process of 260% over less than 2 years. A 400% increase in number of employees using DOORS reveals the fact that the RM process and infrastructure became operational in MBT projects.

At the end of 2009, 4 projects in a product family were using their requirements in DOORS according to the requirements management process required for maturity level 5. This means that, on average, each one of the 4 reused projects saved the development effort of 45% of other projects subsystems.

Managing the algorithms specifications on the right requirements level enabled the identification for reuse of more than 50% of algorithms resulting in 100% of software components SRS reuse in a major subsystem. This means that, on the average, each of the related software components was reused keeping compliance between projects in the same product family.

Reuse of contract requirements to compare differences between customer requirements was done in about 2 days, providing critical data for project managers for schedule and budget estimation before the sign-on of the contract.

Developing software intensive system in MBT usually takes 5 to 10 years to accomplish full delivery, in which software development is 75%. Based on results of software requirements reuse presented above, the time to develop the software components in successive projects in the same projects family, reduced development time to 33%. Thus, in a family of projects that takes five years to develop each project, the reuse of requirements of the first project reduced time to market of each successive projects in the family to 60%.

VI. CONCLUSION

The MBT daily use of the RM process enables visibility of projects in product family, making requirements commonality apparent. Dynamic and multi faceted organization and tracking of product family requirements is a very powerful working asset used by all project stakeholders which leads to defining and prioritizing requirements, tracking development, testing and acceptance.

Over the last 2 years, MBT is already reaping the benefits of the RM process and infrastructure. Senior management and system engineering are now focusing on full-scale product family deployment, over the breadth and depth of the company.

Requirements have become the backbone of the whole development process, in system, subsystems, algorithms and components development. Requirements are now tightly coupled with design and testing.

As a result of orderly RM process deployment, requirements reuse is obvious and requirements change control is done online, because the infrastructure is there to support such activities. We are at the point where the reuse of requirements as part of the requirements management process start to show a Return on Investment.

ACKNOWLEDGMENT

Many thanks to Lev Avigdor, Dellus Francois and Zadka Ezra. Without their efforts over the years, the Requirements Management project would not have been possible.

Special thanks to Boaz Levi, Director of Air and Missiles Defense Systems at MBT, who was the driving force behind the Requirements Management Process. His tenacity and ownership kept the process on track and continually moving forward.

REFERENCES

[1] Capability Maturity Model Integration (CMMI), http://www.sei.cmu.edu/cmmi/
Leah Goldin, CEO of Golden Solutions, is an independent consultant specializing in software engineering, process, and quality. She has accumulated over 20 years of experience developing embedded systems. During that period she has fulfilled various management and technical roles, including software development, SQA and process improvement, at Rafael, IAI, MBT, Converse, Nice, etc. Dr. Goldin currently divides her time between consulting to high-tech companies and teaching in academia; she is the Head of the Software Engineering Department at Shenkar College of Engineering and Design. Dr. Goldin received her Ph.D. in Computer Science from the Technion, Israel, where her research focused on Requirements Engineering. She is a senior member of IEEE, and currently serves as the Chair of the Israeli Chapter of the IEEE Computer Society.

Michal Matalon beck is the Requirements Engineering Manager at MBT Missiles Division, Systems Missiles and Space Group at IAI Ltd., Israel, where she has been since 1984, holding a variety of positions mainly in simulation and HWIL development. While serving as Requirements Engineering Manager, she has implemented and developed the use of DB tools for requirements engineering, work process and procedures of system engineering. Her B.Sc degree is in Industrial Engineering science from the Ben-Gurion University, Israel, with emphasis on operation research, simulations, decision making and statistic methods.

Judith Lapid-Maoz Program Systems Engineer at the Air and Missiles Defense Systems program, MBT Missiles Division, Systems Missiles and Space Group. She received her B.Sc and M.Sc in Aerospace Engineering from the Technion, Israel Institute of Technology, Haifa, Israel. She also holds an ME in Systems Engineering from the Technion. She was working as a project officer for various R&D programs for the Israeli Defense Force. Currently, she works as a senior system engineer in a multi disciplinary program at IAI.