UQO

UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS (UQO)

Doctorat en sciences et technologies de l'information

THESE

Étude sur l'utilisation de méthodes agiles dans le portage automatique de schémas de projets de développement de circuits intégrés

Study on the Use of Agile Methods in Automatic Layout Porting of Integrated Circuits Development Projects

Par

EBRU DALBUDAK

Professeurs:

Ahmed Lakhssassi, Directeur, Département d'informatique et d'ingénierie, UQO

Karim El Guemhioui, codirecteur, Département d'informatique et d'ingénierie, UQO

Table of Contents

1. RESUME DE THESE

Utilisation des méthodes agiles dans les projets de développement d'outils EDA (Electronic Design Automation)

Les compagnies de fabrication de semi-conducteurs font face à de nombreux défis tels que la complexité accrue dans la conception des circuits intégrés (IC), le besoin croissant de plus grandes vitesses et capacités, ainsi qu'une pression continue de livrer rapidement des designs compliqués. En raison de l'accroissement de la complexité des designs et des délais de livraison de plus en plus serrés, les outils EDA jouent un rôle critique dans la conception et le développement des semi-conducteurs.

Une des principales contributions de cette thèse est l'élaboration d'un cadre (Framework) d'évaluation méthodologique pour permettre aux développeurs et gestionnaires de projets de décider entre une approche agile, une approche traditionnelle (planifiée) ou une approche hybride, pour le développement d'outils logiciels de type EDA (Electronic Design Automation). La méthodologie proposée repose sur une analyse multicritères poussée et fait appel à des méthodes du domaine de la gestion.

Le cadre d'évaluation présenté, nommé Methodology Assessment Framework (MAF), constitue un outil d'aide aux décideurs pour choisir la méthodologie de développement logiciel la plus appropriée pour leurs projets. L'outil utilise 7 facteurs décisionnels qui sont les résultats, la portée, la complexité (CYNEFIN), les composants, les principes agiles, les connaissances et expériences de l'équipe de développement, et finalement la capacité et maturité organisationnelles. Chacun des 7 facteurs est explicitement détaillé, avec des métriques pour évaluer la pertinence d'utiliser une approche agile ou planifiée dans le développement de projets logiciels.

Une étude de cas est ensuite présentée qui montre la mise en œuvre du framework MAF pour déterminer l'approche logicielle la plus adaptée au développement d'un outil de type EDA dans le cadre du projet OPIC (Outil de portabilité de circuits intégrés entre procédés technologiques).

2. THESIS STATEMENT

Semiconductor process technology has been advancing and evolving at a tremendous pace. According to World Semiconductor Trade Statistics (WSTS), an independent non-profit organization representing the vast majority of the world semiconductor industry, and the Semiconductor Industry Association (SIA), representing U.S. leadership in semiconductor manufacturing, design, and research, the worldwide semiconductor market was up from US$ 408 billion (in 2017) to US$ 412.1 billion in 2019 and the year 2020 is forecasted to be stronger.

Since the Advent of Integrated circuits (IC) in the 1960s, semiconductor industry has achieved a phenomenal growth in circuit miniaturization and transistor count per IC has increased from few thousands to billions. Latest 32-core AMD Epyc IC has 19.2 billion transistors embedded in an area of 14nm. IC design rule checks (DRC) and process technology are also evolving at the same pace with IC.

In this fast-paced environment, semiconductor companies are faced with numerous challenges and difficulties such as the increased complexity in Integrated Circuit (IC) designs with multi-billion gate chips; the growing need for increased capacity, speed and capabilities as well as the constant pressure to rapidly deliver these complex designs. On top of that, they need to compete against each other to reduce time-to-market and product development costs, while increasing product features and quality.

With the increased design complexity and tighter time-to-market schedules, design productivity is emerging as a key area of differentiation. In order to maximize their IC design productivity, the semiconductor companies look for ways to improve their capability to rapidly design complex ICs with billions of transistors. According to the International Technology Roadmap for Semiconductors (ITRS), design productivity is at the top of the list for the challenges facing the IC design both in the near term (within 3 years) and in the long term (>3 years).

It is very important to ensure that the IC design and development follows right processes and is carried out in a planned and controlled manner. In today's highly competitive IC development environment, companies are faced with making difficult design decisions that involve trade-offs among such parameters as design time, cost, power dissipation, and performance. Making the "best trade-off decision is a complex, situation-dependent process – but indeed the notion of *trade-off* (power vs. area vs. speed; solution quality vs. runtime; etc.) is at the core of design technology" (Bryant, Cheng et al. 2001).

Moreover, globalization has forced companies to move manufacturing and even chip design and software off-shore to remain competitive. By outsourcing the fabrication of their semiconductor chips to specialized manufacturers called semiconductor foundries across the globe (in other words by going "fables"), companies get to save a lot of money. Because, building and maintaining their own manufacturing plants with the latest technology would require multi-billion-dollar investments, instead they invest this money into research and development of new technologies while still maintaining high production volumes. They also get to enjoy the flexibility of choosing the foundry (or multiple foundries) that offers the best fabrication processes and options to meet their clients' needs.

The trend toward outsourcing of chip manufacturing to dedicated silicon foundries (in order words, going "fabless") and the increase in the use of multiple fabs brings forward the need to migrate existing designs between process technologies and within internal and external foundry processes (Cadence 2008).

As a response to these challenges, to deliver better results in a shorter time-to-market, semiconductor companies are turning to design reuse by mixing and matching pre-designed and best-in-class functional blocks (i.e. Intellectual Property (IP)) with less engineering resources (Systems 2000).

There are many factors that contribute to the success of IC projects. Through a review of the literature, this research aims to produce a synthesis by integrating various factors shown in Figure 1: Factors influencing Success in IC Projects to guide an organization towards successful management of IC development projects.

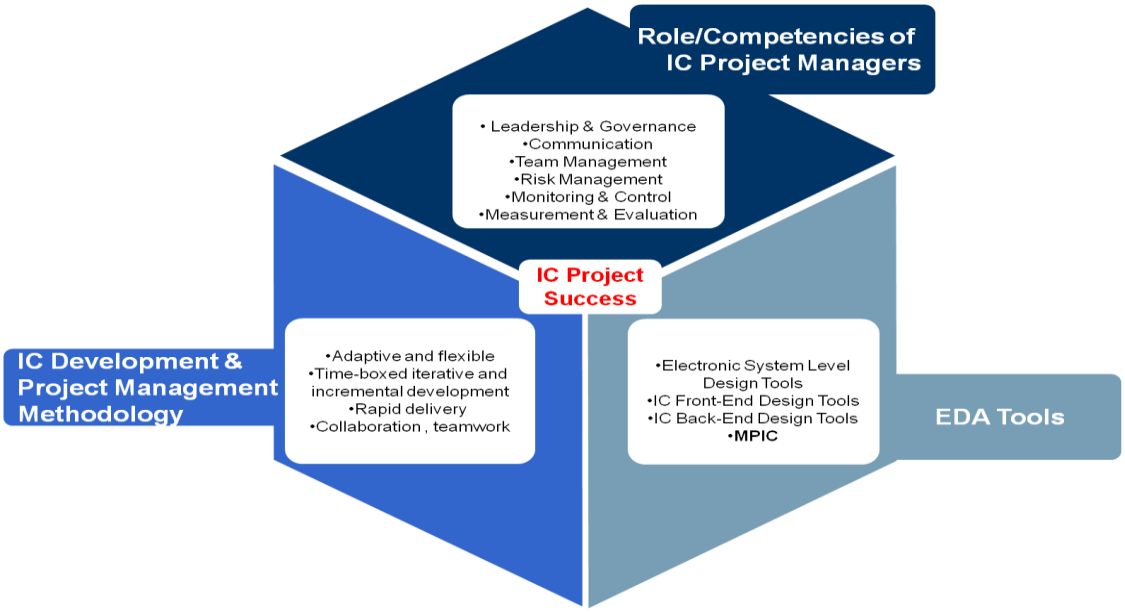


Figure 1: Factors influencing Success in IC Projects

One of the important factors that influence the successful delivery of IC projects is the *role/competencies of IC project managers*. Project managers lead the coordination and integration of the project activities to ensure the successful delivery of the desired outcomes. Another important factor is *IC Development & Project Management Methodology*. Making an informed development and project management methodology decision is the first step of every project. Mansor, Yahya, Arshad argue that "choosing the right method in software development methodology will determine the success of software development" (Mansor, Yahya et al. 2011). This research introduces a new framework called Methodology Assessment Framework (MAF) that will provide a systematic and analytical way of understanding, comparing and evaluating software development methodologies to help determine whether an agile or a traditional/plan-driven methodology should be used to deliver a development project successfully. The use of agile methods in hardware development projects is not as common as in software development projects (Johnson 2013). This research aims to assess the use of *Agile Project Management Methodology* in the IC development projects. Another very important factor that contributes to the success of IC projects is the *Electronic Design Automation (EDA) tools.* According to Bryant et al. "The quality of the design tools and associated methodologies determine the design time, performance, cost and correctness of the final system product" (Bryant, Cheng et al. 2001). Innovation is a great challenge in the EDA industry and to provide faster, better, and cheaper products, the major players in the industry focus on improving existing tools and methodologies (Sangiovanni-Vincentelli 2003).

As a part of this research, two innovative solutions will also be proposed to improve IC design productivity; 1) a new IC design model for IP porting from one technology to another by carrying out schematics and layout porting in parallel and 2) a new innovative EDA tool designed for hard-IP market that will allow replication of an existing layout in different technology nodes by automatically porting analog and mixed -signal circuits.

The project called MPIC (which stands for "**Development of new fully automated methodology for porting of integrated circuits between different technologies nodes**"), will be used as a use case for this research, first to introduce a new EDA tool called OPIC (stands for "Outil Portable pour Intégré Circuit") developed by a team of PhD students at the Université du Québec en Outaouais (UQO) for automatically porting the layout between technologies; next to apply a new framework called MAF to determine whether an agile, traditional or hybrid system development methodology would be suitable to use in this project; and finally, based on the result of the MAF assessment to demonstrate the use of an agile methodology during the implementation of this EDA tool.

The proposed tool will be able to port circuits within different foundries and technology nodes. This new automated tool will be used for porting analog and mixed signal circuits and will provide a quick and robust analysis of IP using existing layout, floorplan and routing between the blocks. The tool will preserve the key characteristics of an existing layout including matching of critical components and their relative placement. It also will generate a Layout Versus Schematic (LVS) and Design Rule Check (DRC) clean layout in the targeted technology with minimum human intervention.

The introduction of this automated porting tool (OPIC) is exclusively driven by the need in the hard IP[1] market, which is facing complex technical challenges because design rules, which tend to differ from company to company and from process to process, make porting an existing design between different processes a time-consuming task. According to Zhu et al.(Zhu, Fang et al. 2005), "manufacturing processes are updated every 18 months each with a different set of design rules", and companies need to offer different versions for different foundries. Automatic layout migration technology can amortize the high development cost of hard IPs across different foundries and processes (Zhu, Fang et al. 2005).

This project responds as well to an immediate need in the field of semiconductors and historical transition from traditional planar Complementary Metal-Oxide Semiconductor (CMOS) transistors to Fin-shaped Field Effect Transistors (FinFETs), which is pending any adequate solution. The tool will be particularly useful in a variety of different sectors, including IP providers, suppliers of IP libraries, silicon foundries, etc., which are important sectors for the Canadian economy.

As a project manager, I contribute to the MPIC project by proposing the new framework called MAF to provide a systematic and analytical way of understanding, comparing and evaluating methodologies to help determine whether an agile or a traditional/plan-driven SDM should be used to deliver the MPIC project successfully. This new framework (MAF) is based on seven factors, each of which will be evaluated via a series of self-evaluation tools. Please refer to The New MAF Conceptual Framework for more information on MAF.

The MPIC project will be implemented and managed using an agile methodology as it was identified as the more suitable methodology according to the evaluation results obtained by applying the MAF to the MPIC project. For more information on the recommended methodology, please refer to

[1] Please go to page 11 for the definition of "hard IP" versus "soft IP"

Recommended Agile Methodology for the MPIC Project and for the project team and the project milestones, please refer to ANNEX : SYNTHESE

Cette recherche a contribué en proposant trois approches innovantes qui visent à améliorer la productivité et le succès de la conception de circuits intégrés, facilitant ainsi le développement d'outils EDA (Electronic Design Automation) de conception plus efficaces réduisant le temps de mise sur le marché.

- Le processus de migration des IP (Intellectual Property) de Circuits Intégrés (CI) d'un nœud technologique à un autre nécessite de nombreuses tâches manuelles et interactives répétées. Par conséquent, un support d'outils pour rendre ce processus plus efficace est nécessaire. A l'heure actuelle, il n'existe aucun outil connu qui automatise la migration ou portage du ''Layout'' d'un nœud technologique à un autre pour les circuits analogiques et mixtes. Cette recherche a proposé et introduit un nouvel outil EDA pour la migration des IP en générant son *Layout* dans la technologie cible.

- En plus d'un nouvel outil EDA, la recherche a également proposé un nouveau modèle de conception plus agile pour les projets de nouveaux CI. En utilisant ce modèle de portage des CI proposé, les phases de conception schématique et de *Layout* pourraient progresser en parallèle, au lieu de séquentiellement l'une après l'autre, ce qui augmenterait la productivité de la conception.

- Le thème « Quelle méthodologie de développement de système (SDM) devrions-nous utiliser? » est l'une des premières décisions à prendre pour la mise en œuvre des projets. Après avoir systématiquement examiné la littérature académique sur les SDM disponibles, étant donné que les cadres d'évaluation et les outils de comparaison existants ne répondent pas à tous les besoins des chefs de projet, cette recherche a introduit un nouveau cadre appelé MAF pour aider à décider de la SDM la mieux adaptée pour un projet donné. Ce cadre a identifié 7 éléments qui contribuent au choix d'une SDM appropriée pour un projet.

Les chefs de projet doivent sélectionner la SDM la plus appropriée pour leurs projets. La sélection et la mise en œuvre d'une SDM appropriée sont cruciales car elles maximisent les chances de succès du projet. Décider d'utiliser une méthodologie agile, planifiée ou hybride dans un projet n'est pas évident, mais cela nécessite des réponses honnêtes à des questions difficiles et du courage pour prendre la bonne décision. L'évaluation de chacun de ces 7 éléments est subjective et sera influencée par le chef de projet et/ou les décideurs. La façon dont ils perçoivent et réagissent aux complexités est davantage une considération individuelle et interactive que ne le représente la littérature actuelle. Il n'y aura jamais de méthode parfaite

car tous les projets sont différents, mais pour la majorité, il existe une applroche la mieux adaptée. Le succès consiste à faire les bons choix.

Il serait intéressant de savoir si le MAF est un cadre fiable pouvant être utilisé comme un outil précis d'évaluation et de validation. Même si l'application pratique du MAF a démontré qu'il fournissait une structure pour évaluer et valider un projet afin de déterminer une SDM appropriée, les éléments du cadre pourraient faire l'objet d'une analyse plus approfondie. La définition et l'utilisation de paramètres quantitatifs pour chacun des sept facteurs (par opposition aux paramètres qualitatifs) feraient du MAF un cadre d'évaluation plus complet.

Des recherches supplémentaires pourraient être utiles pour améliorer le MAF. Cette recherche bénéficierait d'une étude empirique pour affiner et mettre à jour le MAF proposé et l'appliquer à l'évaluation de divers projets de taille et de complexité différentes. Grâce à un retour d'expérience empirique à partir de situations réelles de projet, le critère d'évaluation et les métriques utilisées peuvent être améliorés afin de fournir des résultats plus précis.

Le choix de SDM doit être accompagné d'une approche de gestion de projet appropriée. Cette recherche a porté sur l'utilisation de la méthodologie de développement agile avec la méthodologie de gestion de projet agile. En utilisant le projet OPIC comme étude de cas et en évaluant le projet en utilisant le nouveau cadre MAF, et en se basant sur la revue de la littérature qui a démontré les avantages attendus de l'utilisation d'une méthodologie de gestion de projet agile dans la conception de logiciels, de matériel et dans le co-développement de projets impliquant matériel et logiciel, cette recherche a abouti à la recommandation d'adopter une méthodologie de gestion de projet agile hybride (combinaison de XP et SCRUM) dans les projets de développement et portage de circuits intégrés.

L'analyse pourrait également être étendue pour couvrir une étude d'utilisation du nouvel outil EDA appelé OPIC dans différents projets de développement de circuits intégrés.

Appendix A: The MPIC Project Team And Project Milestones.

3. LITERATURE REVIEW

3.1. Challenges of IC Design

Each generation of IC design process introduces new improvements by reducing the size of the transistors and wires while increasing the number of transistors and wires that can fit on a chip. IC design complexity has been exponentially increasing since the appearance of transistors in the 1960s.

Over the past 55 years, the IC industry has been driven by Moore's Law, which predicted that the transistor densities double every 18-24 months (Birnbaum 2004; Jansen 2003). Since the appearance of transistors in 1960s, transistor counts[2] of integrated circuits have increased from tens to billions. As a matter of fact, while as of 2014, the highest transistor count on a chip was around 4.3 billion transistors, as of 2016, the largest transistor count in a commercially available CPU reached over 7.2 billion (the Intel Broadwell-EP Xeon) and in 2019, Cerebras announced the largest chip to date called Wafer Scale Engine (WSE) with 1.2 trillion transistors.

The difference between having tens of transistors versus billions of transistors is in complexity, methodology and business models (Jansen 2003).

- *Complexity* – There has been a significant increase in the complexity of IC designs over the years. Due to the advancements in the semiconductor industry, the transistors continue to shrink in size, which means we get more transistors per die area. As the number of transistors per die and the functional IP blocks being integrated together increase, the design complexity increases (Moretti 2014).

  The number of lines in the design rule check deck files has increased in each new technology. While 250nm had 5,300 lines in the design rule check deck file, 130nm had 13,500 lines, 90nm had 38,400 lines and 65nm: 89,300 lines. IC designs have become increasingly more complex also because of the increase in the number and the complexity of foundry design rules from one technology node to the next (Moretti 2014). For every new technology node, special design rules are independently developed by foundries. Semiconductor market is now facing enormous challenges in terms of complex design rules which are not compatible with previous generations. Moreover, IC designs are getting more complex also due to double patterning technology, Design for Manufacturability (DFM) and Design for Power (DFP). The

[2] Wikipedia defines "Transistor count of a device" as "the number of transistors in the device", http://en.wikipedia.org/wiki/Transistor_count

increased complexity of IC designs could not be handled without automation tools (Birnbaum 2004; Jansen 2003).

- *Methodology* - The methodology used in the production of these complex designs also evolved; the design process has been decomposed into many tasks, such as requirements gathering, specification generation, component selection, architectural design, component synthesis, physical design, verification, simulation, prototyping and manufacturing.
- *Business Models* - "The business model has changed from vertical integration, in which one company did all the tasks from product specification to manufacturing, to a globally distributed business model, in which most of the design and manufacturing tasks are outsourced" (Jansen 2003). Today, most of the semiconductor companies operate using a fabless business model and manufacture their chips using foundries located around the world. This way, they can benefit from lower capital costs while focusing on research and development. However, owning a fab no longer offers the competitive advantage it once did, as fabless chip houses have just as much access to leading edge fabrication technology as vertically integrated device manufacturers (Systems 2000).

Nowadays, the key differentiating factors for the semiconductor companies are their design capability; productivity of their engineering workforces and time-to-market (Systems 2000). We see design re-use as an emerging key area of differentiation among companies competing in the electronics industry to increase the design productivity and to gain competitive advantage (Francken and Gielen 1999; Systems 2000).

3.2. IC Design and Development Process

IC development projects consist of a combination of hardware and software development components because the chip functionality is not only determined by the hardware, but also by the software implemented on the chip. Even though having a combination of hardware and software development components in IC projects increases the complexity of development, this allows for the design and development of functionalities, which would not be feasible with hardware alone due to their complexity (Jansen 2003).

On the hardware side, we have complex IC hardware designs with millions of gates, each with sophisticated interactions and paths that need to be modeled and tested. On the software side, we are

looking at many components and layers that are interdependent and connected. Building software is an inherently complex process (Xia and Lee 2005).

Moreover, at the beginning of the design cycle, the partitioning of what needs to be done in hardware versus what needs to be done in software has to be identified. This partitioning task should be done in such a way that the hardware and software tasks can be decoupled from each other in order to be able to carry out both of these tasks in parallel and as independently as possible (Jansen 2003).

In addition to these, IC development projects need to adapt and respond to changing conditions that result from shorter product lifecycles, advancements in technology and changes in customer requirements. Therefore, they exhibit the characteristics of complex adaptive systems.

Design of an IC can be an extremely difficult task and it requires a structured approach (Birnbaum 2004; Jansen 2003). The development starts with the specifications of the IC (behavioural representation of **what** the chip does) and it will be completed once the geometrical design (layout capturing the geometry of **where** the elements such as gates, transistors, resistors, etc. will be located) that is composed of a set of polygons is ready to be sent to manufacturing. As depicted in Figure 2: IC Development Process, IC development goes through several stages: IC Specifications, Electronic System Level (ESL) Design, Register Transfer Level (RTL) Design, Logical Gate Level Design, Transistor Level Design, and finally IC Fabrication.

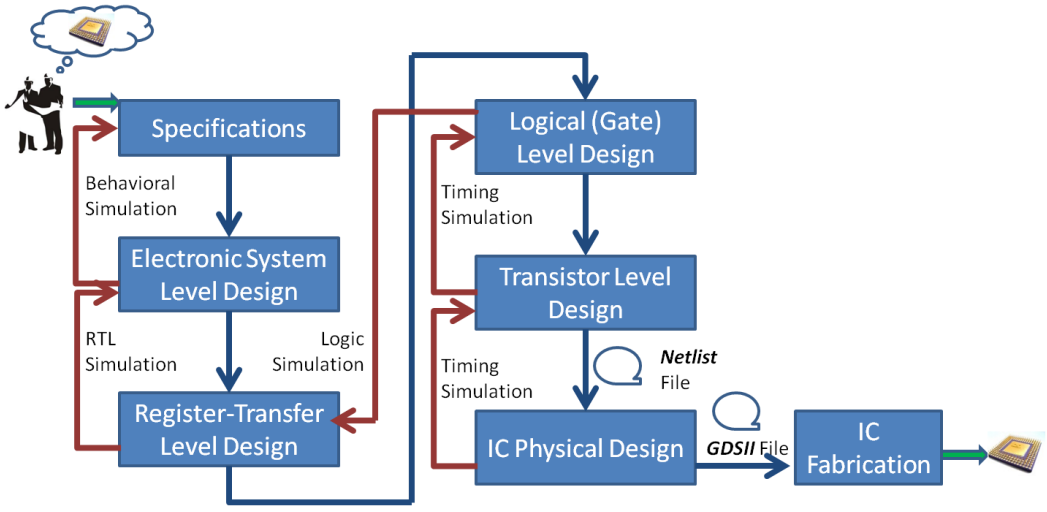


Figure 2: IC Development Process

The set of IC requirements are created by IC system engineers. The design specifications describe a set of functionalities that the IC is expected to provide and a set of constraints that it must satisfy (Bertacco

2003). This description captures the logical (functional) behaviour, electrical parameters (e.g. clock frequencies, timing behaviour, etc.) and environmental conditions (e.g. temperature, supply voltages, etc.).

The ESL design is the initial process of deriving a potential and realizable solution from the design specifications and requirements. This is sometimes referred to as modeling and includes such activities as hardware/software tradeoffs and a micro-architecture design. From the system level design model, the hardware design team proceeds to the RTL design phase. During this phase, the architectural description is further refined: memory element and functional components of each model are designed using a Hardware Description Languages (HDL), such as Verilog or VHDL. This phase also sees the development of the clocking system of the design and architectural trade-offs such as speed/power.

After the RTL hardware description is taken through the Logical (Gate) Level design, the detailed model that describes the design in terms of its basic logic components, such as AND, OR, NOT or XOR and memory elements is produced. After the Transistor Level Design, a technology dependent data file called *Netlist* is created; this file describes the logical composition of the chip in various groups of transistors, or gates, flip-flops, registers and their interconnections (nets) to be built on the chip.

"Optimizing the *netlist* or gate-level description for constraints such as timing and power requirements is an increasingly challenging activity for current developments and it usually involves multiple iterations of trial-and-error attempts before it converges to a solution that satisfies both these requirements" (Bertacco 2003).

Logic design may be performed manually or using logic synthesis tools. In either case, the design will probably go through several refinement steps before completion. Initial design verification steps will concentrate on logical correctness and basic timing properties. Once the basic structure of the logic has taken shape, scan registers can be inserted and power consumption can be analyzed. A more detailed set of timing checks can also be performed, including delay, clock skew, and setup/hold times.

Next, IC layout designers take this design plan (i.e. the *netlist* file) and create a layout of the physical representation of the transistors and gates with floorplanning, placement and routing. Physical design is one of the most critical steps in the IC design as the placement and routing will impact the area, power and performance of the final design. Each semiconductor technology has its own set of rules that a physical design has to comply with in regards to the *width* and the *distance* between the polygons.

Physical design starts with floorplanning to determine the overall structure of the layout. If the logic was designed in large blocks, it may be necessary to partition those large blocks into smaller pieces at this point. Placement and routing will determine the geometric placement of each of the blocks and their interconnects on the silicon area in a way that this area is optimized. Once the layout is complete, the wiring parasitics[3] must be extracted and back-annotated to the logic design. The back-annotated design can then be simulated to verify that layout did not violate any timing.

Large chips are often designed with IP blocks. IP-based design offers advantages over reducing design time; the blocks have been functionally verified; and they provide more accurate estimates of area, speed, and power early in the design process. Hard IP blocks are complete designs, and provide accurate information about area, delay and power as soon as they are selected but they are also harder to port to new processes and harder to modify to meet variations in customer requirements. The development cost of hard IPs is very high due to their custom layout design across different foundries and processes (Zhu, Fang et al. 2005). Soft IPs, on the other hand, are not as well-characterized but can be more easily adapted to changing needs (Wolf 2002).

At the end of the physical design stage, a Graphics Data System II (*GDSII)* file, which is the de facto industry standard for data exchange of IC design layout capturing the hierarchical representation of the integrated circuit in planar geometric shapes, text labels, and other information, gets generated. This file is then sent to the IC fabrication plant to produce the chip.

If a chip is a rework of an existing design – a design shrink, a few added features, etc. – then the architectural design is simple. But when designing something new, a great deal of work is required to transform requirements into a detailed micro architecture ready for logic design. Architectural design requires extensive debugging for both functionality and performance; errors that are allowed to slip through this phase are much more expensive to fix later in the design process.

The ultimate goal of IC design and development is to design layouts for circuits. Layout design requires not only a knowledge of the components and rules of layout, but also strategies for designing layouts which fit together with other circuits and which have good electrical properties.

[3] a **parasitic** element is a circuit element (resistance, inductance or capacitance) that is possessed by an electrical component but which it is not desirable for it to have for its intended purpose.

Typical IC design flow requires several general steps that are illustrated in the following process diagram (Figure 3: IC Design Flow). The starting point for layout is the circuit schematic. The schematic shows all electrical connections between transistors and provides specification for implementing the transistors and connections in the layout. The design flow starts with specifications (modeling) and system level design verification. Until the desired design reaches the level of detail that can be processed by logic synthesis, it is debugged recursively with simulations such as, behavioral simulation, RTL simulation, logic simulation, gate level simulations, timing simulations, etc.

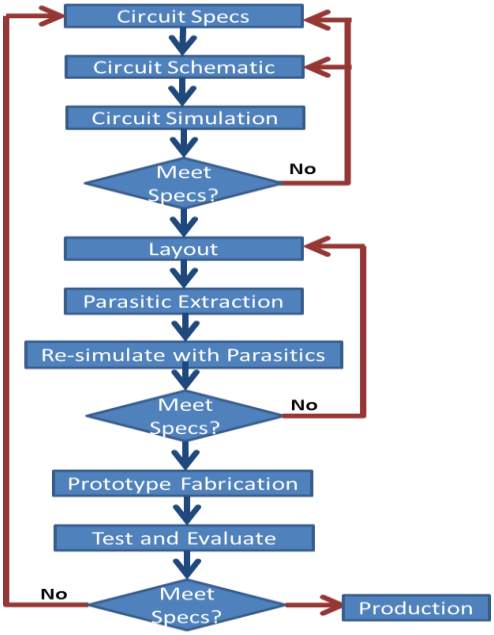


Figure 3: IC Design Flow

As the design gets more complex and process geometries get smaller, the impact of wire resistance, capacitance and inductance (aka parasitics) becomes important. Wiring forms a complex geometry that introduces capacitive, resistive and inductive parasitics. This impacts on delay, energy consumption, and power distribution. The goal in IC design is to decrease the resistance and capacitance (especially, the coupling capacity) as much as possible.

Several types of design checks, including, Design Rule Checking (DRC), Layout versus Schematic (LVS), electrical checking and timing analysis are followed to ensure that there are no fundamental errors before the design is sent to fabrication.

DRC checks determine if the layout satisfies a set of rules required for manufacturing. The design rules are a series of geometric and connectivity parameters provided by semiconductor manufacturers to ensure

sufficient margins to account for variability in semiconductor manufacturing processes (Mukundan 2013). All the metal layers as per the technology have to follow certain rules. Three of the most important design rules are:

- Minimum width of metal layer
- Minimum spacing between the metal layers
- Minimum overlap between different layers

The width, spacing and thickness dimensions of metals vary as per the technology and as we go down in the technology (for example from 90nm technology to 65nm technology), these dimensions decrease continuously (VLSIExpert 2012). These dimensions are critical and if they fall below specified values, the probability of manufacturing errors rises. So, prior to fabrication, design rules must be checked to ensure that the IC complies with these rules. If the design rules are violated, the chip may not be functional.

An input to the design rule tool is a 'design rule file'. A successful DRC ensures that the layout conforms to the rules designed/required for faultless fabrication. However, it does not guarantee that it really represents the circuit we desire to fabricate. This is where an LVS check is used. LVS is another major check in the physical verification stage. It verifies that the layout that was created is functionally the same as the schematic/netlist of the design (Mukundan 2013).

The LVS tool creates a layout netlist, by extracting the geometries. This layout netlist is compared with the schematic netlist. The tool may require some steps to create either of these netlists. If the two netlists match, we get an LVS clean result. Otherwise, the tool reports the mismatch as well as the component and location of the mismatch. Along with formal verification, which checks if the pre-layout netlist matches the post-layout netlist, LVS verifies the correctness of the layout with respect to intended functionality (Mukundan 2013). Some of the LVS errors are:

- Shorts – Wires that should not be connected are overlapping.
- Opens – Connections that are not complete for certain nets.
- Parameter mismatch – LVS also checks for parameter mismatches; e.g. it may match a resistor in both layout and schematic, but the resistor values may be different. This will be reported as a parameter mismatch.
- Unbound pins – If the pins don't have a geometry, but all the connection to the net are made, an unbound pin is reported.

3.3. IC Layout migration

The fast moving and highly competitive microelectronics markets require "increased levels of integration, frequent upgrades, and technology shrinks" (Macintosh 2014). As the complexity of IC designs increase, to meet design deadlines with limited design engineering resources and to maximize their design investments, semiconductor companies turn to design reuse by mixing and matching pre-designed and best-in-class functional blocks and hard IPs from previous projects whenever possible, instead of designing everything from scratch (Francken and Gielen 1999). One effective method to reuse existing circuits is the layout to layout migration. "Migrating a layout means adapting it to newer process design rules. This method does not require a major change in the design methodology and it is an easier way to implement reuse methodology" (jain 1999).

Layout designs are migrated for various reasons: for design reuse in system-on-a-chip applications, for shorter time to market, for the second-source production which is popular among fabless semiconductor companies that try not to be dependent on one silicon vendor, for cost reduction with a smaller die by a process with smaller minimum feature size, and for performance improvement and power reduction (Choi, Chun et al. 2004). Therefore, there is a strong need for fast automatic methods of technology migration for layout.

Porting successful designs from one technology to another can distinctly reduce the design cycle. Instead of spending time and effort to completely redesign an existing circuit in another technology process, semiconductor companies can work on developing new parts of the system to add extra functionality and/or increase performance (Francken and Gielen 1999).

With the trend to go fabless, many chip designs have to be migrated between technologies and within external and internal foundry processes by reusing past designs, IPs and circuits, instead of designing everything from scratch.

Often a new design uses a new or different process technology. Thus, it is necessary to migrate and apply a given circuit IP for use with this new technology. Frequently, the different foundries are not fully aligned at the same process technology node, so designs are often alternated between foundries and processes. Often, we need to migrate an existing design from one foundry to another.

The aim of layout porting between technological process nodes is to be able to replicate an existing layout (reference design) in different technology nodes by modifying or adapting it to create multiple copies.

While doing that, the ultimate goal is to achieve high quality layout with high integration and low power and also to shorten the design-time-to-market.

Porting designs efficiently and correctly between process technologies is a challenge. Design migration and IP porting process includes two different types of tasks (Dornelas, Schmidt et al. ; MunEDA 2014): **Horizontal porting** and **Vertical porting**. The former is about migrating IP from one technology node to the same node of a different foundry due to foundry migration, second sourcing or fab consolidation while the latter is about migrating IP from a technology node to a smaller one, usually from the same fab or foundry (Dornelas, Schmidt et al.). Both are challenging, time-consuming and error-prone, especially for analog-/mixed-signal designs (AMS), Radio Frequency (RF) designs, IP libraries, and memory cells because many blocks and even entire System on Chip (SoC) must be migrated in a short time, mostly by a very limited number of designers. Furthermore, there is no simple rule for shrinking AMS/RF, Input/Output (I/O) and full-custom digital designs (Dornelas, Schmidt et al.). Every block needs adjustment of geometries, biasing, etc., even if specs don't change. Therefore, it is necessary to migrate and port the schematics individually to conform to technology constraints, or to meet enhanced functionality or performance specifications (Dornelas, Schmidt et al.).

Porting an existing design from technology A (the reference design) to technology B (the target design) usually follows a 2-step design flow. In the first phase, the given schematics and topologies are converted from the source to the target technology and the existing layout and mapping layers are migrated. In the second phase, these circuits and IPs are sized for the new target specifications and optimized for the new target process technology (MunEDA 2014; Sobe, Graupner et al. 2009).

Sobe et al. describe a porting strategy based on transferring the circuit topology followed by sizing using optimization (Sobe, Graupner et al. 2009). As shown in Figure 4: IC Layout Migration Flow, first a schematic transfer of circuit topology from technology A to technology B takes place using the Process Development Kit (PDK). Since during the conversion process the circuit performance is not considered for sizing, after the schematic conversion step, the circuit performance can be reduced or even may not work at all. Therefore, the circuit must be resized/optimized to restore its performance or to meet the new specifications. As a final step, before the layout is created, the circuit is verified to check performance over Process, Voltage and Temperature (PVT) conditions (Sobe, Graupner et al. 2009).

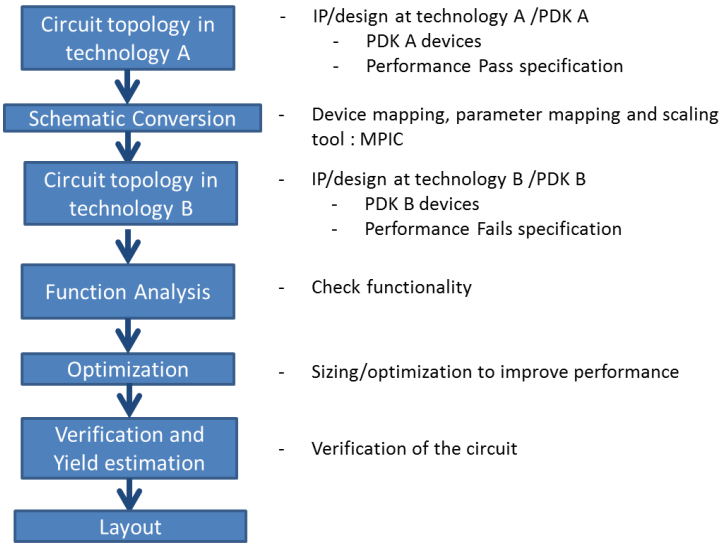


Figure 4: IC Layout Migration Flow

3.3.1 Challenges – IC Layout migration

As already mentioned, layout migration refers to porting layouts from a given technology node to a different technology node. This is achieved by first building a symbolic representation of the layout through linear constraints. The constraints relate layout shapes with respect to technology rules and connectivity information.

There are several challenges associated with layout migration. For example, migration often requires underlying circuit *netlist* changes to achieve desired electrical behavior. Typically, these changes pertain to different sizes and features for transistors and devices.

IC products usually comprise of multiple functional blocks and circuits (digital, mixed-signal and analog). The majority of today's ICs are mixed-signal designs, i.e., they consist of analog and digital circuits (blocks, partitions) (Scheible and Lienig 2015).

The mixed-signal and analog portions of these ICs are the most difficult portions of the chip to migrate between foundries and technology nodes (Macintosh 2014). As stated by Scheible, Juergen, and Jens Lienig, "in typical mixed signal ICs, the effort needed to design the analog part often matches or even exceeds the effort for the digital part by far" (Scheible and Lienig 2015). Because, "Analog designs are characterized by a much richer and more complex set of design constraints that need to be considered simultaneously and which may span several domains (e.g., electrical, electro-thermal, electro-mechanical, technological, geometrical domain)" (Scheible and Lienig 2015). According to Qian et al. porting of

analog and mixed-signal circuits are “cumbersome and resource-intensive” because “most analog and mixed-signal circuits are designed for a specific technology” (Qian, Bi et al. 2015). Because of the highly nonlinear relationship between the device parameters and circuit characteristics, most analog and mixed-signal circuits are designed for a specific technology, which greatly limits the flexibility to be reused between different process nodes (Qian, Bi et al. 2015).

In digital design, reuse of IPs has already been in use; this allows suppliers to distribute circuit blocks available on different technologies. With the use of standard cell based design methodology and advanced EDA tools, technology migration for digital circuits can be achieved by rerunning the fully automated synthesis flow (Qian, Bi et al. 2015). In analog and mixed-signal circuit designs, however, reusing designs have challenges, such as coping with trade-offs among analog specific technological requirements, such as noise, linearity, gain, supply voltage, speed, power consumption, heating, etc. (Dornelas, Schmidt et al.).

When the technology changes, some differences will follow the structure of the devices, design rules, supply voltage etc., even if the specifications and schematics are unchanged. Special resistor and capacitor layers that are available in one technology may not be available in another. According to Francken et al. (Francken and Gielen 1999), “the number of metal layers that are used for interconnection can also differ and the most restricting limitation is the problem of automatically recognizing devices in an efficiently hand-crafted layout”.

In many cases, it becomes necessary to simultaneously perform a technology shrink to achieve higher clock speeds, lower power consumption, smaller die size, and lower chip cost. Therefore, the technology migration tools that are available for digital circuits cannot be used for analog and mixed-signal circuits, and “a complete redesign is often unavoidable and tedious resizing task is inevitable” (Qian, Bi et al. 2015). The migration process involves adapting circuits to the new process by iterating layout design changes, extraction, and simulation until the desired characteristics are met (Macintosh 2014). Our challenge, therefore, is to create a design flow that would enable us to reuse the same IP blocks, shrink them without losing the circuit characteristics and place each component in the layout as compact as possible while porting them to a different technology easily and quickly.

To ensure that the migration will work under all situations, and not only for a specific circuit, the migration technique should consider the overall library architecture, such as power/ground net width, routing-track

number and port matching, as opposed to migrating a specific circuit that uses a library of cells. Another important consideration would include the treatment of IPs and the new issues for complex design rules.

To ensure that the circuit and layout level considerations of the original design are not corrupted, preservation of space and nonspace polygons has to be addressed.

A layout consists of a set of polygons, each associated with a different layer, such as metal, polysilicon or diffusion. The library of IPs typically has a predefined layout architecture. For example, all standard cells are constrained with the same height. As another example, the datapath cells have a predefined routing architecture where the data signals run horizontally and the control signals run vertically (Zhu, Fang et al. 2005). Maintaining these architecture features in addition to satisfying design rules can be another challenge, especially when the IC design rules for what can and cannot be manufactured are getting more complex. These rules determine whether the physical layout of a particular chip layout satisfies a series of recommended parameters or not. Moreover, the IC design complexity has been increasing. There are a lot of challenges associated with such difficult design migration and sizing for full-custom designs.

Porting involves iterating layout design changes, extraction, and simulation until the desired characteristics are met. In addition, if design changes are required after simulation, further manual modifications are required in the layout. During design migration, experienced designers are involved to manually readjust the device sizes to pull the design into compliance over all required conditions (Qian, Bi et al. 2015).

The goal of porting is to reuse proven functional IP blocks as much as possible instead of redesigning every block from scratch with the purposes of saving design time and of reducing the risks associated with redesigning the circuit (Qian, Bi et al. 2015). Given the size and complexity of each migration and the number of required migrations within a short period of time, traditional porting methods are too time-consuming and costly. So, the challenge therefore, is to create a design flow that would enable us to reuse the same IP blocks, shrink them without losing the circuit characteristics, and port them to a different foundry process easily and quickly.

3.3.2 Techniques for Layout migration

The goal of layout migration is to efficiently shrink and place every component in the layout as compact as possible (Fu, Chaung et al. 2009) and to convert the underlying physical layout, comprising billions of

polygons, into the new technology where they must satisfy complex geometric rules (Shaphir, Pinter et al. 2015).

There are a few different layout migration approaches available in the literature. In hard-IP reuse, the polygons are converted by compaction algorithms. In general, the reuse of IPs most commonly occurred through linear scaling process, because this has been the easiest way to convert a physical layout design to a different or new process (Shaphir, Pinter et al. 2015). Although linear scaling preserves the original shape of the layout without compromising circuit performance or design-rule correctness, compaction ratio is poor since the smallest reduction ratio is used when layers or components shrinks differently in the new technology (Choi, Chun et al. 2004). The linear scaling has become ineffective with nanometer technologies at 0.13µm and below because in new process technologies, due to various physical effects associated with the manufacturing process, each layer and object is scaled differently, reflecting its specific manufacturing tolerances and sensitivities (Zelnik 2002). Moreover, "the relationships between polygon size and the characteristics of the underlying elements, such as interconnect characteristics, threshold voltages and corresponding transistor conductance, are dramatically nonlinear", therefore a straightforward uniform shrink of feature sizes will not generate a feasible (manufacturable) layout or maintain circuit performance and behavior (Zelnik 2002).

Another method is converting layout by non-linear scaling. The non-linear porting technologies are largely classified into two categories: "polygon-based" and "object-based". Polygon-based approach relies on layout compaction of the original design with a new design rule. This technique is generally focused on maintaining the original shape optimized for flattened layout rather than for the hierarchical layout design (Choi, Chun et al. 2004).

Layout migration algorithms are classified as "constraint graph based" and "integer linear programming based" algorithms (Fu, Chaung et al. 2009). Constraint graph based algorithms usually scan all components via a virtual scan line of the design and then construct a constraint graph. New design rules are then employed on the graph to identify the position of every component with the longest path algorithm. However, the conventional constraint graph algorithm intends to produce a compact layout with numerous changes in interconnection shapes and topologies because it only considers space utilization. The resultant changes in the shape and topology degrade the timing delay and other properties. With original design intention destroyed, designers must spend time comprehending and modifying the migration layout (Fu, Chaung et al. 2009).

Another well-known research about compaction is topological layout model which focuses on the topological relation between wires and objects rather than physical geometries and the imposed design rules. A topological layout is composed of topological points and topological wires, where object polygons are regarded as topological points. A topological wire is represented by its relative positions (such as the wire next to this wire) and connection points as opposed to a physical geometrical layout, which thoroughly describes the size, shape and location of each wire and clearly identifies the design rule violation (Fu, Chaung et al. 2009).

Fu et al. (Fu, Chaung et al. 2009) introduce a topological layout migration which comprises the following four components: 1- Layout extraction, 2-Topological Layout Model Builder, 3- Constraint-graph based device migration, 4- Topology driven migration. Since the input layout is composed of polygons, the layout extraction first recognizes devices from polygons by geometrical Boolean operations. After device recognition, remainder polygons will be supposed to be wire polygons and then the centerline of wire polygons will be extracted from wire extraction stage. All polygons of the devices are sent to device migration process, which migrates devices to the target technology by a device migration algorithm. Device migration would produce original devices and migrated devices. Using the "topology constraint builder" topology constraints, that will ensure that layout topology remains consistent before and after migration (Fu, Chaung et al. 2009). Topological layout migration is then realized and finally physical wires are restored within associated space tiles to transform the topological layout into a physical layout.

Even though most of the layout migration techniques are based on layout compaction, there is also a cell-swapping based migration methodology which is used to migrate hierarchical layouts from one technology to another. This methodology uses techniques that enable retention of layout hierarchy and maintenance of parametrized cells (P-cells) and macros in the target layout. P-cells are scripts that generate layout instances for devices. In a layout database, there exists a link between instances and the master script for a P-cell. During migration a P-cell instance may have to be swapped with another instance with different parameter values (such as device width, length, multiplicity, etc.) to achieve the electrical specifications of the target design. Therefore, swapping P-cells is necessary for migration. With cell-swapping based technique, designers can selectively swap cells, retain P-cell and macro sanctity upon migration and generally retain layout hierarchy (Batterywala, Bhattacharya et al. 2008).

Shaphir et al.(Shaphir, Pinter et al. 2015) presented a hierarchy driven cell-based layout migration algorithm for hard-IP reuse by taking advantage of the natural separation of physical layout generation into placement and routing phases.

3.3.3 Automatic Porting

Literature review has revealed that even though there is an abundant research in the field of IP reuse, design migration and layout porting, the use of these techniques in analog and mixed-signal circuits is not as common. For analog and mixed-signal circuits, available commercial tools do not include at the present time features like automatic porting between different technology nodes and foundries. Consequently, very often circuit designers must adapt circuits to the new process by a time-consuming, costly and tedious manual process, which involves adapting circuits to the new process by iterating layout design changes, extraction, and simulation until the desired characteristics are met (Macintosh 2014). A lot of repeated manual and interactive tasks dominate the process to transfer the design data between technologies (Sobe, Graupner et al. 2009). On an average, to perform porting of a 28 nm chip requires 100 engineer-years of effort and costs around 30 million dollars. Double efforts and cost will be incurred for 14 nm chip. Faced with all these challenges, semiconductor companies are looking for a reliable and rapid conversion of IPs in a cost-efficient manner. Using an automated tool instead of a manual process can increase the effectiveness and efficiency of the conversion process by reducing the human resources required and by shortening the porting time while reducing the failures that might get introduced into the new layout during conversion. This also can save considerable amount of time in testing the new layout.

Note that there are a few commercially available tools from Synopsys or Cadence that can be used to check if there are any design violations. However, these tools do not take the existing layout into consideration and make changes on it; instead they regenerate the layout from scratch. This method may impact the quality of the IP as it may result in the loss of work already done in previous technology. It also means time loss as the work that was previously done cannot be reused. Moreover, none of these tools can identify all the errors that are present (e.g. density errors, oxide diffusion errors, etc.). After running the tool, a VLSI engineer has to manually check and correct the errors. Another level of checking is performance. When the DRC checking tool is run, it might say that all the design rules are respected; however, performance issues are identified only when run simulation is run. This time a VLSI engineer has to manually change the physical design to enhance the design.

There are three major steps involved in automatic porting of layouts:

1. Schematic porting, IP re-use (circuit and process migration);
2. Design assessment (topology adjustment, simulations);
3. Sizing for sign-off (circuit analysis, optimization, verification, yield estimation).

According to Payne (Payne 2016), migrating a design from one technology node (e.g., 28nm) to another one (e.g., 16nm) requires a lot of work because we have to consider the impact of process variation on the design yield, how aging and reliability affect reliability at the smaller node, and how to achieve the best Power, Performance, Area, Cost (PPAC) in the time allotted. In a pure digital world, the designers typically use logic synthesis, try some floorplanning, run some standard timing analysis and iterate until timing closure is reached. In an analog and mixed-signal circuit world, when it comes to porting cells, we need to change device sizes, adjust geometries like Metal Oxide Semiconductor (MOS) width and length, update biasing, and verify new drain supply voltage, i.e.Voltage Drain Drain (Vdd) levels

Literature review has shown that there are studies that were focused on IP reuse and technology migrations for analog and mixed-signal circuits due to their expected benefits.

Sobe et al. (Sobe, Graupner et al. 2009) presents a method, that reflects common practice of analog design divided into design data conversion and sizing by optimization, to convert and optimize a circuit topology. Fu et al. (Fu, Chaung et al. 2009) introduce a new rectangular topological layout for topology-driven cell migration. Kar et al.(Kar and Roy 1999) talk about the technology migration process and introduce a new layout preserving migration tool called: TECHMIG. Francken et al. (Francken and Gielen 1999) present a methodology for technology porting of analog circuit design from one technology process to another, considering both the sizing and the layout phase. Dornelas, Helga et al. (Dornelas, Schmidt et al.) introduce a new technology migration methodology for analog IC design and instead of doing the migration manually or at netlist level, they performed automated migration on a schematic level, followed by a robustness verification with the usage of MunEDA tools. Qian et al. (Qian, Bi et al. 2015) propose a hierarchical optimization-simulation loop based methodology which can be used for the automation of the process migration for mixed-signal circuits. Their method takes into account Process, Voltage and Temperature (PVT) variations and layout parasitic to obtain parasitic-closure design.

3.4 EDA Tools

EDA tools are gaining more importance with the continuous scaling of semiconductor devices and the growing complexities of their use in circuits and systems. Demands for lower-power, higher-reliability and more agile electronic systems raise new challenges to both design and design automation of such

systems. EDA has driven advances in semiconductor design technologies over the past 50 years and will continue to do so because of "the fast and continuing evolution of design technology and enormous growth in the complexity and sophistication" (Wang, Chang et al. 2009).

In a competitive semiconductor industry, every company wants to ensure the fastest path from product concept to consumer delivery. Under the pressure of time-to-market to deliver innovative, high quality products, the semiconductor industry is getting increasingly competitive and fast-paced. Time-to-market pressure combined with the continuous increase in design complexity can be considered as the primary driving force for the need to automate the design process with the use of EDA tools.

As Sterman has put it correctly, "when experimentation is too slow, too costly, unethical, or just plain impossible, when the consequences of decisions take months, years, or centuries to manifest, simulation becomes the main -perhaps the only- way we can discover *for ourselves* how complex systems work" (Sterman 2002).

With the increased level of complexity in designs and market pressures to produce designs quickly, the IC process has turned to the EDA tools. These tools have played a crucial role to understand the element of complexity and address IC design challenges (e.g. electrical coupling, interference on the wires, thermal problems, etc.), to handle the scalability issues, and to enable the rapid development of hardware and software systems by reducing the circuit design and development time while increasing the designer productivity. When the hardware or a device is not yet available for testing, the testing and quality analysis can be performed using EDA tools, this saves money and time. Without having to wait for and compete over hardware resources, developers can test the code they developed sooner and more rigorously and this result in an increase in the efficiency and quality of development.

Designs with billions of transistors and connecting wires cannot be made without the help of automation (Birnbaum 2004; Jansen 2003). Without these tools, the designers could not handle the IC complexity, as it would be extremely difficult and time-consuming to design an IC with billions of transistors and wires and ensure with high probability that it would work as per the specifications. EDA tools are effectively used to analyze the chips for performance design, power and thermal design, DSM physical effects and new materials under development.

The semiconductor industry trends have and will have great influence on the EDA tool landscape. “EDA tool trends include design closure, formal verification, design repair, design for test, and memory system design tools”(Birnbaum 2004). The cost of testing is increasing rapidly as the chip complexity grows.

By using EDA tools throughout the design process (system design, logic design, physical layout design), semiconductor companies can develop more complex chips with lower costs and shorter time to market (Birnbaum 2004). EDA tools not only reduce the design development time and cost of development but also reduce the risk to a project because of design and testing errors (Wang, Chang et al. 2009). Automation tools use complex algorithms on very large data sets to ensure that the designs (logic, board layout) are correct and their behavior are verified before going to the IC manufacturing, because as Birnbaum stated “correcting an error after manufacturing is very expensive and time-consuming” (Birnbaum 2004).

EDA can be viewed as a collection of design and test automation tools. Design automation tools deal with the correctness aspects of the electronic system across all levels, be it ESL, RTL, gate level, switch level or physical level. The test automation tools manage the quality aspects of the electronic system, be it defect level, test, cost, or ease of self-test and diagnosis.

Birnbaum talks about three major types of EDA tools: 1) ESL design tools, which help with defining requirements, modeling the system and exploring different design approaches; 2) IC front-end (FE) design tools, which include the RTL-level design capture, verification, simulation, timing, thermal, power, signal integrity and synthesis tools; 3) IC back-end (BE) design tools, which help with the physical design , such as floorplanning, placement, routing, extraction and rule checking tools (Birnbaum 2004). Wang et al. categorize the EDA algorithms, techniques and software under three broad categories, which include logic design automation, verification and testing, and physical design automation (Wang, Chang et al. 2009).

Given an electronic system modeled at the ESL, EDA automates the design and test processes of verifying the correctness of the ESL design against the specifications of the electronic system, taking the ESL design through various synthesis and verification steps, and finally testing the manufactured electronic system to ensure that it meets the specifications and quality requirements of the electronic system.

As can be seen in Figure 5: EDA Tool Use in IC development projects, each of the three major types of EDA tool consists of design tasks followed by verification and checking steps. If during verification and checking steps errors are found, the design is revised; this design-verify loop is repeated until the design

is error-free. The iteration may even go back to the front-end or system level design requirements if errors are found in the back-end design phase.

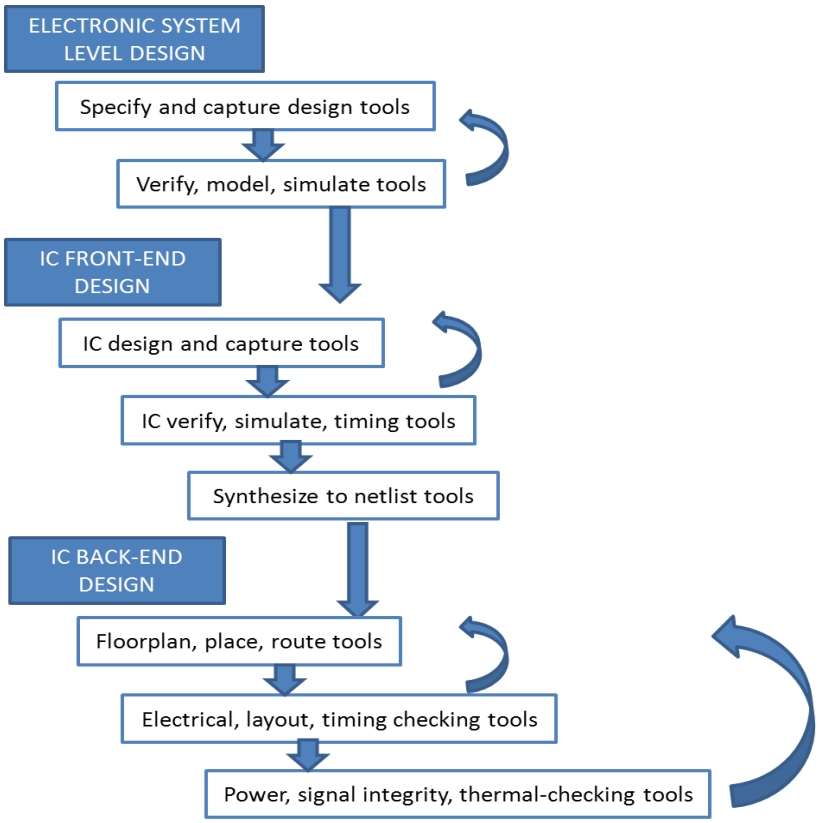


Figure 5: EDA Tool Use in IC development projects

Since the number of transistors per IC continues to double every two years or so and since there is a growing need for increased capacity, speed and capabilities, while the time-to-market is getting shorter and shorter, EDA tools will likely continue play a significant role in IC design process.

3.5 System Development Methodologies (SDM)

One size does not fit all! In other words, ”one system development methodology is not necessarily suitable for use by all projects or organizations” (CMS 2005). Each organization is unique, so one development method that works for one organization might not work for another. Moreover, “the same methodology is unlikely to work in the same organization on all projects”(Alexander 2018). “Different types of projects require different methods”(Griffiths 2018).

Software-development methods exist on a continuum from adaptive (Agile or value-driven types) to predictive (traditional or waterfall types)(Boehm and Turner 2004). “Predictive methods focus on analyzing and planning the future in detail and cater for known risks. They rely on effective early phase

analysis and if this goes very wrong, the project may have difficulty changing direction. Adaptive methods, on the other hand, focus on adapting quickly to changing realities"(EPMC).

Over the last 50 years a wide variety of system development methodologies have been created and used, each with its own recognized strengths and weaknesses.

Waterfall methodology was created by Royce in 1970 (Royce 1987). Since then, organizations have come to realize that while Waterfall software development methodology, which (an inheritance from the US Department of Defence) had clear benefits; it tended to be cumbersome, resource intensive and rather inflexible. Consequently, a flurry of alternatives, including Waterfall "light" was developed.

Agile was born in the 1990s and became over time the method of choice. It provided developers and clients with a sense of progression and allowed them to constantly re-evaluate the product. This flexibility, indeed, is Agile methods' trademark.

As with every continuum, there is a middle-of-the-road point, which combines the strengths of both waterfall and agile, referred to as hybrids – they attempt to extract the best qualities of both Agile and Waterfall while minimizing their drawbacks. After all, the key to success is to find the right balance between agility and discipline (Boehm and Turner 2004).

Many studies have been done to assess the success / failure rates of both Waterfall and Agile IT projects. The Ambysoft's Project Success Rates Survey (Ambysoft 2013) and the CHAOS report (Mersino 2018) from the Standish group had similar results.

	Success	Partial failure	Failure
Agile (Ambysoft)	**64**	**28**	**8**
Agile (Standish)	**42**	**50**	**8**
Waterfall (Ambysoft)	**49**	**33**	**18**
Waterfall (Standish)	**26**	**53**	**21**

With statistics like these, one can quickly conclude that Agile has a much higher probability of success and lower probability of failure than Waterfall. However, such a conclusion should be questioned because surveys cannot adequately factor elements such as:

- the comparability of projects in terms of nature, scope and depth;
- the capacity of the team involved on the project;
- the extent of the constituent base; and
- the choice of the appropriate development methodology.

The following section provides an overview on existing development methods: *adaptive* (*agile* or *value-driven*), *predictive (plan-driven)* and *incremental/hybrid.*

3.5.1 Adaptive (Agile or value-driven) Development

Traditionally, software was developed using waterfall-style, sequential software development methods, in which one phase is completed before proceeding to the next one. These linear methods have relatively long development cycles, they are documentation-driven and process heavy and they are not flexible to adapt to uncertainties and change (Highsmith 2004; Morien 2014). Agile methods have emerged as a new paradigm in IT development and project management as the waterfall approach often resulted in project failure because "every foundation assumption of the Waterfall Project Management Model is wrong and/or damaging to the prospect of a successful development activity" (Morien 2014). So-called *lightweight* agile software development methods evolved in the mid-1990s as a reaction against the *heavyweight* waterfall-oriented methods, which were characterized by their critics as being heavily regulated, regimented, micromanaged approaches to development.

Agile/iterative methods lie on the *adaptive* side of this continuum and they focus on rapid value delivery and responding to change (Fowler and Highsmith 2001). They are characterized by dynamic prioritization of requirements; effective and frequent interaction, communication, and reliance on tacit knowledge than documentation. They are flexible and focused on speed, performance, delivery and quality (Highsmith 2004; Paetsch, Eberlein et al. 2003). One primary goal of agile is to deliver a working product as soon as possible, which can then be rolled out to actual users and enhanced based on their feedback. With these methods, solutions evolve through collaboration between the clients and the developers during the entire development process aiming to embrace change and reduce the cost of change during the project's lifecycle (Mishra and Mishra 2011). This promotes adaptive planning, evolutionary development and delivery and makes it possible to adapt & adjust the software product almost daily (Highsmith 2004).

Agility provides ability to create and respond to change. Instead of trying to plan out a large part of the software process in great detail for a long a span of time, in agile methods, planning is done at several levels; detailed plans are produced only for short term as the plans tend to get obsolete very quickly in an ever changing environment. So frequent, iterative-cycle planning is required based on current conditions. The software is developed in short and iterative development cycles and it gets integrated and tested on a continuous basis (Highsmith 2002). At the end of each iteration, customers receive a working code and

they can provide their feedback to help incrementally build the system they desired (Fowler and Highsmith 2001). Testing is performed at every step of the development which means that defaults are immediately identified and fixed.

The *Agile Manifesto* introduced the term "agile" in 2001. The manifesto highlighted four values: 1) *individuals and interactions over processes and tools*, 2) *working software over comprehensive documentation*, 3) *customer collaboration over contract negotiation*, and 4) *responding to change over following a plan*.

- Individuals and interactions – In agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
- Working software – Working software will be more useful and welcome than just presenting documents to clients in meetings.
- Customer collaboration – Requirements cannot be fully collected at the beginning of the software development cycle, therefore continuous customer and/or stakeholder involvement is very important.
- Responding to change – Agile development is focused on quick responses to change and continuous development.

As stated in the Agile Manifesto, delivering a working code is more important than developing a comprehensive documentation. In a letter to IEEE *Computer*, Steven Rakitin (Rakitin 2001) expressed cynicism about agile development, calling an article supporting agile software development "yet another attempt to undermine the discipline of software engineering" and translating "Working software over comprehensive documentation" as "We want to spend all our time coding. Remember, real programmers don't write documentation". This approach is disputed by proponents of agile software development, who state that developers should write documentation if that's the best way to achieve the relevant goals, but that there are often better ways to achieve those goals than writing static documentation. Scott Ambler (Ambler 2014) states that documentation should be "Just Barely Good Enough" (JBGE), that too much or comprehensive documentation would usually cause waste, and developers rarely trust detailed documentation because it's usually out of sync with code, while too little documentation may also cause problems for maintenance, communication, learning and knowledge sharing.

The following figure (Figure 6: Agile development) captures the conceptual framework that promotes foreseen tight iterations throughout the development cycle.

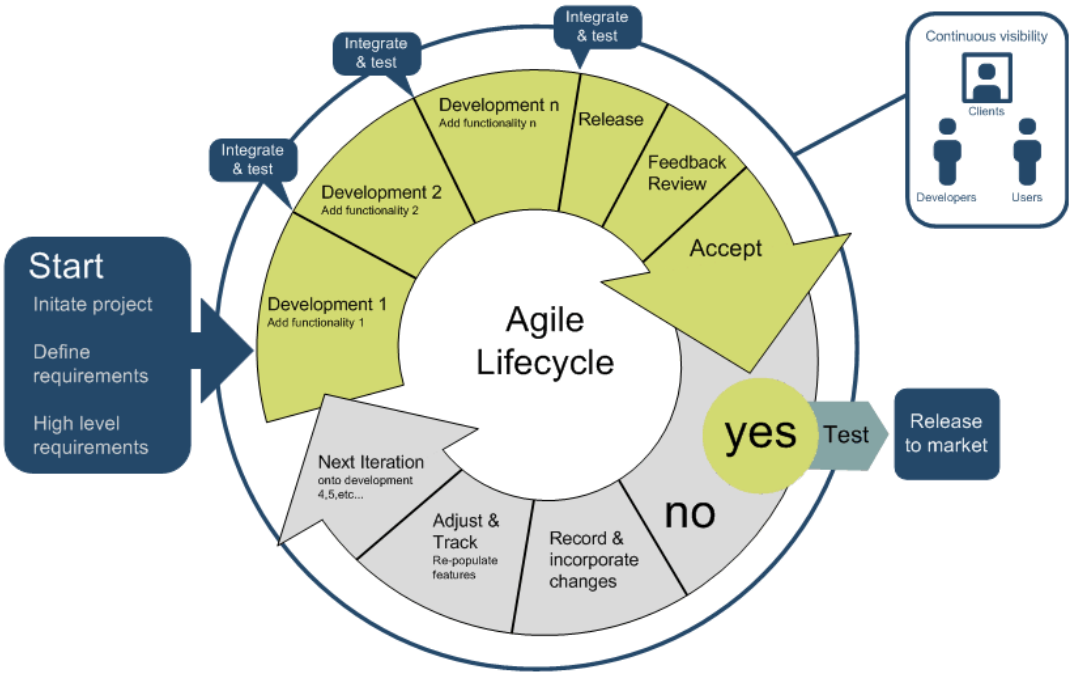


Figure 6: Agile development

One key of adaptive development methods is a "Rolling Wave" approach to schedule planning, which identifies milestones but leaves flexibility in the path to reach them, and also allows for the milestones themselves to change. Adaptive methods focus on adapting quickly to changing realities. When the needs of a project change, an adaptive team changes as well. An adaptive team will have difficulty describing exactly what will happen in the future. The further away a date is, the vaguer an adaptive method will be about what will happen on that date. An adaptive team cannot report exactly what tasks they will do next week, but only which features they plan for next month. When asked about a release six months from now, an adaptive team might be able to report only the mission statement for the release, or a statement of expected value vs. cost.

Since the agile manifesto (Fowler and Highsmith 2001), a lot of research has been developed on the agile software development ((Boehm and Turner 2004; Dingsøyr, Nerur et al. 2012; Dybå and Dingsøyr 2008). A systematic search of relevant literature shows that agile methods prove useful when:

- the project outcomes are of lower criticality;
- culture demands a capacity to respond to change;

- requirements and specifications are not fully defined or documented up front either because the user community has not fully ironed them out or because external forces are mandating requirements changes;
- client and users can "truly" be involved with the development team;
- the project team is small; and,
- time and cost are important outcomes.

On the other hand:

- particularly at the beginning of the project, due to the uncertainty of all the requirements, it may be difficult to assess the required effort and estimate how much work is required ;
- the communication is crucial throughout the project; if communication breaks down at any point or feedback from clients is unclear, teams may concentrate on the wrong development areas leading scope creep, the project going over budget and final deliverable deviating from what was originally planned (Rivera 2020);
- all project team key members must be "senior and experienced" in project development and must understand and accept the Agile methodology - failing to do so results in increased time and cost and ultimately could stall or kill the project;
- the lack of "more complete" documentation makes the transfer from the development teams to the operations and maintenance teams more difficult;
- the involvement of users at every stage can lead to scope creep which, if left un-managed, it can result in increase cost and timeline or worst.
- Agile being about the adaptability to change and adapt requires fast decision cycles by the product owner. If the product owner does not understand or care about the method, then the project is at risk.
- Agile methods are typically used in small and medium-sized projects; their use in large-scale projects is considered challenging because when complexity of the software; the size of the development team and the number of stakeholders increase, the need for more systematic approach and effective coordination will grow (Boehm and Turner 2004; Mishra and Mishra 2011).

3.5.2 Predictive (plan-driven) Development

Predictive methods, in contrast, focus on analysing and planning the future in detail and cater for known risks. In the extremes, a predictive team can report exactly what features and tasks are planned for the

entire length of the development process. Predictive methods rely on effective early phase analysis and if this goes very wrong, the project may have difficulty changing direction. Predictive teams will often institute a Change Control Board to ensure that only the most valuable changes are considered.

Linear/Waterfall methods lie on the predictive side of the development continuum. They are the traditional approach, based on a series of sequential phases (requirements analysis, design, testing, implementation, and maintenance) – a phase must be completed before moving to another phase. Once the development is complete a Quality Assurance (QA) team will test the product for conformance to the user requirements and specifications. Failing QA meant going back to the development phase to correct defaults.

One of the differences between agile and waterfall is that testing of the software is conducted at different stages during the software development lifecycle. In the Waterfall model, there is always a separate *testing phase* near the completion of an *implementation phase*. However, in Agile and especially XP, testing is usually done concurrently with coding, or at least, testing jobs start in early iterations.

In essence, if the planning and design phases were done correctly, one can predict the results.

The pure Waterfall method is considered plan-driven, process-heavy, document-centric and too formal (Boehm and Turner 2004; Highsmith 2004). With relatively long development cycles, they are more suitable to predictable, stable environments (Highsmith 2002). As a result, many variations were developed and used. Some of these variations include the Rapid Development models such as the "modified waterfalls", "sashimi model (waterfall with overlapping phases)", waterfall with sub projects, and waterfall with risk reduction (McConnell 1996). Other software model combinations such as "incremental waterfall model", "spiral model" and "V-Model", which may be considered and extension of the waterfall model also exist (Kostigoff 2003).

According to Eriksson (Eriksson 2016) and Schwaber (Schwaber K. 2017), waterfall methods proved useful when:

- project outcome's criticality is high or extreme;
- culture demands extreme quality;
- requirements are well defined - simple or complicated processes do not matter if they can be defined, modeled and remain stable;
- complete documentation is needed including architecture, functional and technical specifications to facilitate integration into production and transfer to maintenance; and,

- time is not the essence, but results are – plenty of time for testing.

On the other hand:

- requirements are not always initially available, which results in trials and errors (time and costs increases);
- adequate time is not always available and often resulted in poor documentation, incomplete software and poorly tested products; and,
- Fixing defects prove to be an expensive and time-consuming proposition.

According to Micic, waterfall methods are recommended for "teams with less experience as well as for organizations in which management has no experience of running projects". However, due to their rigid, process-heavy, control oriented nature, these methods are not recommended for projects that require quick response or when requirements are not well understood/defined or are likely to change in the course of the project (Micic 2017).

Boehm & Turner states "predictability, repeatability, and optimization" as the main objectives of the more traditional waterfall methods. (Boehm and Turner 2004).

3.5.3 Incremental / Hybrid Development

Incremental/Hybrid methods lie in the middle of the adaptive and predictive continuum. By combining the elements from the both ends of the spectrum, they attempt to mitigate the drawbacks of extreme iterative or extreme plan-driven methods. Waterfall could benefit, in many projects, from the adaptability of Agile method and Agile methods could benefit from the more structured and systemic approach of Waterfall (Eriksson 2016).

"Water-scrum-fall' and "Agifall" are amongst the two popular hybrid methods that are used in the industry (Eriksson 2016). According to Eriksson, "Water-scrum-fall" method suggests using the traditional waterfall approach for the pre-development activities such as planning, requirements gathering, budgeting and documenting the projects progress and advocates the use of agile approach (time-boxed, iterative process) during the software design and development. "Agifall" method is another hybrid method which aims to combine the best of waterfall and agile development methods. In this approach, agile approach is used in the planning and requirements activities of the project and loose waterfall process is used during the development, where one phase can begin before the previous phase is completed but agile principles are used during the development phase.

Erickson(Eriksson 2016) states that the Hybrid methods are not perfect either, but we believe that these somewhat neglected development methods would gain to be better known since, if adopted, they could improve the overall project failure rate.

3.6 SDM Evaluation & Selection Frameworks

Project managers need to select the appropriate SDM for their projects. According to Micic (Micic 2017), the process of selecting a methodology is more subjective and less precise than technical and according to Jones "selecting a software development methodology has more in common with joining a cult than it does with making a decision" and "many companies do not even attempt to evaluate methods, but merely adopt the most popular, which today constitute the many faces of agile" (Jones 2013).

How do decision makers recognize and measure the concept of "fit" between the chosen SDM and the project? According to Mullaly & Thomas,"attaining fit suggests that there is an alignment between what is being implemented and the environment and situation of an organization"(Mullaly and Thomas 2009).

Alqudah & Razali performed a systematic literature review of 53 articles that were published between 2001 and 2015 to identify the key factors to be considered when selecting an appropriate agile method and they identified "the nature of the project, development team skills, project constraints, organizational culture and customer involvement" as the crucial factors that can be used in selecting Agile methods (Alqudah and Razali 2017).

Griffiths introduced various agile suitability filters, such as the Gartner bi-modal IT, Alistair Cockburn's Crystal family of methods, DSDM Suitability filter, Boehm and Turner's radar chart and the organizational suitability filter, to help assess if an agile approach is suitable to an organization and project (Griffiths 2013). Gartner bi-modal IT uses three attributes (the perceived degree of governance, likelihood of change and the type of solution) to suggest if a plan-driven or an agile would be the right approach. Alistair Cockburn's work argues that the team size and system criticality are two factors that should be considered for assessing the project's agile suitability. DSDM suitability filter uses a list of Yes/No questions to determine if project would benefit from an agile development approach. Boehm and Turner define five crucial attributes (Personnel, project criticality, dynamism, project size and culture) to assess a project for either an agile approach or more of a traditional/plan-driven approach (Boehm and Turner 2004). "This approach is an exceptional contribution to the notion of tailoring the software process to match the project context" (Geras, Smith et al. 2006).

Datta (Datta 2006) introduced an "Agility Measurement Index" as an indicator for determining whether a Waterfall, Unified Software Development Process (UP), or eXtreme Programming (XP) methodology should be used in a project.

Abrahamsson et al. (Abrahamsson, Warsta et al. 2003) used a comparative framework to compare ten agile methods based on six analytical criteria: project management support, s/w development lifecycle coverage, availability of concrete guidance for application, adaptability in actual use, research objective and empirical evidence.

Qumer et al. (Qumer and Henderson-Sellers 2006) introduced the 4 Dimensional Analytical Tool called 4-DAT which uses four attributes (method scope, agility, agile values, software process) for the analysis and comparison of XP and SCRUM agile methods.

Taromirad & Ramsin (Taromirad and Ramsin 2008) researched and assessed the evaluation frameworks that were introduced to facilitate the selection of an appropriate agile development methodology and concluded that "although several evaluation frameworks or methods have been introduced for comparing, analyzing or evaluating agile methodologies, they lack in addressing method engineering and project management requirements". In another study, Taromirad & Ramsin introduced a new evaluation framework called CEFAM, which uses five attributes (the nature of the project, development team skills, project constraints, customer involvement and organizational culture) to guide decision makers in the selection of an appropriate agile method (Taromirad and Ramsin 2008).

There are several studies in the literature that compares traditional/waterfall methodologies with agile and discuss project characteristics that will make a difference when it comes to choosing the right development methodology (van Casteren 2017). Jones introduced several standard metrics, such as "function points, defect removal efficiency (DRE), Cost of Quality (COQ), and Total Cost of Ownership (TCO) to compare a sample of contemporary software development methods" (Jones 2013).

However, even though several studies have investigated the selection of agile methods, according to our knowledge, there is yet no comprehensive framework that uses the crucial factors to determine whether a traditional/plan-driven/waterfall, agile or hybrid type of SDM would be most suited for a project. This research aims to address this need by introducing a new comparative framework called MAF that provides a full coverage of seven factors that are regarded as important when it comes to making such a decision.

After applying this new framework to the MPIC project, the study will demonstrate why the agile methodology should be used.

The Agile movement is not anti-methodology; in fact, many of the founding members want to restore credibility to the word methodology. They embrace modeling, but not in order to file some diagram in a corporate repository. They embrace documentation, but not hundreds of pages never-maintained and rarely-used. They plan, but recognize the limits of planning in a turbulent environment.

The use of agile software development methods has increased in recent years because agile methods provide organizations with the ability to create, innovate and respond to change; deliver higher quality software in a shorter time; increase customer satisfaction by responding to their emerging requirements through the lifecycle of the project (Mishra and Mishra 2011).

Agile methods are adaptive and flexible because they are built around the assumption that requirements are likely to change as the project progresses. Instead of trying to gather all the requirements upfront, they break up the requirements into iterations. The software is developed in short and iterative development cycles and it gets integrated and tested on a continuous basis (Highsmith 2002). At the end of each iteration, clients receive a working code and they can provide their feedback to help incrementally build the system they desired (Fowler and Highsmith 2001). Agile methods are people-oriented; they are based on having a partnership with stakeholders in order to discover and deliver the stakeholder requirements just in time; they rely on the talent, knowledge and strengths of the development team; they are based on constant interaction and collaboration between developers and customers (Highsmith 2004).

Compared to traditional software engineering, agile development is mainly targeted at complex systems and projects with dynamic, nondeterministic and non-linear characteristics, where accurate estimates, stable plans and predictions are often hard to get in early stages, and big up-front designs and arrangements will probably cause a lot of waste, i.e. not economically sound. These basic arguments and precious industry experiences learned from years of successes and failures have helped shape agile's favor of adaptive, iterative and evolutionary development.

There are several agile methods discussed in the literature, such as eXtreme Programming (XP), Scrum, Kanban, Dynamic Systems Development Method (DSDM), Feature Driven Development (FDD), Crystal, Unified Process (UP), Agile Unified Process (AUP) and Adaptive Software Development (ASD); but XP and Scrum are the most common described agile methods in literature (Boehm and Turner 2004; Dingsøyr,

Nerur et al. 2012; Schwaber K. 2017). Boehm and Turner (Boehm and Turner 2004) discuss and compare these various methods. Refer to Appendix B: Overview of Several Agile Methods that are commonly used for an overview of several agile methods that are commonly used.

The most widespread project management approach used in complex software projects is mainly based on a hybrid usage of Scrum and XP. This is probably because of the fact that Scrum and XP provide complementary practices and rules. Scrum is used to manage the steps taken to develop software in conjunction with the use of XP to ensure the quality of the software. The framework of Scrum activities and XP's feedback and communication are the concepts that are used for the management processes. Both XP and Scrum emphasises iterative and incremental development. However, in contrast to Scrum, XP provides explicit and hands-on methods for developers. Refer to Appendix C: Comparison of commonly cited agile methods in the literature for a comparison of some of the characteristics of some agile methods.

Each organization is unique, so one agile method that works for one organization might not work for another. There is not one single, perfect agile method that suits all organizations; so combining more than one method together; especially XP (for development practices) with Scrum (for project management) appears to be more effective (Parsons, Ryu et al. 2007).

3.7 AGILE Project Management

Agile is not only a development methodology but it is also a project management framework. All Agile methods advocate the core agile characteristics such as direct client engagement, frequent incremental releases, adaptability to change, self-organization/empowerment, emphasis on simplicity and continuous improvement.

The traditional project management is usually manifested as "command and control" style of management (Morien 2014). "Plan the work and work the plan" type of traditional phased project management approach seems to end often in chaos, or "the delivery of a system that is less than useful, and has low business value" (Morien 2014).

Agile project management is a new approach to planning and managing software projects, which is described with "highly iterative, fast feedback cycles, total transparency of progress and outcomes, self-managed, validation and verification frequently and systematically" (Morien 2014). Literature from practitioners and scholars suggests that agile project management is about adapting quickly to changing

requirements, fostering innovative solutions (products or services), leading people into action through self-organized teams, and delivering a result that is acceptable to the customer on an iterative basis.

Scrum, which appears as the most popular method in literature, is an agile project management method based on flexibility, adaptability and productivity. Scrum distributes the project manager roles and responsibilities across the Product owner, Scrum Master and the Scrum team. Fry and Greene recommend using Scrum since it provides all essentials and has a very low learning curve (Fry and Greene 2007).

The Product owner is responsible for ensuring that the right product is built, and in the right order. He/she has the ultimate decision making power about the product and the business aspects of the project. The Scrum Master acts as the team's coach, helps team members work together in the most effective manner possible, ensures that Scrum processes are followed, removes obstacles to progress, facilitates daily stand-up meetings and discussions, and tracks the progress and issues. The Scrum team is self-organized and has the authority to decide on the necessary actions to complete the sprint goal. Team members collaboratively decide which person should work on which tasks, to achieve stated sprint goals. At the beginning of each sprint, sprint planning meetings take place, to divide the tasks across the team members. Daily scrum meetings, which usually take about 10-15 minutes, are held to track the progress and obstacles. Each member answers three questions, which are: “What did you do yesterday?”, “What are you doing today?” and “Do you have any blocking issue that prevents you from achieving your goals?”.

In order to successfully adopt agile methodologies in IC development projects, organizations need to understand that this will be a learning process and it will take a number of iterations before the improvement is seen. Through continuous learning, through experimenting and collaboration, the organization will attain a deeper understanding of agile development and they will move towards a higher level of maturity in their agile adoption process. At the heart of all this process lies the people and communication; therefore organizations also need to be careful about personnel management at the project team level; and they should view people and communication as the critical factors for building great technology.

4. THESIS OBJECTIVES

4.1 Research Questions

This research aims to develop a framework based on theory and practice, the main research question to be answered is:

- **Research Question 1:** *How we can improve design productivity and success in Integrated Circuit projects hence would facilitate more efficient, faster semiconductor development and time-to-market.*

Based on the answer of the research question 1, the research will also work on the following research questions:

- **Research Question 2:** *Can we successfully adopt lightweight, rapid, flexible and adaptive development methods and frameworks like Agile in IC development projects, particularly in the MPIC project?*
- **Research Question 3:** *What elements contribute to the choice of suitable SDM for a given project and how should these be included in a framework to be used to identify an SDM that would be more suitable for the MPIC project?*
- **Research Question 4:** *Can we make use of design tools such as Electronic Design Automation(EDA) tools to improve the productivity and success in IC projects?*

The research questions listed above will shape and guide the qualitative research study.

4.2 Conceptual Framework

This research proposes a causal model that incorporates and integrates two factors that affect success in IC development projects. These two factors are: 1) IC Development & Project Management Methodology and 2) EDA Tools.

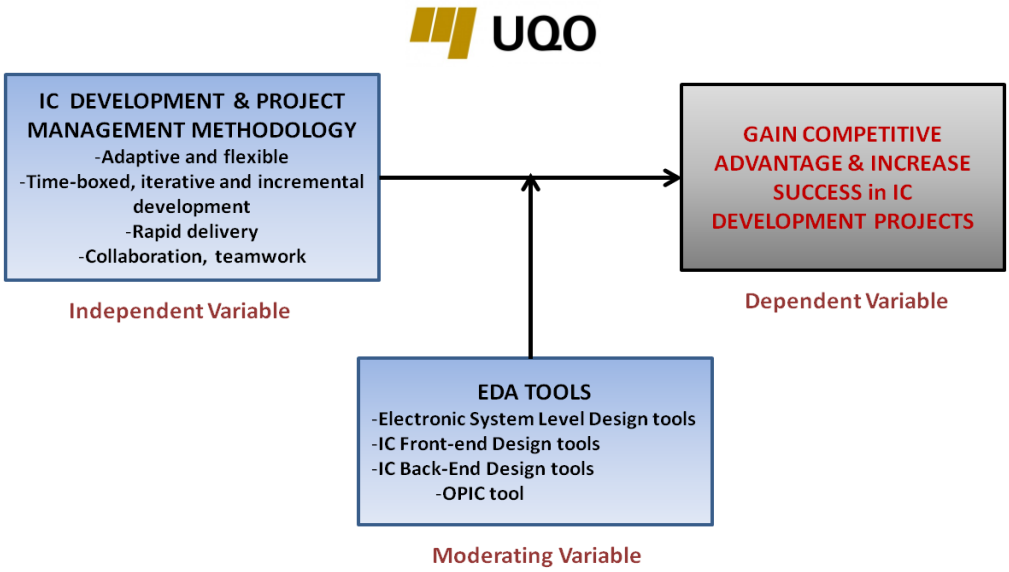


Figure 7: The research model that incorporates and integrates 2 factors

As can be seen in the figure above (Figure 7: The research model that incorporates and integrates 2 factors), *IC Development & Project Management Methodology* has been defined as an independent variable while *EDA Tools* is defined as a moderating variable, as it has appeared to have a moderating effect on the relationship between the independent variable (cause) and the dependent (effect) variable. The dependent variable of this research study, in other words, the variable that is believed to be impacted by the changes in the independent variables, is the *success in IC development projects*.

Dependent Variable (Consequence)	Success in IC Development Projects
Independent variable	IC Development & Project Management Methodology
Moderating Variable	EDA Tools

Table 1: Research Variables

As identified and listed in Table 1: Research Variables, one independent variable and one moderating variable will be used in this research study. The research model used in this study suggests that the "Tools" factor produces a moderating effect on the relationship between dependent and independent variables as the use of proper tools can positively influence the strength of the relationship between dependent and independent variables.

Based on the model shown in Figure 7: The research model that incorporates and integrates 2 factors, the following theoretical propositions have been put forward. These propositions will be addressed and tested by this research study.

P1: There is a significant positive relationship between the use of Agile Development and Project management methodology in the IC Development projects and the project success.

P2: The use of proper EDA tools in IC Development projects has a moderating effect on the relationship between and project success.

4.3 Novelty and Originality

Through a review of the literature, this research aims to explore the use of agile development and project management methodologies in the IC Development projects, specifically for the EDA tools development.

The objectives of this research can be defined as follows:

1) *To create a new EDA tool that will allow replication of an existing layout (reference design) in different technology nodes by automatically porting analog and mixed signal circuits*

   Semiconductor market faces huge challenges in terms of porting complex cases in variety of latest technologies (28nm, 22nm, 14nm, 10nm and 7nm, etc.). To perform the porting tasks between different technology processes, major IP players rely on heavy usage of human labour (usually outsourced to countries to lower the cost), which has a higher risk of introducing errors. Because commercial tools that are currently available in the market do not include features like automatic porting between different technology nodes and foundries for analog and mixed-signal circuits. Moreover, they do not re-use the existing layout by making changes on it; instead they regenerate the layout from scratch. This method is risky and could impact the quality of the IP resulting from the loss of work that was already done in previous technology. Therefore, the new layout needs to get retested rigorously for reliability, which impacts the IP delivery timelines.

   Since available commercial tools do not include features like automatic porting between technology nodes and foundries, traditional manual porting methods are used. Given the size and complexity of each migration, traditional manual porting methods are too time-consuming and costly. A tool that would automatically migrate any Intellectual Property (IP) design and perform porting flow with minimum human intervention from one foundry specific technology to another would be very valuable to the industry. By using this kind of automated tool, semiconductor companies can increase the efficiency of their IC design process by reducing the porting time and the risk of errors, as well as the costs. This type of technology will aptly fit in an the industry that is fast moving and highly competitive with increased levels of integration, frequent upgrades, and technology shrinks" (Macintosh 2014).

   MPIC project aims to introduce a new EDA tool to automate porting of analog and mixed signal circuits (including standard cell library migration) from one technology node to another. This tool will

automatically perform porting tasks with minimum human intervention and will replicate an existing layout (reference design) in different technology nodes and foundries by preserving the key characteristics of an existing layout including matching of critical components and their relative placement. The tool is targeted for use in the hard IP market.

This new EDA tool aims to provide a fast and robust analysis of hard IP designs attached to a technological process for manufacturing integrated circuits through different modules throughout the layout, floorplan, component placement plan, routing between blocks, and finally the preservation of matching critical components between reference and target designs. It allows for an analysis of different modules characterizing their performance (speed, gain, consumption, etc.) to generate its own mask pattern, including LVS and DRC conforming to reference circuit.

2) *To create a new IC design model for IP porting from one technology to another by carrying out schematics and layout porting steps in parallel*

Without design technology it would be impossible to implement, verify and test the complex IC systems (Bryant, Cheng et al. 2001). Tools and methodologies need to co-evolve in step with process technology characteristics and design challenges. To increase the design productivity, this research proposes a new IC design model as explained in 2) Proposed IC Design model . Please refer to **Erreur ! Source du renvoi introuvable.** for more information.

3) *To introduce a new methodology assessment framework called MAF, in order to identify an SDM that would be more suitable for a given project, including the MPIC project.*

To address challenges introduced by the increasing levels of design complexity of the microelectronic systems, not only new tools, but also new methodologies are required. This research introduces a new framework called MAF which helps decision makers assess a given project against the seven factors and determine the type of SDM that would be best suited. Next, by applying this new framework into the MPIC project, it identifies the agile/iterative methodology as the most suited SDM for the project.

4) *To adopt agile development and project methodology in the MPIC, an EDA tool development project*

This research while providing insights into the EDA development process, aims to demonstrate if and how agile methodologies can be applied to IC Development projects, specifically to the EDA tools development.

Because by adopting agile practices, such as the XP and/or Scrum agile methodologies to MPIC project, I believe we can deliver higher quality software in a shorter time. By using an iterative and incremental development approach, we can quickly respond to emerging requirements throughout the lifecycle of the project. This project will illustrate how a hybrid of XP and Scrum agile methodologies can also be used to foster continual process of evaluation, planning, setting requirements, analysis, design and deployment of code in the MPIC project.

4.4 Importance of the Research

Agile methods and practices have been developed mainly for the software development projects. Literature review has also revealed that agile methods have been mainly applied in software development projects. There seems to be a lack of research on the adoption of agile in IC development projects. The focus of this research has been on understanding if iterative, incremental development concepts of agile methods can be used in the IC development projects. Please see Appendix F: Papers Published for a proposed conceptual framework for agile hardware/software co-design that can be applied in the IC development projects.

In order to enhance our understanding and to show how the workflow for the IC development projects can be managed more effectively, this research will focus on developing a new framework to help project managers select the appropriate SDM for their project. The research will also explore the adaptation of agile methodologies into IC development projects by following a case study approach on an EDA tool development project called MPIC. By applying agile methodologies successfully into the MPIC project, I believe that the project team can be made more effective and productive. Splitting the IC development into sub-tasks, developing in sprints, incorporating continuous integration and testing into the development process should allow the development team to address issues while they are fresh. Through constant team communication and feedback received over the Scrum meetings, sprint reviews and retrospectives, the product and the processes can be continuously improved and working functionality can be demonstrated more quickly.

4.5 The New MAF Conceptual Framework

Making an informed System Development Methodology (SDM) decision must be the first step of every system project. Micic (Micic 2017) argues that the process of selecting a methodology is more subjective and less precise than technical and he states “the choice of methodology, among other things, greatly depends on the size of the organization, the type of technology used, the style of management, and the

structure of employees, locations, companies, clients/users and a number of other factors that need to be taken into account" (Micic 2017). According to Jones "selecting a software development methodology has more in common with joining a cult than it does with making a decision" and "many companies do not even attempt to evaluate methods, but merely adopt the most popular, which today constitute the many faces of agile" (Jones 2013).

How do decision makers recognize and measure the concept of "fit" between the chosen SDM and the project? According to Mullaly & Thomas, "attaining fit suggests that there is an alignment between what is being implemented and the environment and situation of an organization"(Mullaly and Thomas 2009).

"A framework is a well-used method for clarifying the properties of, and comparing methodologies" (Strode 2006). Our literature review has proven that there are several frameworks that have been used to define, compare and evaluate SDMs (Alqudah and Razali 2017; Griffiths 2013; Micic 2017; Taromirad and Ramsin 2008). However, there is yet no comprehensive framework that uses the crucial factors to determine whether the traditional/plan-driven/waterfall, agile or hybrid type of system development methodology would be most suited for a project.

This research proposes a causal model that incorporates these seven factors and introduces a new framework called "Methodology Assessment Framework (MAF)". This framework is a tool that factors in these seven factors to come up with an overall assessment that will help determine the system methodology approach that is best suited for the project.

Next, by applying the MAF conceptual framework into the MPIC project, this research aims to determine whether traditional/plan-driven/waterfall, agile/iterative or hybrid type of system development methodology would be most suited for the project.

Through a review of the literature, this study found out that existing evaluation frameworks lack several aspects. Most of them have not considered evaluating agility. The nature and the complexity of the project as well as the extent and enrollment of the project's constituent base have been neglected or partially addressed. To address the need of a comprehensive evaluation framework, this research has identified seven factors that are critical to providing the project's executive sponsor and governance body with a dashboard view over the project landscape. They are as follows:

1. Outcomes being addressed by the project **(OUTCOMES)**
2. Scope /features of the project **(SCOPE)**

3. Nature and complexity of the project - CYNEFIN framework **(CYNEFIN)**
4. Extent and enrollment of the project's constituent base **(CONSTITUENTS)**
5. Applicability of Agile principles to the project **(AGILE PRINCIPLES)**
6. Team expertise and experience in system development methodologies **(TEAM)**
7. Maturity of the organizations involved on the project **(ORGANIZATION)**

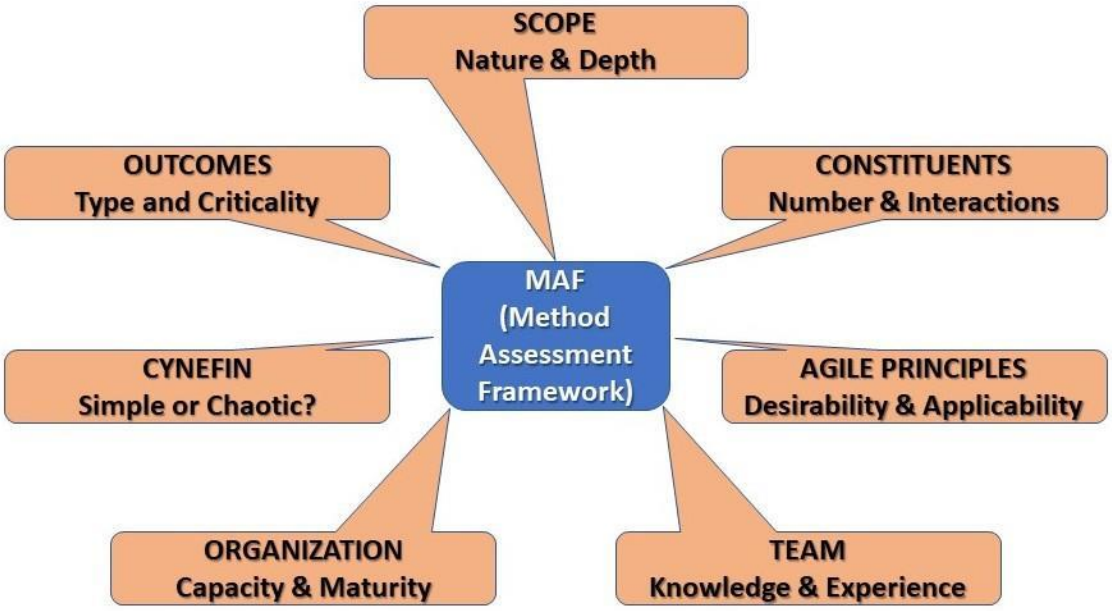


Figure 8: Conceptual methodology selection framework based on MAF

The following table presents, for these seven factors, the postulate for the selection of the appropriate system development methodology and the approach to assessing and rating each factor. Each factor and their assessment metrics will be discussed in more detail later in the document.

FACTORS	OUR UNDERSTANDING	RATINGS
OUTCOMES	The number and criticality of outcomes have bearing on the development method selected. Projects with many important, substantial or critical outcomes will generally be better served by a formal method like Waterfall as the end product, and the results are defined in advance with these methods. Please see Outcomes – how critical? for more information.	A low rating (0: Somewhat Important – 1: Important) will tend to indicate that Agile would work fine. A high rating (3: Critical - 4: Absolutely critical), on the other hand, will tend to indicate that a formal methodology like Waterfall should be used. A middle-of-the-road rating would likely lead to a method that includes elements of both Agile and Waterfall.
SCOPE	Change is a reality, so we should prefer adaptability where possible. The "larger" a project is (with lots of	A low rating (0: Very small – 1:Small) will tend to indicate that Agile would work fine as agile welcomes changes.

	features), the more likely the Executive Owner will want as clear a picture of the end product and results as possible as he/she will have to sell, support, convince a large number of people (constituents). Please see Scope of the project for more information.	A high rating (3: Large – 4: Very Large), on the other hand, will tend to indicate that a formal methodology like Waterfall should be used as traditional methods work better when scope is known in advance or when changes are limited. A middle-of-the-road rating would likely lead to a method that includes elements of both Agile and Waterfall.
CYNEFIN	The nature of the project (simple, complicated, complex, chaotic) has a bearing upon the method to be used. Agile will be most useful in context where there is a lot of uncertainty, complexity, disorder and even chaos as it is based on quick products / component iterations. At the other end of the spectrum, Waterfall will work best when the project is simple with the requirements, users, processes are known. Please see CYNEFIN of project for more information.	Simple and Complicated projects (ratings of 0 and 1) are best suited for Waterfall methods. When a project is characterized as Complex or Chaotic (rating of 2 or 3) an iterative rapid application development method like Agile will have the highest probability of delivering, as a minimum, some results / product. Waterfall projects in such an environment will unlikely ever be completed! With regards to Complex project (rating of 2) it is also likely that a mix of Iterative and formal method will work best as these types of projects usually involved many sub or associated projects governed and delivered outside of the core project team.
CONSTITUENTS	People make systems work or not. As Saunders stated, the human element is of paramount importance in the successful control and implementation of a project" and most projects fail because of human error (Saunders 1992). The best system will be a failure if stakeholders, partners and service providers do not or cannot support it. Clearly identifying and managing all constituents at each phase of a project will make or break a system project. Please see Constituents of the project for more information.	Stated in a simple way, projects with few constituents (e.g.: executive sponsor, project director and a small programmer team) and only a few basic roles (e.g.: Inform, Consult or Educate) (rating of 0 or 1) will work best with Agile type methods, providing that the team is experienced. At the other end, projects with a very large number of constituents and an extended number of roles (e.g. Develop, Engage, Involve or Approve) (ratings of 3 or 4) will require more planning and a capacity to present what the result will be and, consequently, would require a more formal approach like Waterfall. As for other factors, a middle-of-the-road result could work best with a hybrid

		approach (a bit of this and a bit of that).
PRINCIPLES	Time has proven that the twelve Agile principles are as important as the method itself. In many cases, some or most of the principles are impractical, undesirable or irrelevant. I suggest that if there is little adherence to the Agile principles, there is a strong probability that the project would be better served by a formal method like Waterfall. Please see Applicability of Agile Principles for more information.	A low rating (0 : 20 % support of agile traits – 1: 40% support of agile traits) will tend to indicate that Agile would not work fine. A high rating (3: 80% support of agile traits – 4: 100% support of agile traits), on the other hand, will tend to indicate that an Agile method is likely appropriate. A middle-of-the-road rating (2: 60% support of agile traits) road rating would likely lead to a method that includes elements of both Agile and Waterfall could work best.
TEAM	As stated above, people make systems work or not. Having a team business manager and IM/IT staff that are experienced and knowledgeable in IM/IT and business aspects is the key to project's success. It is suggested that Agile team members should be more "experienced" than those of traditional/waterfall methods. Formal methods are fully documented and supported by software engineering practices and can, therefore, be applied by less-experienced people. They can "follow the recipe" much more so than in the case of an Agile method where the team if given an "open book" and must rely on its own capacities. Please see Team expertise for more information.	When the team self assessment values are 2: Proficient, 3: Advanced, or 4: Expert, the team is clearly strong enough to deal with Agile method. When the assessment is 0: Limited or 1: Beginner, formal methods might work better… if they can get some external help regarding applying and managing the formal method.
ORGANIZATION	Lots have been said and written about organizational capacity and maturity. Developing systems in organization that operate in an ad-hoc inconsistent manner in all or most IM/IT or business fields will have difficulty successfully delivering system projects. We believe that when such is the case (as we did for team) a formal method will work better as, by itself, it provides a structure…… follow the recipe, and you will get a product! At the other end, highly mature and capable	When the organizational capacity and maturity values are 2: Established, 3: Predictable, or 4:Optimizing, the organization is likely strong enough to deal with Agile method. When the assessment is 0: Performed or 1: Managed, formal methods might work better as the success is, to a large degree, dependent on following a detailed methodology.

	organisations can handle any type of method, especially Agile as they can easily provide Agile teams with the experience or resources they may not have. Please see Organization's maturity for more information.	

The following radar chart can be used to assess where your projects currently are with respect to the 7 key axes that were discussed in the table above. If all of your ratings are near the center, you are in agile methodology territory. If they are at the periphery, you will best succeed with a plan-driven approach. If you are mostly in one or the other, you need to treat exceptions as sources of risk and devise risk management approaches to address them.

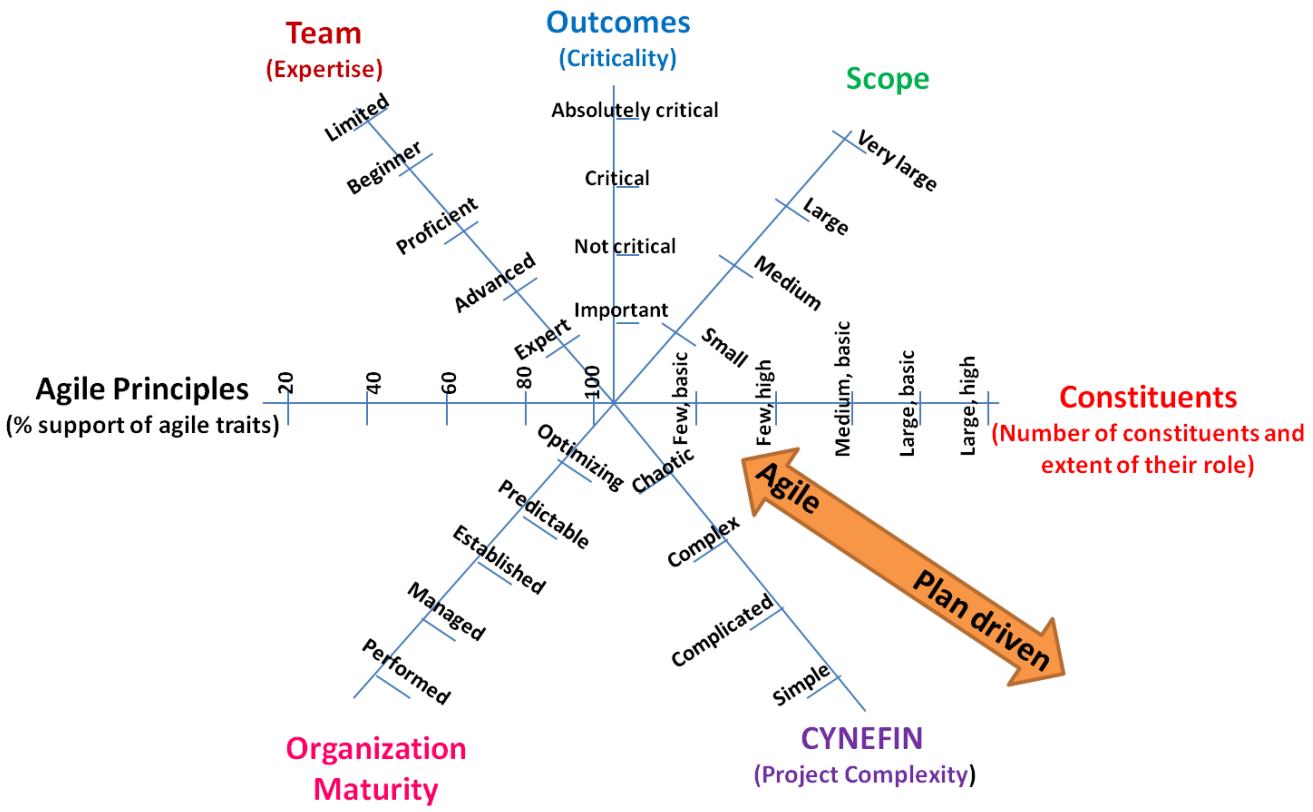


Figure 9: 7 key MAF dimensions affecting SDM selection

Since no project is the same, patterns will differ. In Proposed System Development Methodology (SDM), a practical application of this framework is presented to demonstrate how it can be used to evaluate a project against these seven dimensions to determine the best suited SDM.

4.6 Case Study: MPIC Project

The following sections provide an overview of the MPIC project and explore how agile development and project management methodologies can be used in this EDA software development project.

MPIC project aims to introduce an automated porting tool to increase the effectiveness and efficiencies of the design conversion process by reducing the human resources required and by shortening the porting time while reducing the failures during conversion. As a part of this project, a new agile IC design model will also be introduced which proposes to carry out the schematics and layout design phases in parallel. Furthermore, the new framework called MAF will be used to assess the MPIC project against the seven factors and will propose an SDM that will be best suited for the MPIC project.

4.6.1 Proposed Layout Porting Tool

"Hard IP represents a custom design that has already been implemented in a physical design, verified and delivered in a GDSII representation" (Zelnik 2002). Consequently, reuse of hard IP means migration of a given GDSII representation to a new process technology. However, manipulating the data in GDSII form presents a significant challenge to engineers because the file contains billions of polygon edges described in the physical design which have been sized and placed precisely to meet the particular technology design rules and device performance objectives (Zelnik 2002). Migration requires the ability to deal with every polygon in the file, adjusting each one in size and in relative spacing with respect to the millions of other objects in that physical design. With GDSII files of this size and complexity, manual efforts have become impractical and a single mistake can lead to design rule violations resulting in a non-manufacturable design (Zelnik 2002).

So, introduction of an automation tool for IP portability between different technologies, which offers a reliable and rapid conversion of IPs in a cost-efficient manner, would be of great value in the semiconductors market as it would increase their IC design productivity and time-to-market.

OPIC is a new EDA tool that is currently under development. It was designed for hard IP market to automate porting analog and mixed signal circuits within different technology process. Hard IP provides the fully designed, proven and tested circuitry needed to serve the increasing array of applications that

require high-performance digital or specialized analog/mixed-signal functionality (Zelnik 2002). OPIC tool can be used to do the circuit and IP porting of a layout in source technology A to target technology B by converting the given schematics and topologies from the source to the target technology.

As also explained in Appendix F: Papers Published , the tool takes an existing design layout in technology A which consists of network of transistors and is described by a geometrically exact GDSII file, the design spec for the new layout in technology B and the technical mappings (for both A and B). It generates a new layout design in technology B, which can be reviewed by a physical layout designer before it is sent out to the foundry for manufacturing.

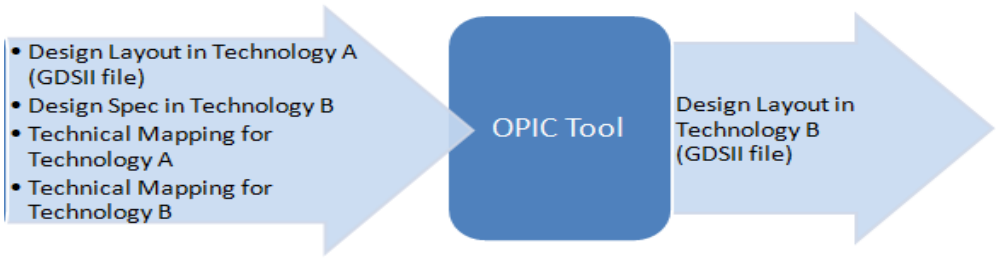


Figure 10: The OPIC tool

Input: Existing Layout Design in technology A in GDSII format, the design spec for the new layout in technology B, technical mapping for technology A, and technical mapping for technology B.

Steps:

1. Open a given layout design in technology A (in GDSII format) with the design spec for the new layout in technology B. Using the technical mapping for technology A, and technical mapping for technology B migrate the layout into new layout design into technology B.

2. Run the EDA tool called DRC on Layout of technology B to identify the violations of design rules.

3. Correct all the violations automatically and iteratively until the violations are cleaned.

4. Obtain 100% accurate (target) layout for new design in technology B.

Output : New layout design in technology B in GDSII format (binary file), which is ready to be given to IC foundries.

We can run simulations on this new layout design in technology B to test it further and send this layout to a physical layout designer for verification before the design is sent to the foundry.

MPIC is a long term project developed in phases, module by module. The end-goal is to create a new type of back-end design EDA tool which ports the graphical patterns (polygons) for each transistor and wire on the IC from one technology to another. The tool will check the geometries with a DRC tool against geometry design rules, iteratively, until the violations are cleaned. As a final product, it will generate an IC design in GDSII format, which is ready to be shipped to foundry for manufacturing

So far, DRC rule cleanup is the first module of OPIC tool that has been implemented and successfully tested. OPIC DRC cleanup tool was developed at UQO in the LIMA research laboratory. The tool was implemented using JAVA and it is compatible with major EDA tools, such as Cadence and Synopsys (Lakhssassi, Fouzar et al. 2016). The tool has been tested to ensure that it can correct DRC based on GDSII and the results (Table 2: Initial Results) have shown that the tool could reduce the tedious work of DRC checking from 25 hours of manual porting labor to 30 seconds. 65nm and 28nm technologies were the two platforms used for verification.

Reference Design	Errors #	Manuel correction	OPIC DRC-Cleanup
BIAS_GENERATOR	56 (7 type of violation: Overlap, spacing, Width, Extension, Clearance etc.)	50 minutes	30 seconds (24 CPU)
Oscillator (7x7 inverters 0,18um)	378 (7 type of violation: Overlap, spacing, Width, Extension, Clearance etc.)	240 minutes (no MACRO used assuming different errors and no compaction problem)	183 seconds (24 CPU)
Porting TestChip5 ($25mm^2$,18um to TSMC 0,18um)	TBD (PLL, DLL, BIAS_GENERATOR et le BANDGAP_GENERATOR etc..)	To do from scratch take 8 to 12 months	TBD

Table 2: Initial Results

A reference design including various analog and mixed-signal circuits will be built in 65nm technology and OPIC DRC tool will be used to determine if the layout satisfies a set of rules required for manufacturing to migrate it to 28nm. DRC cleanup is based on a proprietary algorithm and it includes 9 categories of common DRC errors that are related to spacing between metals, minimum width, etc. A DRC engine for correcting each category has already been created. The tool is able to recognize the error category and select the engine that will correct the error. Until all the errors are cleaned-up and the DRC is clean, the tool runs iteratively, without human intervention. Automated DRC cleanup tool can be used as a standalone product or can be integrated with major commercially available tools to perform a DRC cleanup. It will allow designers save considerable amount of time and use this time to focus on precision crafting their designs without sacrificing creativity to repetitive manual tasks.

Automated DRC cleanup tool represents a major step for layout porting. Next step is to create an LVS tool, which takes a schematic to identify and check IC electrical connectivity against IC schematics and makes corresponding changes to the layout. Ability to perform an automatic LVS, DRC cleanup is the major step for layout porting. Once fully implemented, OPIC tool will automatically perform both the LVS and DRC corrections.

This research project includes not only the development of an automatic porting tool but also the development of an entirely new methodology for more efficient and agile way of porting designs (including layout and schematics) from technology A to technology B.

4.6.2 Proposed IC Design model

At the present moment, in the IC design process, schematics and layout design phases are carried out sequentially. As can be seen in Figure 11: IC Design Model: Schematics and Layout phases in sequence, we start with the original reference design (for example 65nm), when the technology changes and we want for example 45nm, we need to:

1. first migrate the circuit specs and create new design rules (voltage is different, Vref(min, max));
2. get the schematics and circuit topologies for the new technology;
3. translate schematics and topologies from the reference design (65nm process technology) to target design (45nm process technology) via script;
4. port the schematics from the reference design (65nm) to target design (45nm);
5. verify schematics, then use the schematics to generate/port the layout from the reference design (65nm) to target design (45nm) and
6. verify the layout created in the target design (45nm).

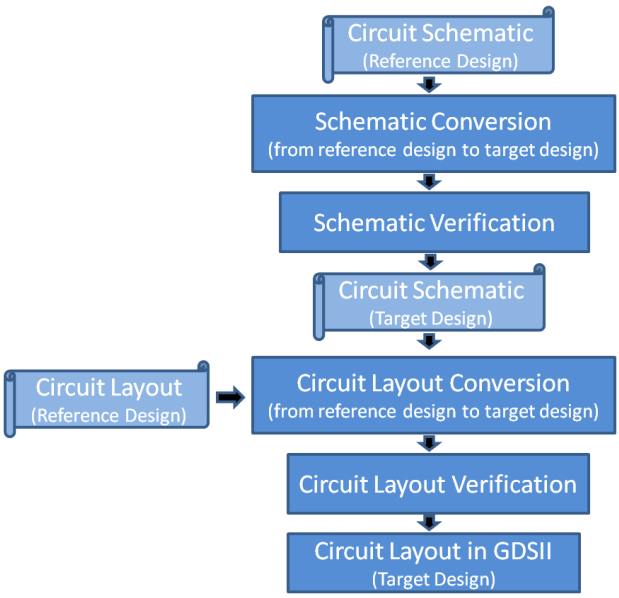


Figure 11: IC Design Model: Schematics and Layout phases in sequence

In this type of sequential design, neither we follow an agile development approach nor we take advantage of IP-reuse; design that works in one technology will not work in the other technology. So IP has to be completely redesigned when the technology changes, which leads to more errors in the new design, added time, cost and human resources. Just to give an idea, today, about 100 people work together to design a new IP.

Since chips are getting more complex and faster, efficient and reliable implementation of concurrency (i.e. implementing multiple tasks at the same time) will increase design productivity (Bryant, Cheng et al. 2001). This requires a change in the way we design ICs. As a part of the MPIC project, we propose an alternative, agile and more efficient IC design model where the schematics and layout design phases are carried out in parallel and the schematics created in the target technology can be used as an input to the layout creation process for target technology. Please refer to Figure 12: Proposed IC design model, below.

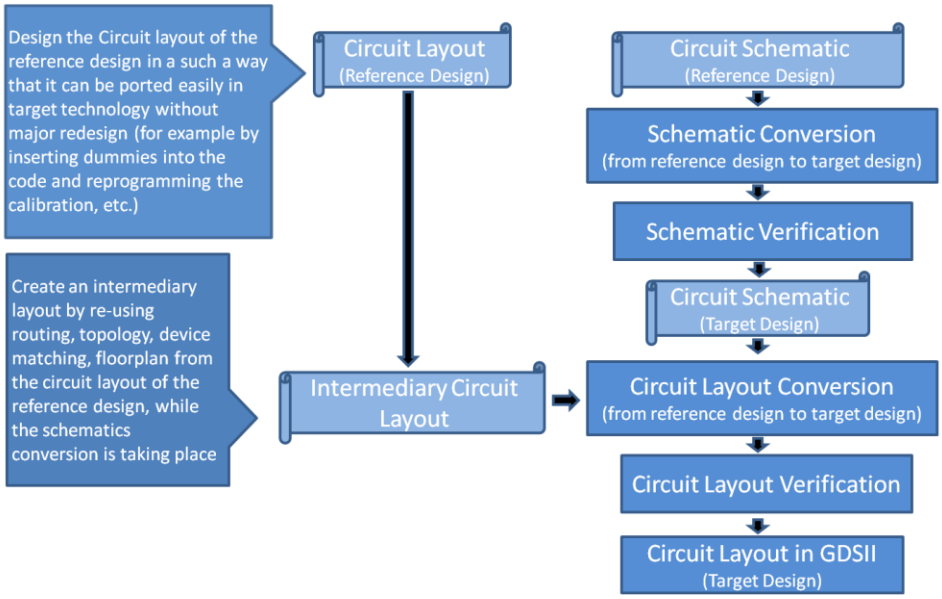


Figure 12: Proposed IC design model

For this to be achieved, first, we need to create a reference design that is flexible and conducive to IP re-use. This enables easy porting during technology change. Original IP reference design has to be designed in such a way that it can be easily re-used and ported into the new technology when the technology changes without having to change the layout.

Original schematic has to allow re-use, as well. When the technology changes (for example from 65nm to 45nm), we would like to be able to re-use IP without any major re-design. We need to design the IP so that it can be easily ported into target design (45nm) by looking at the technology file and using it against the reference design (65nm).

This proposed IC design model has to be integrated with a design methodology that would enforce IP re-use coding/programming guidelines for designers and programmers to follow. This design methodology would help them think of IP re-use when they design the ICs. For example, by considering adding dummies on the layout and/or using calibration, they could make their designs more re-usable. The key challenge for designers and programmers would be to develop an understanding of what it means to program a complex IC efficiently and learn to view the underlying hardware and input/output systems and comprehend the issue of concurrency. Therefore, we would need to impose certain restrictions on designers and programmers so that they would follow best practices without impacting their work. We could consider having a checklist in place or to have software to not allow designer to release the design if IP cannot be re-used.

4.6.3 Proposed Layout Porting Methodology

MPIC is based on pure software development to automate the process of designing, verification and porting of analog and mixed-signal IP. The development is based on three major phases:

- Schematic porting and full verification using full automation
- Layout database build-up and mapping to different technologies
- Test on real circuit designs using different technologies available in the LIMA library, like 65nm and 28nm.

4.6.3.1 Schematic Porting methodology

The schematic design porting will be based on g_m/I_D- based sizing methodology. The g_m/I_D sizing methodology was first introduced by the paper of Silveira et al. (Silveira, Flandre et al. 1996) as a resourceful tool for performing sizing. The objective of this technique is to quickly and accurately size any linear analog circuit, top–down, from some required specifications and evaluate the remaining ones. This method exploits the transconductance (g_m) over DC drain current (I_D) ratio relationship versus the normalized current [I_D / (W/ L)]. The ratio of g_m/I_D, which is the transconductance-to-current-ratio (should be high), controls the gain and power consumption. The larger the g_m/I_D, the smaller the drain current and the larger the gain (Silveira, Flandre et al. 1996). The g_m/I_D ratio versus the normalized current i = [I_D / (W/ L)] is an intrinsic MOS characteristic, which indicates the inversion level of the transistor (i.e. strong, moderate or weak inversion) (Silveira, Flandre et al. 1996).

In CMOS analog and mixed-signal design style the higher the trans-conductor efficiency (g_m/I_D ratio) the better the design (Silveira, Flandre et al. 1996). It is the efficiency of the transistor to translate given current into an equivalent transconductance. So to quantify how good of a job our transistor does and to generate efficient designs, we propose to use g_m/I_D sizing methodology.

The choice of g_m/I_D is based on its relevance for the three following reasons:

- It is strongly related to the performance of analog circuits
- It gives an indication of device operating region (g_m/I_D methodology is based on a unified synthesis methodology in all the regions of operation of MOS transistor)
- It provides a tool for calculating the transistors dimensions

Hence, it is proposed to capture this methodology in a fully automated approach under MPIC. This method will generate efficient designs regardless of the technology and hence it will help the circuits to be portable between different technology nodes.

4.6.3.2 Layout porting methodology

The layout of reference design (from reference technology A) is used to generate any subsequent layouts in different technologies. Reference layout incorporates a knowledge that has to be used to build ported layouts. The knowledge could be, for example, the floorplan and device matching, as well as routing. In the MPIC project, we propose to build a universal reference database for layout, which can be used to map into other technologies. It allows fast modification in a new technology to reflect the schematic changes, as well. This process is very different than that of any commercially available tool that is usually based on a layout data stored in a file. Available commercial tools usually store the layout data in a file. When the layout file becomes large, for example on the chip level, these tools start to behave abnormally, such as the viewing of the layout becomes slow or the layout session crashes. Having an universal layout database that is built by using any viable database, such as MySQL, will allow a flexible layer mapping between technologies and will remove the dependency of the layout to a tool. Having an efficient and easy to create database structure will also ensure compatibility and consistency. Once this database is built, designers can reuse it with any commercial tool of their choice and will be able to modify it easily in a new technology to reflect schematic changes.

We will also create a library of functions and of hardware and software implementations that can be used for all new designs. It is important to have a multilevel library, since the lower levels that are closer to the physical implementation would change due to the changes in technology while the higher levels could be stable across different technologies. This way we can assemble components from the library with little effort.

4.6.3.3 Testing the porting methodology

In order to test the new methodology, we propose to build a reference design along with its layout and verification platform (test benches for simulation) in TSMC[4] CMOS 65nm technology. Next step is to start the porting in different technologies, such as the FDSOI[5] 28nm by STMicroelectronics. The duration

[4] Taiwan Semiconductor Manufacturing Company

[5] Fully Depleted Silicon On Insulator, or FDSOI, is a planar process technology that delivers the benefits of reduced silicon geometries while actually simplifying the manufacturing process.

and number of iterations will measure the effectiveness of the porting methodology. Note as well the simulation results will be compared to specifications for FDSOI 28nm given the porting methodology works as planned.

4.6.4 Proposed System Development Methodology (SDM)

To determine the most suitable SDM that should be used in the MPIC project, The New MAF Conceptual Framework that has been introduced earlier in the document has been used. For each of the seven factors or dimensions (*Outcomes*, *Scope*, *CYNEFIN*, *Constituents*, *Principles*, *Team*, and *Organization*) that the MAF is based on, a series of self-evaluation tools have been developed, each of which are described below.

As it will be explained in the section below, the MPIC project was evaluated against these seven factors by using the self-evaluation tools and the evaluation results for each factor were demonstrated in the table below and visually using radar charts to determine whether an agile, plan-driven or hybrid SDM would be more suitable for the project.

The MAF suggests that an agile methodology is best suited for the MPIC project because the project scored very low on *Outcomes*, *Scope* and *Constituents* dimensions while reasonably high on the *CYNEFIN*, *Organization*, *Team* and Agile *Principles* dimensions as demonstrated in the table below.

Table 3: The MAF evaluation results for the MPIC project

MAF DIMENSIONS						
Outcomes	**Scope**	**Constituents**	**CYNEFIN**	**Organization**	**Team**	**Principles**
Important	Small with low complexity	Few constituents with basic role (*Inform*, *Consult* or *Educate)*	Complex	Established	Advanced	100% supports agile values and principles

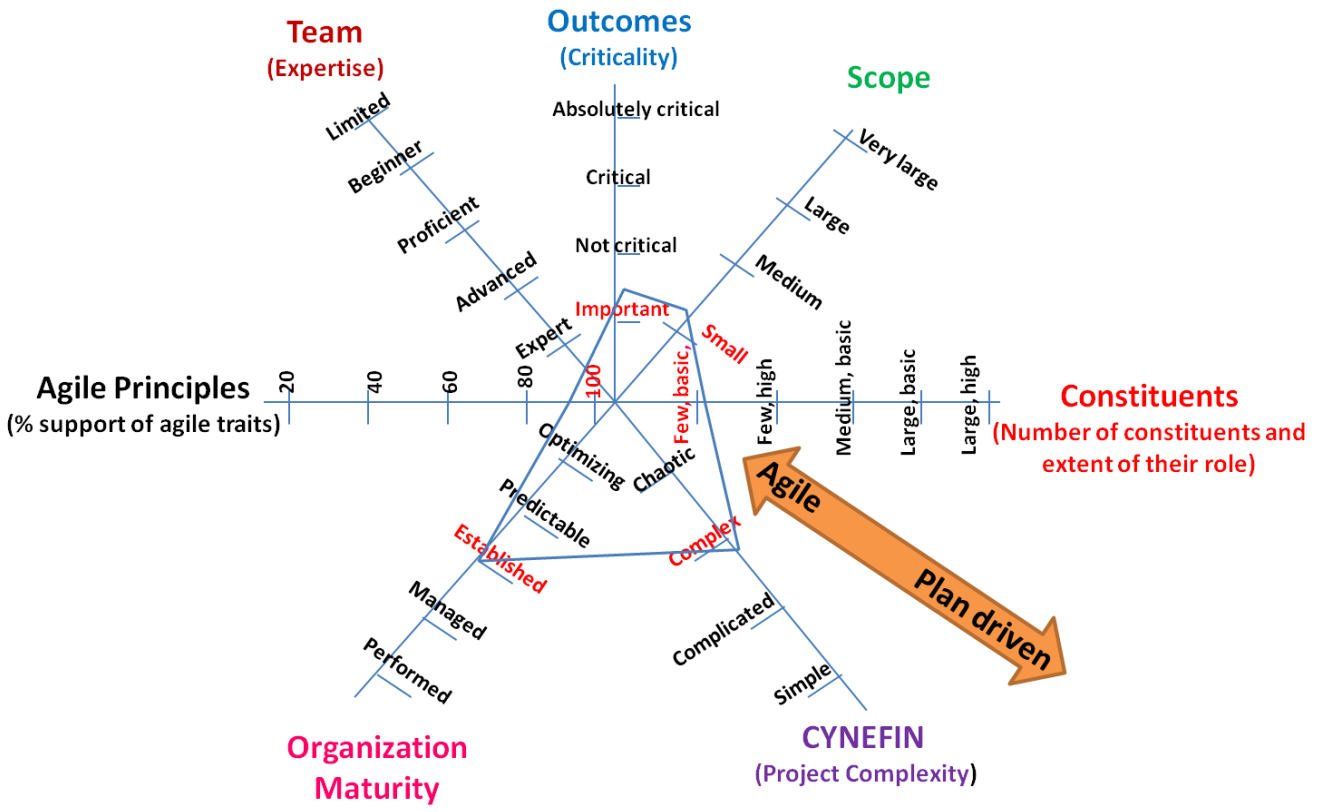


Figure 13: The MAF evaluation of the MPIC project using Radar Chart

Each factor, their assessment metrics and evaluation results for the MPIC project are discussed below.

4.6.4.1 Outcomes – how critical?

By design, a "system" uses resources as inputs and turns them into outputs in order to make an organization hopefully more efficient. It is important to highlight the difference between outcomes and outputs. Allen makes the distinction between outputs and outcomes by stating that "Outputs are the goods and services that result from activities. Outcomes are the constructive impacts on people or environments" (Allen 2019). When the Chief Executive officer of a company asks his/her Chief Information Officer how a specific project will contribute towards the attainment of the organization's mandate, he/she means how will this project explicitly contribute to the strategic outcomes of the organization. If the CIO has no answer or can only say that it will save some time and money, he may not be listened to for a very long time or paid any money.

In this research, the goal is to select a suitable SDM methodology for a given project. The degree of fit between the SDM and the project will be demonstrated through the outcome improvements, in terms of social, economic and organizational capacities, such as learning, understanding, benefits and economic

changes (Allen 2019; Mullaly and Thomas 2009). According to Saunders, "the means of achieving the desired outcomes in real life situations could be provided by the use of the suitable systems methodology" (Saunders 1992).

The following assessment grid aims to assess the importance of *speed*, *innovation*, *reliability*, *security* and *efficiency* aspects of the project outcomes. I argue that the more critical the outcomes associated to the project the more important it is to have solid and proven system development methodology and that the right set of constituents be part to the project. Therefore, the higher a project scores on this dimension (Level 3 or 4), the more likely that a solid, plan-driven methodology will be required.

Note 1: For each outcome, select the appropriate rating.
Note 2: Total all points and use the following scale to identify your In-Range Level:

- 0 to 4 points = level 0 (Somewhat important)
- 5 to 9 points = level 1 (Important)
- 10 to 13 points = level 2 (Important but not critical)
- 14 to 17 points = level 3 (Critical)
- 18 to 20 points = level 4 (Absolutely critical)

HOW APPLICABLE ARE THESE OUTCOMES TO YOUR PROJECT?	**0 point:** not applicable	**1 point:** somewhat important	**2 points:** Important	**3 points:** important but not critical	**4 points:** absolutely critical
FAST: the faster we reach new markets, the higher the probability that we will grow or bottom line and capture market share.			2		
INNOVATIVE: innovative products are critical to the on-going success of our organization.				3	
RELIABLE: availability 365/12/7/24 without interruptions at the expected levels of performance secures our market's position and benefits.			2		
SECURE: Securing client data and information from unauthorized access and use has a direct linkage on the value of our shares and assets.	0				
EFFICIENT: To maintain our profitability, we must reduce our cost to the minimum without risking major financial issues.			2		

GRAND TOTAL: 9
IN RANGE LEVEL: 1
CAVEAT: The scores that are presented in the table above represent the evaluation results of the MPIC project. For the MPIC project, the *speed*, *reliability* and the *efficiency* aspects of the project outcome,

which is the successful implementation of the OPIC tool, are important, while the *innovation* aspect is important but not critical and the *security* aspect is not applicable. As a result, with the grand total score of 9, the project falls into the category Level 1, which indicates that an Agile methodology would be better suited.

4.6.4.2 Scope of the project

The Project Management Body of Knowledge (PMBOK), defines scope as the "sum of products and services to be provided as a project" to ensure that the project includes all the work required, and only the work required, for completing the project successfully (PMI 2017). In other words, scope defines the work required to complete the project successfully. Scope of the project is related to the complexity profile of the project and it impacts the project constituents as there is a need to develop and maintain a common understanding of what products or services the project will deliver (Darnall and Preston 2010).

"The scope of the project is one of the key software process determinants"(Alqudah and Razali 2017). It relates to the project size, competencies and experience it will require, number of organizational divisions to be involved and the environmental forces (political, economical, social, technological, environmental, legal) that the project will be impacted by. Therefore, it is almost a given that the scope of a given project will play a role in selecting the proper methodology such as Agile or not. At the extremes of the spectrum lie a simple software update project and the development of a new set of applications with significant and complex linkages. In the former case, Agile might be the method of choice while in the latter, a more formal methodology should likely be used. As Griffiths states "While an agile method can work well on life-critical systems, it takes much more skill and effort to implement. Agile is much easier to use on small, non-life-critical applications" (Griffiths 2013). The projects that seem to benefit most from agile approach are the ones that develop a new system for a totally new environment (van Casteren 2017).

The assessment grid below provides you with a way to define the scope of your project. By answering each of the 20 questions captured in the table below, you will identify the base score for each element. Next, you choose an appropriate weight factor for each element and then multiply the base score of each element by its weight factor to obtain a final score of the element. When all the final scores are added up, this will provide the scope assessment for the project. A low score (0 or 1) for the scope assessment would indicate that Agile methodology would be better while a high score (3 or 4) would indicate that formal methodology like Waterfall should be used for the project.

Note 1: Use the following scale to identify the base score for each question. Your score is the selected base score times the appropriate weight factor (1, 2 or 4) for the question.

0: not true, appropriate or applicable to this project
1: To some degree true, appropriate or applicable to this project
2: Somewhat true, appropriate or applicable to this project
3: Mostly true, appropriate or applicable to this project
4: Totally true, appropriate or applicable to this project

Note 2: This self-evaluation grid uses weighted scores. The scale used (weight factor: 1 - 2 – 4) reflects and amplifies, *based on my work experience*, the relative influence of the various factors.

1: important
2: quite important
4: critically important

Note 3: The table below provides you with the range score that will be used to determine scope size of the project.

TOTAL SCORE	IN RANGE LEVEL
Less than 50 points	0 – Very Small / Very-low complexity
Less than 70 points (0-29%)	1 – Small / Low complexity
71 and 120 points (30%+)	2 – Medium /Medium complexity
120 and 190 points (55%+)	3 – Large / High complexity
More than 190 points (80%+)	4 – Very large / high complexity

ID	ELEMENTS	QUESTIONS	BASE SCORE	WEIGHT FACTOR	TOTAL SCORE
1	**Competencies**	Does the project involve many IM/IT competencies such as architects, modellers, telecom experts, programmers, other(s)	2	4	8
2	**Data elements**	Does the system include the management of a very large number of data elements, datasets, data interfaces, marts and warehouses?	2	4	8
3	**Dependencies**	Is this project dependant/linked to other projects, databases and IT infrastructures?	2	4	8
4	**Environment**	Will the project be carried out in difficult work conditions such as poor accommodations, extra long work hours, health and safety risks, etc.?	0	1	0
5	**Financial**	Does the project require big investment; tight and specific financial conditions?	1	4	4
6	**Geography**	Is the project team geographically dispersed? Different languages?	1	2	2

7	**Leading Edge Technologies**	Does this project involve leading edge technologies?	0	2	0
8	**Legal**	Are there legal requirements to be taken into accounts in the requirements or project as a whole?	0	1	0
9	**Multi-disciplinary**	Does this project cross many fields of organizational divisions and business such as financial, human resources, legal or other(s)?	0	2	0
10	**Nature of System Work**	Is this mostly a technology project?	3	1	3
11	**Process**	Is this project the automation of a complex set of business processes and rules?	3	2	6
12	**International**	Are there any countries, other than yours involved in this project (e.g.: USA, Canada, China?)	1	1	1
13	**Security**	Is there a need to secure / encrypt information to limit / control access?	1	4	4
14	**Technical Integration**	Does this project integrate many technologies or platforms such as cloud computing, encryption, business intelligence, other(s)?	1	4	4
15	**Time**	Is the project time sensitive?	3	4	12

GRAND TOTAL: 60
IN RANGE LEVEL: 1
CAVEAT: The scores that are presented in the table above represent the evaluation results of the MPIC project. As demonstrated by the answers to each question, the grand total of 60 qualified the project for the category Level 1 and the scope of the MPIC project has been evaluated to be *small with low complexity*. Therefore an agile methodology would be better suited for the project.

4.6.4.3 CYNEFIN of project

Project managers need to deal with the complexities of projects in practice in order to "improve the likelihood of project success or at least to understand the reasons for failure" (Mikkelsen 2018). However, the assessment of complexity is subjective and will be influenced by how the project manager perceives and responds to it (Geraldi, Maylor et al. 2011). The complexity level of the project will influence the decision of what kind of methodology would give us the best chance of success with this project (Mikkelsen 2018).

CYNEFIN (a Welsh word meaning habitat) is a conceptual framework that was developed by Dave Snowden in 1999 to help decision making based on the project complexity level (Snowden & Boone, 2007). It has evolved over the years but the project habitat (CYNEFIN) has not changed much. We find that the "simple – complicated – complex – chaotic – disorder" nomenclature can easily be used to assess a project environment. The "CYNEFIN picture" of a project will, with other factors, ensure the selection of the right development methodology.

In this research, CYNEFIN was chosen the framework to assess the "nature" and the complexity of the project. Knowing the complexity of a given project will help us determine the right system development methodology (traditional, hybrid, agile) to be used to reduce the complexity and to achieve desired project outcomes (Mikkelsen 2018). There are, of course, other ways and methods to assess the nature of the project and there will always be people arguing the pros and cons of a specific method. The key point to keep in mind is that none is perfect, and their goal is simply to ***help*** you assess the complexity level of the project.

According to CYNEFIN, a project will be born in the "Disorder" state and go through the "Chaotic", the "Complex", the "Complicated" and end in "Simple" (Mikkelsen 2018).

In traditional methods, stakeholders agree on deliverables, create a WBS, make schedules and execute the plan. Whereas in agile methods we focus on reducing complexity one sprint at a time (Mikkelsen 2018). So, the waterfall is not suitable is requirements are not well-understood/defined or likely to change during the course of the project (van Casteren 2017). According to Apke, "most software development is complex and that is the reason that Agile works well and is generally preferable to Waterfall. Those projects that might benefit from Waterfall are those that are complicated, those where all the answers can be known up front and experts are effective" (Apke 2020).

While agile can be used in "simple", "complicated" and "complex" projects, it works best for "complex" projects, which have some uncertainty around both requirements and technology but not so much that they are chaotic or impossible to get our hands around (Griffiths 2018). According to Griffiths, the "simple" and "complicated" projects can benefit from the benefits of increased collaboration, communication and visibility aspects of the agile methods, but these kinds of projects can also be run with a traditional approach. "Complex" projects, on the other hand, become a struggle if team tries to use traditional methods. According to Mikkelsen, as we can foresee the future in the "complicated" domain, the waterfall model would be a better choice, whereas in the "complex" domain, a better choice would be an agile

approach with flexibility and adaptability (Mikkelsen 2018). Boehm & Turner assert that agile is more suitable for "trivial applications where failure of the system results in a loss of convenience, such as losing personal time if a video game crashes or losing work time if a word processor fails (Boehm and Turner 2004). However, for mission-critical or life-critical applications agile would be less applicable.

The following table provides you with elements, which will help you to assess the CYNEFIN "habitat" of your project to determine if it is going to be a *simple*, *complicated*, *complex* or *chaotic* undertaking. For *Simple* and *Complicated* projects (ratings of 0 and 1) waterfall methods should be preferred while for the *complex* or *chaotic* projects (ratings of 2 or 3), agile would be better suited.

Note 1: Identify the level that best describes your CYNEFIN habitat based on the definitions listed in the table below.

1: Simple
2: Complicated
3: Complex
4: Chaotic
0. Disorder

LEVELS	DEFINITIONS	THE LEVEL THAT BEST DESCRIBES YOUR CYNEFIN HABITAT
SIMPLE **"Just-do-it"** **Example: Cooking an omelet or a steak**	• Encompasses some basic issues of technology, techniques, expertise and terminology – scaling is not an issue. • Problems and answers are well known • Requirements are clear and stable • The relationship between cause and effect is obvious • There is one or a few right answers • Many similar projects delivered successfully • It can be considered as a standard practice / operation • Often compelled to use a vendor's method	
COMPLICATED **"Plan it"** **Example: Building an electronic printed circuit, building a house or car, banks, manufacturing public schools, healthcare providers**	• Scaling is an issue but there also is a significant need for coordination, technology and/or specialized expertise. • The problem is open ended with a range of solutions • Requirements are clear • Well-defined relationship between cause and effect • Focuses on the content that is knowable and therefore possible to plan • Owners and users are known and numerous • Large number of known requirements and rules • A range of possible answers • Requires significant analysis and investigation	

COMPLEX **"Frame it"** **Example:** **Stock markets, New Product Development, Innovation/Invention**	• People, relationships and their properties of self-organization, interconnections and evolution are key drivers and triggers. • Problems and solutions are evolving • Requirements are clear • Owners, partners, clients and users are changing • There is no known right solution • Innovation and experimentation are required • Dealing with new technology	3
CHAOTIC **"Survive it"** **Example:** **Negotiating a peace treaty in the middle east**	• Unpredictable behaviours, often triggered by small changes in conditions, result in a constant state of change. • Unclear what the requirements are • Requirements are evolving • No answer / solution seems to satisfy everybody • Constituents are often or constantly changing • Creativity is the only possible avenue	
DISORDER	• Complete solution impossible • Attempts to develop solutions are turned down • Requirements are difficult to define • Feelings and emotions run high in the team – it is on the verge of breaking up • Sponsors are becoming uninvolved or becoming negative	

GRAND TOTAL: not applicable
IN RANGE LEVEL: 3
CAVEAT: The scores that are presented in the table above represent the evaluation results of the MPIC project. According to the CYNEFIN framework, MPIC project is considered to be a *complex* project because the solution will be discovered by developing a safe environment for experimentation, through which we will be able to probe (explore) and sense (inspect) and then will be able to create emergent solutions for the development of the OPIC tool. As a result, an iterative development method (agile methodology) would work best for this type of project.

4.6.4.4 Constituents of the project

Public and Private sectors program and project management best practices clearly point to the people aspects of a project as the key elements of success. The following list of success factors illustrates that point – in a bracket, the ones that relate to people/relationship management were identified.

1. Be 100% aligned with business (**people**)
2. Have the commitment and involvement of executives (**people**)
3. Have a strong Governance in place (**people**)
4. Have a Project Management Office

5. Enforce Project Planning
6. Ensure that trained and competent resources are available (**people**)
7. Manage risks and complexity
8. Use a phased approach
9. Involve users from day one (**people**)
10. Measure and report on performance
11. Carefully manage and track scope
12. Communicate, communicate, etc... (**people**)

Of these 12 success factors six (50%) have a strong people “flavor" and very few have a system “flavor." This is not, far from it, to minimize the importance of system development, but to highlight the importance of people in ensuring success. In a way, we can say that people make a project successful or not.

It is important to highlight the difference between constituents and stakeholders. Why talk about constituents and not stakeholders? While most IM/IT executives used the term stakeholders as an all-encompassing term, it is not. Stakeholders are those that have a stake in the outcome of the project and they are affected *indirectly* by the project outcomes. While constituents are those whose voice matter the most, they are the active partners, co-creators who play a direct role in the project and who are *directly* affected by the project outcomes (Keystone 2018).

Constituent relationship management is a structured approach to manage who; what; how; for what purpose of interactions between the project team and the project players. In other words it refers to all relationships associated with all aspects of a project. To deliver the project outcomes that reflect the actual needs of the constituents, we need to work closely and collaboratively with them to identify what they think they want, produce something which reflects that understanding, get feedback from them, and then update our solution to reflect our improved understanding (Ambler 2018). Collaboration and knowledge sharing are crucial both in plan-driven/traditional and agile methodologies. However, in agile knowledge sharing active collaboration and communication are viewed as the key components. In fact, Agile manifesto(PMI 2017) highlights active collaboration in both its 3rd value (“Customer collaboration over contract negotiation”) and 4th principle (“Business people and developers must work together daily throughout the project”). In agile, constituents need to communicate frequently to ensure that everyone is on the same page and kept up to date, since many project failures can be traced back to a failure of communication (Griffiths 2018).

There is a distinction between various types of constituents as each has unique attributes.

- **Advisor(s)** – the people with more and deeper knowledge in a specific area and with cross functional and multidisciplinary expertise
- **Business Owner(s)** – executive who is responsible for the design, development, operations and control of the business processes and supporting tools associated with the project
- **Client(s)** – senior executive(s) that has ultimate authority over the project and provides funding and strategic directions.
- **Community of Interest** – ultimate users of the project deliverables
- **Executive Sponsor** – the vocal and visible project champion who develops and presents the project business case, resolves issues and scope changes, approves major deliverables and provides high level direction to move the project forward
- **Partner(s)** – departments and/or organizations that directly contribute to the delivery of the project and that agree to cooperate to advance their mutual interests
- **Project Enabler(s)** – Organization that are actively engaged/involved and directly contribute to the development, delivery, operations and maintenance of a system or project
- **Public** – individuals, groups of people that either have a keen interest in the project or will be impacted by the project, in one way or another
- **Service Provider(s)** – departments and/or organizations that directly contribute to the delivery of administrative and operational support services and with whom the client has Memoranda of Understanding and/or Service Level Agreements
- **Stakeholder(s)** – Organizations that are affected by a project course of action and results and consequently have a vested interest and are engaged/involved in how the project unfolds
- **Development Team** – Team whose members are assigned to activities within the project until the project is deemed complete.

Relationship Management also makes the distinction between the depths of relationships for any given constituent. While the difference between various terms may appear to be only semantic, the differences are real and have an impact on defining and agreeing on who does what and how and the amount of resources needed to carry out an activity.

- **Approve** : To officially agree to or accept as meeting requirements
- **Consult**: To seek the views of persons or groups of persons on matters affecting them

- **Develop**: To bring to existence or to make more mature a process or system.
- **Educate**: To transfer knowledge, skills and habits from an individual/group to others through teaching, training or research
- **Engage**: To reach out to provide selected groups of people with the opportunity to influence the decision-making process as well as the project outcome, objectives, deliverables, design and implementation
- **Inform**: To exchange thoughts, messages or data and/or information by speech, visuals, writing or behavior
- **Involve**: To be included to contribute in the project due to their specific knowledge, competencies and abilities.

It is important to note that the type and depth of relationship varies from constituent to constituent and often from activity to activity. For example, you could simply inform the Community of Interest that training will take place, or you could involve them in developing the training material. Having a clear agreement on who does what, when, with whom and, for what purpose is fundamental to project management relationship and ultimately to project success.

As the number of constituents involved in a project increases, the complexity of the project increases because understanding, managing and leveraging the relationships between the constituents increases exponentially due to the $n^*(n-1)/2$ formula that calculates the number of communication channels between relationships. Agile development methodology would be more suitable for small and medium size projects with few constituents and only a few basic rules, whereas the large size projects that involve large number of constituents and extended number of roles will benefit more from plan-driven/traditional development methodologies.

The biggest limitation of agile methodologies is how they handle larger development teams. Cockburn and Highsmith both argue that “Agile development is more difficult for larger teams…as size grows coordinating interfaces become a dominant issue”(Cockburn and Highsmith 2001). Boehm &Turner also agrees that “teams of less than ten are a great fit for agile approaches as they can communicate face to face, support tacit knowledge by conversations and facilitate simple, visible tracking systems. As team sizes grow, supporting these agile principles requires additional techniques. It can be done but it takes more work and skill” (Boehm and Turner 2004).

De Lucia & Qusef also agreed that agile works well for small to medium sized team and he argued that the smaller the agile team, the higher the chances of the project success. Because when the team size grows communications and the requirement changes becomes more difficult and complex (De Lucia and Qusef 2010). Both Constantine and Martin Fowler also believe that agile with face-to-face communication breaks down and becomes more difficult and complex with development team size that exceed 20. In contrast, plan-driven, traditional methods scale better to large projects with large number of constituents with extended number of roles.

To help assess the depth and scope of the constituents' management, the following, simple evaluation grid has been proposed. The constituents were grouped by affinities to make the assessment faster. With regards to constituent management, there are many methods, processes and management tools widely used. RASIC (Responsible, Approve, Support, Inform, Consult) is one of the best known but there are many others like PARIS, PACSI, RASCI, RASI, RACIQ, and many others. The approach used in this research is unique, but you could replace it with another one that you are more familiar with.

Note 1: For each element, insert the number of constituents (but not the number of people in a constituency).

Note 2: For each element, determine the type of role being basic (*Inform*, *Consult* or *Educate*) or an extended role (*Develop*, *Engage*, *Involve* or *Approve*) which will be used as a weight factor. The scale used (doubling: 1 - 2 - 4) reflects and amplifies, *based on my experience*, the relative influence of the various constituents.

1: Inform
2: Consult or educate
4: Develop, engage, involve or approve

Note 3: Multiply the number of constituents of each element by its weight factor to obtain a final score for the element. The table below provides you with the range score that will be used to graphically present this element. The lower assessment score and the in range level (0 or 1) would indicate that an agile methodology would work best for the project while the higher score and the in range level (3 or 4) would mean a traditional/plan-driven methodology should be used.

Grand Total Score	IN RANGE LEVEL
Less than 50 points	0 – Small size project, few constituents, most of which have basic influence
Less than 100 points	1 – Medium size project, few constituents, most of which have high influence

100 and 200 points	2 – Medium size project, medium size constituents, most of which have basic influence
200 and 300 points	3 – Large size project, large number of constituents, most of which have basic influence
More than 300 points	4 – Large size project, large number of constituents most of which have high influence

ELEMENTS	NUMBER OF CONSTITUENTS	WEIGHT FACTOR	TOTAL SCORE
Client(s) / Executive Sponsor / Business owners	2	4 (Develop/Engage / Involve / Approve)	8
Development Team(s)	5	4 (Develop/Engage / Involve / Approve)	20
Stakeholders – Community of Interest – Public	10	1 (Inform)	10
Project Enablers / Partners / Service providers	5	2 (Consult / Educate)	10
TOTAL	**22**	**-**	**48**

GRAND TOTAL: 48
IN RANGE LEVEL: 1
CAVEAT: The scores that are presented in the table above represent the evaluation results of the MPIC project. As can be seen from the grid, MPIC project is a small size project with few constituents, most of which has basic influence (*Inform*, *Consult* or *Educate*). Therefore, the project qualifies for category Level 1 and an agile methodology would work best.

4.6.4.5 Applicability of Agile Principles

This indicator characterizes the agile traits based on the set of twelve principles provided by Agile Manifesto (Fowler and Highsmith 2001) and examines the support of agile values and principles. This criterion intends to evaluate the degree of agility because there exist certain characteristics that are inherently associated with agile methodologies which can be used as evaluation criteria(Taromirad and Ramsin 2008). As a part of the MAF framework, each one of these principles must be reviewed to assess its desirability or applicability to the project, organization and culture.

The following segment presents you with the assessment results of the agile principles indicator for the MPIC project. Each decision maker can follow the same steps to complete the assessment metrics of this indicator for their project.

Note 1: to assess principles against your project, please use the following scoring approach for each principle.

0: not true, appropriate or applicable to this project
1: To some degree true, appropriate or applicable to this project
2: Somewhat true, appropriate or applicable to this project
3: Mostly true, appropriate or applicable to this project
4: Totally true, appropriate or applicable to this project

Note 2: total up your individual scores
Note 3: use this grid to identify the In-Range Level. A high rating (3 or 4) would indicate that an agile methodology is better suited for the project, while a low rating (0 or 1) would indicate a plan-driven methodology like waterfall should be chosen.

Grand Total Score	IN RANGE LEVEL
0 to 9 points	0 – 20% supports agile values and principles
10 to 19 points	1 – 40 % supports agile values and principles
20 to 29 points	2 – 60 % supports agile values and principles
30 to 39 points	3 – 80% supports agile values and principles
40 to 48 points	4 – 100% supports agile values and principles

AGILE PRINCIPLES	HOW IT DOES OR DOESN'T APPLY TO THE PROJECT	PROJECT SCORE
#1: Customer satisfaction by rapid delivery of useful software.	The key project outcomes are the delivery of a timely, fully functional, stable, efficient OPIC tool application.	3
#2: Welcome changing requirements, even late in development.	The project will be developed in iterations. Requirements are mostly known at the beginning of the project. Late changes are welcome and they will be discussed during the sprint planning meetings and assessed vis-à-vis the team's capacity to meet the project outcomes.	1
#3: Working software is delivered frequently (weeks rather than months).	Interim software delivery through sprint planning and sprint reviews will ensure that the project is on track towards achieving its goals.	3
#4: Close daily cooperation between business people and developers.	While the cooperation is most important close and daily cooperation is not possible as most of the team members work on the project on a part-time basis. Frequently scheduled JAD sessions will be held to ensure that developers adequately interpret business requirements. With daily scrum meetings	3
#5: Projects are built around motivated individuals, who should be trusted.	Fully agree with the principle and the project team members are all motivated individuals but validation by key stakeholders is still required to ensure integration at all levels.	4
#6: Face-to-face conversation is the best form of communication (co-location).	Since the size of the development team is quite small, co-location is possible. However, since the team members have full-time jobs and work on this project on a part-time basis, weekly meetings and conference calls will be used as often as possible.	3

#7: Working software is the principal measure of progress.	Very important. Working software as an outcome defines success for this project. The solution will be developed in sprints iteratively and incrementally.	4
#8: Sustainable development, able to maintain a constant pace.	Very important. The project cannot be sidetracked by other initiatives if the outcomes are to be met in the project timeline. Through sprint planning meetings, daily scrum meetings and sprint review meetings, the software will be developed in a constant pace.	4
#9: Continuous attention to technical excellence and good design.	Nobody can argue with this principle.	4
#10: Simplicity—the art of maximizing the amount of work not done—is essential.	Nobody can argue with this principle.	4
#11: Self-organizing teams.	Self organising team is a key to the success of the OPIC project in order to make the best decisions with regards to architecture, requirements and design.	4
#12: Regular adaptation to changing circumstances.	Team is quite small to be able to adapt to changing circumstances. During the sprint planning meetings, late changes, depending on their nature, will be assessed vis-à-vis the capacity to meet the project outcomes.	3

GRAND TOTAL: 40
IN RANGE LEVEL: 4
CAVEAT: The scores that are presented in the table above represent the evaluation results of the MPIC project. As can be seen from the grid above, MPIC project fully supports the agile principles. With a grand total of 40, the project qualifies for category Level 4, which means that, it is highly suitable for an agile methodology.

4.6.4.6 Team expertise

The success of agile depends on highly motivated and skilled people because documentation is very lightweight and most of the knowledge is tacit (Eberlein and Leite 2002). Actual implementation is left to the developers who work as self-organizing teams, without providing clear guidance and details on what needs to be done. Boehm & Turner suggests “a critical mass of highly talented people” as one of their five critical factors which can be used to determine the suitability of agile or traditional methods for a particular project (Boehm and Turner 2004).

This study has identified the following as the key elements of team expertise:

- Knowledge of and experience in IM/IT
- Knowledge of and experience in Project Management
- Knowledge of and experience in system development methodologies
- Knowledge of and experience in project contracting and management

- Capacity to work in a team environment
- Capacity to work under stress
- Capacity to communicate orally and in writing

Agile development relies on the tacit knowledge of team members. According to Alistair Cockburn (Boehm and Turner 2004), agile development demands experienced team members, who can "revise a method to fit an unprecedented new situation" and "tailor a method to fit an unprecedented new situation", perhaps because the tacit nature of information flow demands a higher level of expertise. According to Boehm & Turner (Griffiths 2013), agile projects are more likely to go smoothly with a low proportion of beginner developers and high proportion of proficient, advanced and expert level practitioners. If team has a higher percentage of beginners then a more traditional approach may be more successful (Griffiths 2013).

The self assessment grid that is shown below provides the definitions and their associates score value for *Limited*, *Beginner*, *Proficient*, *Advanced* and *Expert* level team members. When the team self assessment scores are 2, 3 or 4, that means that the team is strong and experienced enough to successfully use an agile methodology while an assessment score of 0 or 1 would indicate that a plan-driven methodology would be better suited.

RANGES AND DEFINITIONS	YOUR SCORE
LIMITED (SCORE 0): the team lacks in most if not all elements of knowledge and expertise.	
BEGINNER (SCORE 1): the team has some of the elements of knowledge and expertise but lack in many.	
PROFICIENT (SCORE 2): the team meets most knowledge and expertise elements, but help will be required to complement some elements.	
ADVANCED (SCORE 3): the team meets all elements although some improvement or help might be required in some areas.	3
EXPERT (SCORE 4): the team meets all elements and is considered a model by others.	

GRAND TOTAL: not applicable
IN RANGE LEVEL: 3
CAVEAT: The scores that are presented in the table above represent the evaluation results of the MPIC project. As can be seen from the grid above, MPIC project qualifies for the category Level 3, as the team is composed of members that are considered to be advanced in their fields of expertise. As this is a research

project which aims to introduce a new software tool called OPIC, some improvement or help might be required in some areas. In other words, an agile methodology can be successfully used for this project.

4.6.4.7 Organization's maturity

This indicator aims to measure the capability of an organization to provide the supporting environment conducive to the implementation of an SDM. Kerzner (Kerzner 2017) defines the maturity as "the implementation of a standard methodology and accompanying processes such that there exists a high likelihood of repeated successes". According to Kerzner (Kerzner 2017) "maturity implies that proper foundation of tools, techniques, processes and even culture, exists".

Mullaly (Mullaly 2006) suggests that "the assessment of organizational capabilities is a core dimension of organizational learning and improvement". Assessment of the current capability/maturity level of an organization and its software development and delivery process will provide an indication for whether an agile or traditional or hybrid SDM would be more suitable.

By using maturity models, organizations can carry out an assessment to determine their current maturity level and the list of things they need to work on to improve. According to Fowler, a maturity model is "a tool that helps people assess the current effectiveness of a person or group and supports figuring out what skills and capabilities they need to acquire next in order to improve their performance" (Fowler 2020).

There are several maturity models described in the project management literature that can be used to assess and improve an organization's maturity level. Most of these models are rooted conceptually on the five-level project management maturity model: the Capability Maturity Model (CMM) developed by the Software Engineering Institute (SEI) of Carnegie Mellon between 1986 and 1993. CMM defines five maturity levels: *Initial*, *Managed*, *Defined*, *Quantitatively managed*, *Optimized*. Since then around 30 different models have been developed each addressing a specific business model or industry context.

Though the CMM model "comes from the field of software development, it is also used as a model to aid in business processes generally, and has furthermore been used extensively worldwide in government offices, commerce, and industry"(Wikipedia 2020). However, CMM is considered to be more associated with a document-heavy, plan-driven culture, which is against the nature of agile software development (Fowler 2020).

Organizational and Software-Development Capability Maturity models are two of the many disciplines that have evolved from the original model. It is important to consider both aspects of maturity when assessing an organization's capacity to positively manage all aspects of application's development.

Since the end of 20th century, organizations have been described in terms of their people, processes and technology. These three pillars are known as the "golden triangle". However, due to the rapidly changing environment and different nature of organizations, numerous other dimensions have been proposed to better describe the specifics of the context and the situation of the various organizations.

Maturity Models have, in general, five stages and use a similar nomenclature. This research uses the maturity model that was developed by Stanford's Linear Accelerator Center Laboratory, which evaluates maturity of an organization against 3 dimensions (*People*, *Process*, *Technology*) to determine the maturity level of an organization against the five sequence of stages that define a path from the lowest (Performed) to the highest state to maturity (Organizing) as explained in the Table 4: Five stages of an Organizational Maturity below.

The *People* dimension covers the resources and capacity principles examining both the individual capabilities such as education, training and skills, as well as the organization capabilities such as culture, policy, strategy.

The *Process* dimension covers the methodological aspects, such as the existence and utilization of standards, guidelines, best principles and quality management processes.

The *Technology* dimension analyzes the supporting technology infrastructure, tools, platforms, systems and services that are used in the organization.

Table 4: Five stages of an Organizational Maturity

Stages of Maturity	**Definition**		
	People	**Process**	**Technology**
Performed	Success depends on individual heroics - "Firefighting is a way of life." Relationships between disciplines are uncoordinated, perhaps even adversarial.	Unpredictable process that is poorly controlled and reactive.	Despite security issues, no controls exist.
Managed	Success depends on individuals and management system supports. Commitments are understood and managed. People are trained.	Project process is characterised but is often reactive.	Some controls in development with limited documentation.

Established	Project groups work together, perhaps as an integrated product team. Training is planned and provided according to roles.	Characterised process for the organisation that is proactive.	More controls documented and developed, but over-reliant on individual efforts.
Predictable	A strong sense of teamwork exists within each project.	Process measured and controlled.	Controls monitored, measured for compliance but uneven levels of automation.
Optimizing	A strong sense of teamwork exists across the organization AND everyone is involved in process improvement	Process improvement focus	Controls more comprehensively implemented, automated, subject to continuous improvement.

The Information Management (IM) and Information Technology (IT) domain includes five key disciplines:

- Business / Process
- Security
- Information
- Development
- Operations

Assessing the maturity of an organization for each of these five disciplines can provide an indication of the organization's readiness to successfully implement agile principles and system development methodologies. The higher the maturity level of an organization in the IM/IT domains listed above, the higher the capability of an organization to provide supporting environment conducive to the implementation of a SDM. Agile methodologies encourage the continual improvement of the software delivery process. The principle #12 of the agile manifesto states "At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly" (Fowler and Highsmith 2001).

This research proposes that you use your judgement to assess the maturity of each discipline and tally up the results and average them.

Note 1: to assess each IM/IT domain, select the appropriate maturity level.
Note 2: total up your individual IM/IT domain scores
Note 3: use this grid to identify your In-Range Level

Grand Total Scores	**IN RANGE LEVEL**
0 to 3 points	0 – Performed
4 to 7 points	1 – Managed

8 TO 12 points	2 – Established
13 to 16 points	3 – Predictable
17 to 20 points	4 – Optimizing

		MATURITY LEVELS				
		Performed (0 point)	**Managed** (1 point)	**Established** (2 points)	**Predictable** (3 points)	**Optimizing** (4 points)
IM/IT DOMAINS	Business/Process		1			
	Security		1			
	Information			2		
	Development				3	
	Operations			2		

GRAND TOTAL: 9
IN RANGE LEVEL: 2
CAVEAT: The scores that are presented in the table above represent the evaluation results of the MPIC project. As can be seen from the grid above, with the in range level of 2 (Established), the organizational environment within which the MPIC project is developed, is mature enough use an agile methodology.

5. PROJECT MANAGEMENT METHODOLOGY

The choice of SDM must be accompanied by an appropriate project management approach.

This research uses the systematic review methodology that is based on Kitchenham's methodology (Kitchenham 2004) to provide a background and identify gaps in the field of *agile project methodologies that are used in IC development projects* in order find areas for further investigation.

Kitchenham proposes systematic review guidelines specific for software engineering researchers. Her proposal defines a systematic review as follows: "A systematic review is a means of evaluating and interpreting all available research relevant to a particular research question, topic area, or phenomenon of interest. Systematic reviews aim to present a fair evaluation of a research topic by using a trustworthy, rigorous, and auditable methodology".

"Methodologies are multiple-step models for complex processes. Their purpose is to systematically guide through a full series of steps that must be internalized for growth in a process to occur" (Leise and Beyerlein). As discussed in the Challenges of IC Design section, IC design and development process is very complex and the process complexity is increasing according to Moore's law. So, having a defined and appropriate methodology will allow the IC design and development projects to extract the most efficiency from the project management activities. Greater efficiencies contribute to increased chances of project success. When project managers and project team members have defined appropriate processes, templates, documents and guidelines to refer to, this will assist their planning, execution and monitoring of the project. So overall, having a methodology means a great chance of project success. This project management methodology will be based on best practices from Project Management Body of Knowledge (PMBOK) and with the continuous feedback from team members, will be improved as required.

In this research, as suggested by the MAF assessment of the MPIC project (please see Proposed System Development Methodology (SDM)), an agile project management methodology was selected as the methodology approach and used by the project team members to deliver the MPIC project. This methodology consists of a collection of processes, tools, techniques and templates for managing the project and it was developed by following the steps described in the table below (Smith, Apple et al. 2006).

Step	Description
1. Define the direction	Specify objectives for the process and identify who benefits from it.
2. Identify key issues	Identify key performance factors that affect the quality of the process.
3. Put the process into context	Obtain a systematic overview to determine the scope, focus, and use of the process.
4. Set criteria	Set criteria and determine outcomes that will be used to assess the quality of the process and its results.
5. Inventory information & resources	Collect expertise in the use of the target process, including quality, quantity, timeliness, and the cost of relevant information.
6. Logically order process.	Organize the process into steps; build the methodology iteratively with the team members and document it using process flow chart, user guides and templates. It is very important to include the project team in this step as we need to get their input and their commitment.
7. Execute the methodology	Test the methodology using it as a guide, not as a rule book. By using the Plan-Do-Act-Check cycle, make the necessary adjustments as required and based on the feedback received from the project team members.
8. Assess each step	Collect data and measure performance in “real time” to improve future performance.
9. Facilitate the process.	Use facilitation, assessment, and management skills to help participants learn the process. Note whether it is working as expected and be prepared to make required changes for improvement.
10. Assess performance.	Conduct audits to see if the methodology is being used as expected. Determine necessary changes in the methodology by analyzing the differences between the desired and actual outcomes. This will also reveal opportunities for continuous improvement.

The figure below (Figure 14: Project Management Methodology) shows the process flow chart for the project management methodology that will be used in the MPIC project.

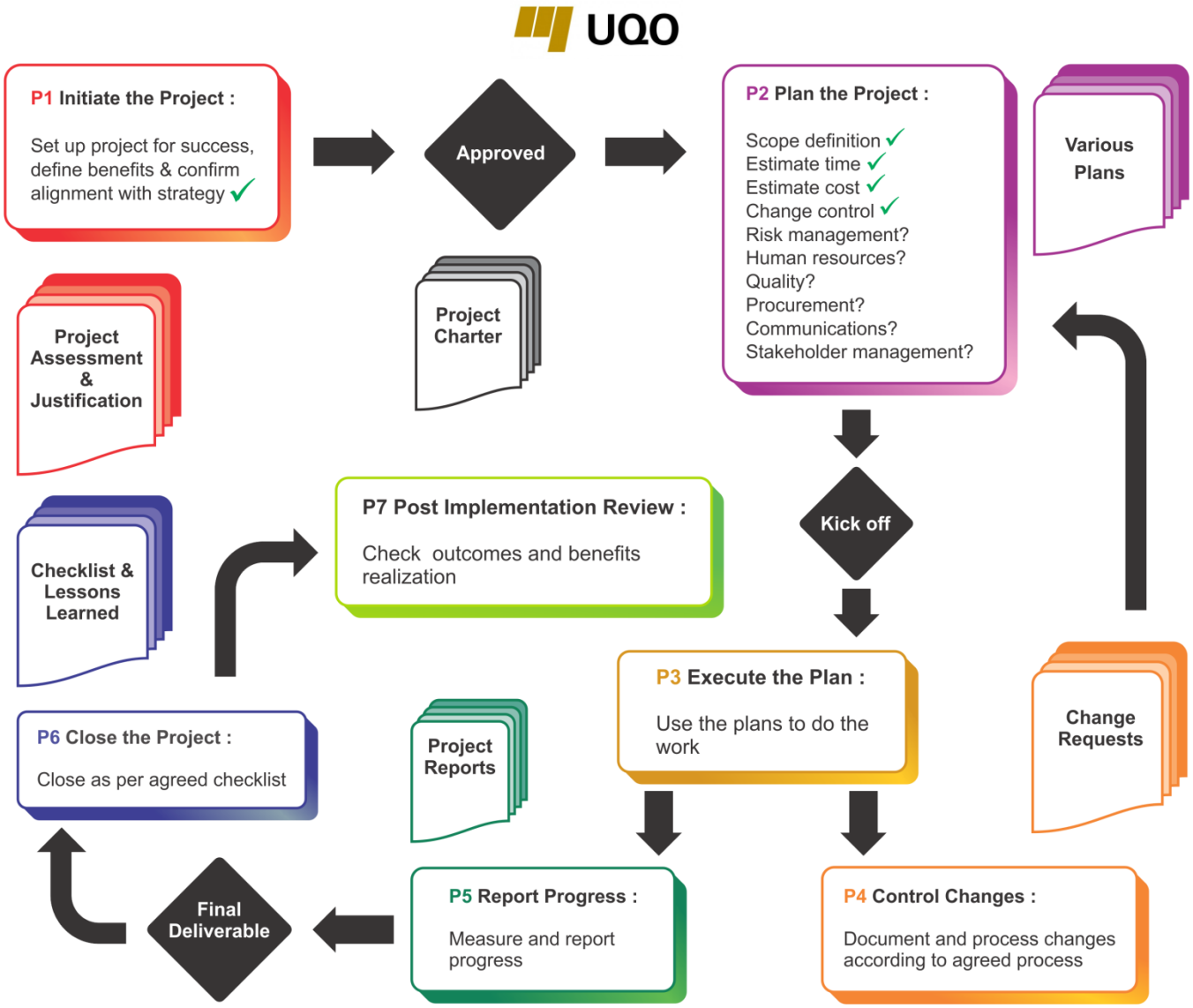


Figure 14: Project Management Methodology

5.1 Recommended Agile Methodology for the MPIC Project

Upon the review of several agile methods that are commonly used (please refer to Appendix B: Overview of Several Agile Methods that are commonly used) and their comparison (please refer to Appendix C: Comparison of commonly cited agile methods in the literature), for the MPIC project, a combination of XP and SCRUM agile approaches were chosen and adopted.

By incremental adoption of agile software development and project management practices as suggested in Appendix D: MPIC Project Activities and Agile Methods that can be used, first, all the software features that are to be implemented were discussed between the domain architects and developers and an overall domain model was created. Next, the feature list was analyzed and prioritized; the dependencies between

features were identified. The team then worked on the feature decomposition where the complex functionalities were divided into smaller size chunks that can be developed and delivered within 3 weeks incrementally and each feature was assigned their owner, the person responsible for the development of the feature. The features were developed on an iterative fashion and the code was integrated continuously with weekly load builds. During each iteration, the team worked through the complete software development cycle (planning, requirements, analysis, design, coding, testing, as shown in Figure 6: Agile development before the working code was demonstrated to stakeholders.

I believe that the MPIC project has benefited from the use of iterative incremental Scrum agile programming practice and processes, as depicted in Figure 15: Suggested Sprint Cycle for MPIC Project (in Scrum) below. In order to implement the Scrum methodology, the project started with sprint planning, by creating a backlog that contains all the user stories (work items). These user stories were broken out into sprints (3-4 week long iterations) and a plan was prepared to get all of these user stories done within a certain timeframe of sprint duration. At the start of each iteration/sprint, the team performed sufficient just-in-time planning by reviewing, prioritizing and agreeing on the user stories that are to be implemented during the sprint, based on the allotted resources in the allotted time.

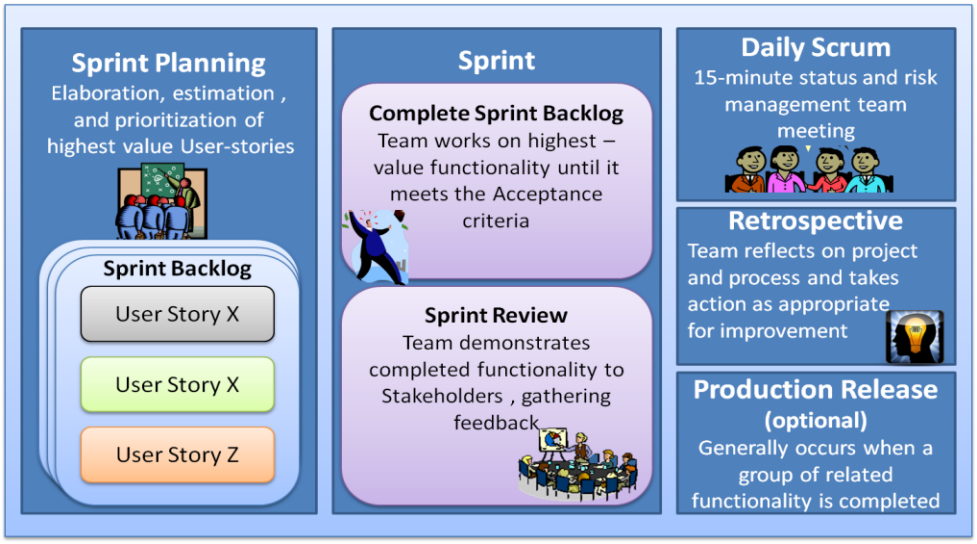


Figure 15: Suggested Sprint Cycle for MPIC Project (in Scrum)

During the sprints, daily Scrum meetings were held to review the progress and productivity and to address the blocking issues. As the user stories were worked on, they move from backlog to working state and once a user story was completed, it moved to closed state. Progress and flow were usually tracked using burndown chart and release burndown chart. Burndown chart showed progress of the number of user stories to be completed within a sprint over the sprint length (workdays) while release burndown chart

showed the progress of the number of user stories for the entire project by the total number of sprints. These charts demonstrated the progress of the team's work through a sprint or release and helped project manager forecast when all user stories would be complete. They also allowed measuring the completion rate of user stories and determining the work capacity and value.

At the end of each iteration/sprint, the project team demonstrated the completed functionality to project stakeholders and gathered the feedback. After each sprint, the team reflected on lessons learned and how to become more effective, in order to tune and adjust their behavior.

The following table (Table 5: Suggested Roles and Responsibilities in the MPIC Project team) lists the roles of each member within the MPIC project. The main roles for project would be *Product Owner*, *Scrum Master* and *Scrum Team*.

Roles	Responsibilities
Product Owner	- Acts as the voice of the end-user to the team and defines and prioritizes features/user stories in the product backlog and accepts them as done at the end of each sprint. - Ensures that the development team has the right understanding of the required features and the functionality of the OPIC tool. - Compiles the project requirements and prepares a prioritized list (Product Backlog), which is composed of user stories for all the planned features and the functionality and administrates the list. - Through discussions with the Scrum Master and the Scrum Team, he determines the goal of each sprint, tasks within each sprint and the duration of sprints - Before each sprint determines the highest prioritized items which are to be transferred to a Sprint Backlog.
Scrum Master	- Ensures that the rules of Scrum are followed and is in charge for removing the possible impediments in the work of the Scrum team. - Facilitates the development, maintains processes, assigns the user stories across Scrum Team members. - Holds the daily Scrum meetings with the Scrum team. - Tracks the progress towards team goals using burndown charts and performs any needed renegotiation or reassignment with the Product Owner. - After every sprint holds an evaluation meeting (Sprint retrospective) with the Scrum team.
Scrum Team (includes the design team)	- The Scrum Team is self-organized and has the authority to decide on the necessary actions to complete the sprint goal. - Develops and tests the user stories they have been assigned. - Sets the status of their assigned user stories as they progress.

Table 5: Suggested Roles and Responsibilities in the MPIC Project team

A typical release structure for the OPIC tool was planned to include six, 3-week long sprints, with the final sprint being four weeks long to allow completion of more extensive release testing, as shown in Table 6 : Suggested Scrum release structure for the MPIC project below.

Sprint 1 (3 weeks)	Feature planning, Design, initial documentation, development start
Sprint 2 (3 weeks)	Development/test
Sprint 3 (3 weeks)	Development/test
Sprint 4 (3 weeks)	Development/test
Sprint 5 (3 weeks)	Development/test
Sprint 6 (4 weeks)	Release testing, final bug fixing, release packaging

Table 6 : Suggested Scrum release structure for the MPIC project

Note that it took some time to refine the Scrum model to the point where team members were comfortable and productive. The Sprint length were determined based on how the team members felt, and 3 to 4 weeks of iteration time seemed to provide more development time with less test/release overhead.

JIRA[6], which is a very useful and flexible tool, was also used for issue tracking and project management. This tool was used to represent each requirement as a user story, which could be assigned across team members and tracked, on a sprint by sprint basis. By using its powerful dashboards, project progress was tracked. Please refer to Appendix E: Proposed Agile Dashboard for the MPIC Project for the dashboard for the MPIC project. This dashboard is composed of 4 gadgets. *Recently Created Chart* displays the created (in Red) and resolved (in Green) user stories on a per day basis; *Created vs. Resolved Chart* displays the trend of created vs. resolved user stories for the project; *Assigned to me* gadget displays all user stories assigned to me; and *Issue Statistics* gadget displays the team members assigned to work on the user stories and their statistics.

[6] JIRA is a proprietary issue tracking product, developed by Atlassian. It provides bug tracking, issue tracking, and project management functions.

6. LESSONS LEARNED

OPIC DRC tool was developed using Scrum for project management and XP engineering practices based on an iterative and incremental development approach since these two agile framework are well aligned and complement each other.

SCRUM framework has been used for defining how work is specified and the process with which features are delivered while making use of XP engineering practices, such as the continuous integration, pair-programming, incremental design practices of XP to build a good quality software.

By applying an agile methodology successfully in the OPIC tool project, the development has been split into sub-tasks and delivered in sprints using an iterative and incremental development approach. SCRUM methodology was an excellent way to make the development cycle more efficient.

Dividing one long marathon into a series of 'sprints' created a customer-centric, agile mentality. We integrated every 3-4 sprints to create a release, which is a “Done”, useable, and potentially releasable product increment. By incorporating continuous integration and testing into the development process, the development team was able to address issues while they are fresh.

Teamwork, collaboration, transparency is crucial– who works on what and how your piece fits in with the rest of others, who needs your work. Through constant team communication and feedback received over the daily scrums, sprint review meetings and retrospective meetings, the product and processes were continuously improved and working functionality was demonstrated quickly.

Agile mindset has to be understood and embraced by everyone in the team. The scrum master needs to set an example for the team and help the team stay on task and aligned with workflows. Their role is to coach and motivate. That’s why choosing the right scrum master is very important. They have to be capable of leading, overseeing and performing follow-through that must be undertaken.

Unlike traditional one-and-done projects, iterating more quickly revealed issues, errors and opportunities for improvement, faster, and generally resulted in smaller teams sharing—and being responsive to—more information continually.

Planning and big-picture view was necessary. We needed to understand what we were working towards. Because each piece should be able to come together in the end. We also needed to define the criteria for done-ness for each user story, for each sprint and for the release. Based on my previous experience, I was

aware that establishing an upfront, common understanding of "done" would save team countless hours of process-thrash, unclear communication, and hidden work.

What does it mean to be done with a user story for a developer, for a tester ? Code complete, code checked-in, integrated, no P1, P2 bugs. What does it mean to be done with a sprint ? For us the sprint success was not determined by the number of user stories completed. Even if all user-stories in the sprint could not be completed, the sprint had to end on the stipulated date and we had to deliver a 5- minute demonstration of the software that demonstrates a working increment. What does it mean to be done with the release? What is the required artifacts, testing & coverage levels, quality or allowed defect levels, results or performance metrics achievement levels. No P1 bugs.

In many Agile environments testing is either pushed to the end of a Sprint or is handled in a separate Sprint. This “mini-waterfall” approach to testing can be the root cause for a number of problems, including

- stress for the testers,
- delays in getting to Potentially Shippable Product Increment,
- missed testing, and
- team interruptions

Agile projects depend on retrospectives being performed so that we can discuss with team members what was learned, how the team is performing, and how the team can improve. If we are not holding proper retrospective meetings, not only will it be more difficult to place where everyone is, but if a team member struggles while another can help them out, you are missing out on an important opportunity for collaborative project success. Make sure to hold retrospective meetings on a regular basis.

This matters from several perspectives: It helps with your estimates. If we don’t have clarity around what should go into completing our work, how could we estimate our work. That’s the very point we have focused on here. Clear understanding of what is expected in completing our work.

It helps with your quality. It provides guidance to the team surrounding what makes each step or deliverable complete. It focused on quality of the steps. And it amplifies consistency – so that every check-in or deliverable meets a consistent level of completeness.

It helps your Product Owner and customer gain confidence as the team delivers. And it’s not just confidence on the individual stories. It’s confidence on the overall plans and teams ability to meet their commitments with consistent quality.

How well the team has met all of the goals and criteria they set forth when they planned their sprint. A large part of these criteria are typically driven by a successful sprint review or demo.

7. CONCLUSION

This research has contributed by proposing three innovative approaches that would improve IC design productivity and success hence would facilitate more efficient, faster semiconductor development and time-to-market.

- The process of migrating from one process technology to another and applying given circuit IP for use with this new target technology requires a lot of repeated manual and interactive tasks. Hence tool support to make this process more efficient is necessary. At the present time, there is not any known tool that automates the layout migration from one technology node to another for the analog and mixed-signal circuits. This research proposed and introduced a new EDA tool for IP layout migration.
- In addition to a new EDA tool, the research also proposed a new, more agile design model for IC projects. Using this proposed IC design model, the schematics and layout design phases could progress in parallel, instead of sequentially one after the other and this would increase the design productivity.
- The topic of "Which system development methodology (SDM) should we use?" is one of the first decisions faced for project implementations. Upon systematically reviewing the academic literature on available SDMs, as existing evaluation frameworks and comparison tools do not satisfy all the needs of project managers, this research introduced a new framework called MAF to help decide on the best suited SDM for a given project. This framework identified 7 elements that contribute to the choice of a suitable SDM for a project.

Project managers need to select the most appropriate SDM for their projects. A selection and implementation of an appropriate SDM is crucial as it maximizes the chance of project success. Deciding on whether to use an agile or plan-driven or hybrid methodology in a project is not a rocket science but it requires honest answers to difficult questions and courage to make the right decision. The assessment of each of these 7 elements is subjective and will be influenced by the project manager and/or the decision makers. How they perceive and respond to complexities is more of an individual and interactive consideration than is represented by the current literature. There will never be a perfect method as all projects are different but for the majority there is a best suited method. Success is about making the right choices.

It would be worth exploring if MAF is a reliable framework that can be used as a precise evaluation and assessment tool. Even though the practical application of MAF demonstrated that it provided a structure for assessing and evaluating a project to determine a suitable SDM, the framework elements might be subject to more analysis. Defining and using quantitative metrics for as many of the seven factors (as opposed to qualitative metrics) would make MAF a more comprehensive evaluation framework.

Further research might be of value to improve the MAF. This research would benefit from an empirical study to refine and update the proposed MAF and apply it to the assessment of various projects of different size and complexity. With empirical feedback that can be acquired from real project situations, the evaluation criterion and the metrics used can be improved in order to provide more precise results.

The choice of SDM must be accompanied by an appropriate project management approach. This research has been about using the agile development methodology with the agile project management methodology. Upon using the MPIC project as a case study and assessing the project by using the new MAF framework, and upon the literature review that demonstrated the expected benefits of using agile project management methodology in software design, in hardware design and in software hardware co-development projects, this research suggested adopting a hybrid agile project management methodology (combination of XP and SCRUM) in IC development projects.

The analysis could also be extended to cover a study of using the new EDA tool called OPIC in different IC development projects.

8. ANNEX : SYNTHESE

Cette recherche a contribué en proposant trois approches innovantes qui visent à améliorer la productivité et le succès de la conception de circuits intégrés, facilitant ainsi le développement d'outils EDA (Electronic Design Automation) de conception plus efficaces réduisant le temps de mise sur le marché.

- Le processus de migration des IP (Intellectual Property) de Circuits Intégrés (CI) d'un nœud technologique à un autre nécessite de nombreuses tâches manuelles et interactives répétées. Par conséquent, un support d'outils pour rendre ce processus plus efficace est nécessaire. A l'heure actuelle, il n'existe aucun outil connu qui automatise la migration ou portage du ''*Layout*'' d'un nœud technologique à un autre pour les circuits analogiques et mixtes. Cette recherche a proposé et introduit un nouvel outil EDA pour la migration des IP en générant son *Layout* dans la technologie cible.

- En plus d'un nouvel outil EDA, la recherche a également proposé un nouveau modèle de conception plus agile pour les projets de nouveaux CI. En utilisant ce modèle de portage des CI proposé, les phases de conception schématique et de *Layout* pourraient progresser en parallèle, au lieu de séquentiellement l'une après l'autre, ce qui augmenterait la productivité de la conception.

- Le thème « Quelle méthodologie de développement de système (SDM) devrions-nous utiliser? » est l'une des premières décisions à prendre pour la mise en œuvre des projets. Après avoir systématiquement examiné la littérature académique sur les SDM disponibles, étant donné que les cadres d'évaluation et les outils de comparaison existants ne répondent pas à tous les besoins des chefs de projet, cette recherche a introduit un nouveau cadre appelé MAF pour aider à décider de la SDM la mieux adaptée pour un projet donné. Ce cadre a identifié 7 éléments qui contribuent au choix d'une SDM appropriée pour un projet.

Les chefs de projet doivent sélectionner la SDM la plus appropriée pour leurs projets. La sélection et la mise en œuvre d'une SDM appropriée sont cruciales car elles maximisent les chances de succès du projet. Décider d'utiliser une méthodologie agile, planifiée ou hybride dans un projet n'est pas évident, mais cela nécessite des réponses honnêtes à des questions difficiles et du courage pour prendre la bonne décision. L'évaluation de chacun de ces 7 éléments est subjective et sera influencée par le chef de projet et/ou les décideurs. La façon dont ils perçoivent et réagissent aux complexités est davantage une considération individuelle et interactive que ne le représente la littérature actuelle. Il n'y aura jamais de méthode parfaite car tous les projets sont différents, mais pour la majorité, il existe une applroche la mieux adaptée. Le succès consiste à faire les bons choix.

Il serait intéressant de savoir si le MAF est un cadre fiable pouvant être utilisé comme un outil précis d'évaluation et de validation. Même si l'application pratique du MAF a démontré qu'il fournissait une structure pour évaluer et valider un projet afin de déterminer une SDM appropriée, les éléments du cadre pourraient faire l'objet d'une analyse plus approfondie. La définition et l'utilisation de paramètres quantitatifs pour chacun des sept facteurs (par opposition aux paramètres qualitatifs) feraient du MAF un cadre d'évaluation plus complet.

Des recherches supplémentaires pourraient être utiles pour améliorer le MAF. Cette recherche bénéficierait d'une étude empirique pour affiner et mettre à jour le MAF proposé et l'appliquer à l'évaluation de divers projets de taille et de complexité différentes. Grâce à un retour d'expérience empirique à partir de situations réelles de projet, le critère d'évaluation et les métriques utilisées peuvent être améliorés afin de fournir des résultats plus précis.

Le choix de SDM doit être accompagné d'une approche de gestion de projet appropriée. Cette recherche a porté sur l'utilisation de la méthodologie de développement agile avec la méthodologie de gestion de projet agile. En utilisant le projet OPIC comme étude de cas et en évaluant le projet en utilisant le nouveau cadre MAF, et en se basant sur la revue de la littérature qui a démontré les avantages attendus de l'utilisation d'une méthodologie de gestion de projet agile dans la conception de logiciels, de matériel et dans le co-développement de projets impliquant matériel et logiciel, cette recherche a abouti à la recommandation d'adopter une méthodologie de gestion de projet agile hybride (combinaison de XP et SCRUM) dans les projets de développement et portage de circuits intégrés.

L'analyse pourrait également être étendue pour couvrir une étude d'utilisation du nouvel outil EDA appelé OPIC dans différents projets de développement de circuits intégrés.

9. Appendix A: The MPIC Project Team And Project Milestones

9.1 Project Team

MPIC Project team consists of the following individuals. The project director Prof. Lakhssassi and project co-director Prof. El Guemhioui lead this team at strategic, tactical and operational levels and oversee the management and technical aspects of the project to be delivered by the project team to ensure that the project gets delivered on time, and on target.

- Project Director: Prof. Ahmed Lakhssassi
- Project co-Director: Prof. Karim El Guemhioui
- Development of layout Porting by schematic processing: Youcef Fouzar
- Development of DRC (Design Rule Check) Cleanup engine: Karim Baratli
- Development of layout porting technology test cases (including the development of circuit topology and layout based on different technologies): Bachir Lakhssassi
- MPIC Project Management Methodology: Ebru Dalbudak

The figure below (Figure 16: MPIC Project Governance) captures the governance model of the project.

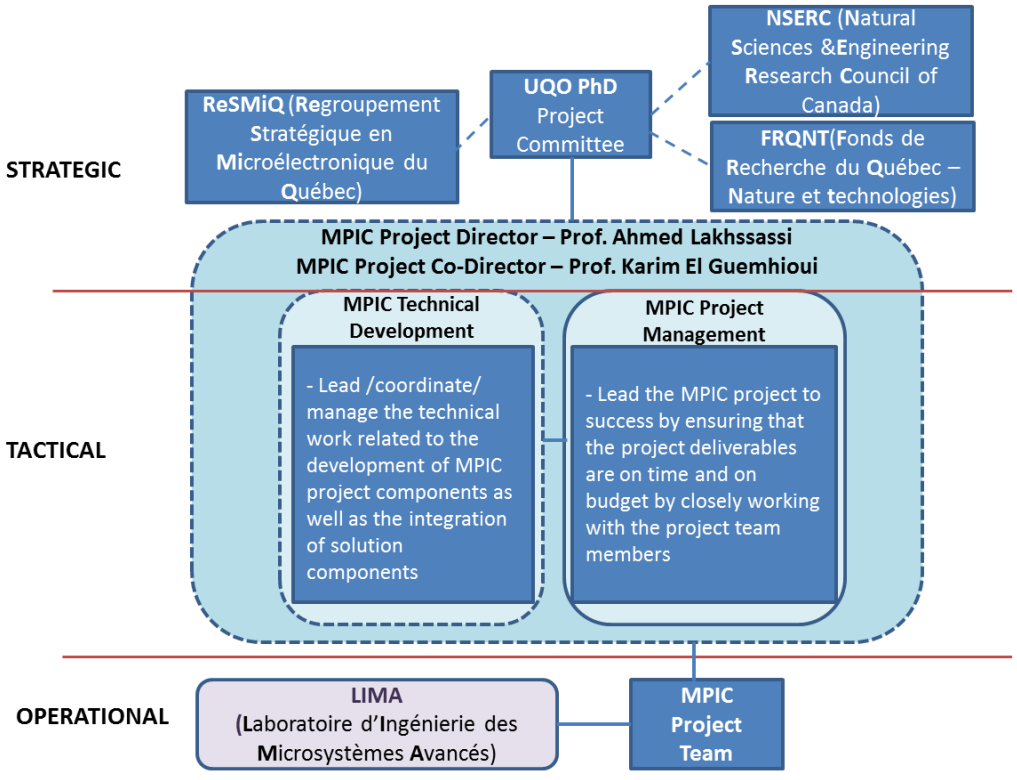


Figure 16: MPIC Project Governance

9.2 Project milestones

As indicated below (Figure 17: MPIC Project milestones), the project started in 2016. In the first year, the goal is to have a fully functional methodology for porting a schematic design. In fact, the methodology will include a whole cycle of fully automated porting including design and verification. In the second year, the layout methodology, including physical checks and verification as well as the testing and verification of the fully automated IP porting methodology will be completed.

Task Number	Milstones	Start	End	Duration in days
1	**Schematic Porting Software**	**12-Sep-2016**	**6-Sep-2017**	**360**
1.1	Develop the gm/id algorithm	12-Sep-2016	10-Nov-2016	60
1.2	Build test benches for gm/id conversion	11-Nov-2016	10-Dec-2016	30
1.3	Build the device mapping software	11-Dec-2016	8-Feb-2017	60
1.4	Build database for target technology	9-Feb-2017	10-Mar-2017	30
1.5	Conversion algorithm for schematic to target technology	11-Mar-2017	8-Jul-2017	120
1.6	Verify target design against the specification	9-Jul-2017	6-Sep-2017	60
2	**Layout Porting Software**	**7-Sep-2017**	**3-Jul-2018**	**300**
2.1	Create software to convert a layout into database	7-Sep-2017	5-Dec-2017	90
2.2	Create software to build generic layout from database	6-Dec-2017	3-Feb-2018	60
2.3	Convert generic layout into target technology	4-Feb-2018	4-Apr-2018	60
2.4	Create software to automate the process of design rule checks and verification	5-Apr-2018	3-Jul-2018	90
3	**Testing the porting methodology**	**4-Jul-2018**	**28-Feb-2019**	**240**
3.1	Build a design in 65nm including layout	4-Jul-2018	31-Oct-2018	120
3.2	Port the schematic from TSMC 65nm to STM FDSOI 28nm	1-Nov-2018	30-Nov-2018	30
3.3	Apply layout porting methodlogy to the design	1-Dec-2018	29-Jan-2019	60
3.4	Create project report	30-Jan-2019	28-Feb-2019	30

Figure 17: MPIC Project milestones

10. APPENDIX B: OVERVIEW OF SEVERAL AGILE METHODS THAT ARE COMMONLY USED

XP (Extreme Programming	XP spirit = 'do the simplest thing that could possibly work'. XP is clearly meant as a tool for software development teams. It advocates the use of simple design and programming practices, and simple methods of planning, tracking and reporting. It is based on a series of concepts that include: Short iterations (one or two weeks—iterations) with small releases and rapid feedback, customer participation, communication and coordination, continuous integration and testing, designing test before writing code, stand-up meetings, refactoring, collective ownership of the code, limited documentation and pair programming. The XP Values are Communication, Simplicity, Feedback and Courage. XP aims at enabling successful software development despite vague or constantly changing requirements in small- to medium-sized teams.
Scrum	Scrum is the most widely accepted agile methodology. It is heavily focused on the iteration management level of agile. It breaks up the development into 2 to 4 week sprints and at the end of each sprint a potentially shippable product increment (working and tested software) is created. Scrum defines a process framework which contains sets of practices and predefined roles such as Scrum Master, Product owner, and Scrum Team. Scrum concentrates on team member interaction and communication in order to facilitate project flexibility in constantly changing environments. Emphasis is not on specific software programming techniques, and it can therefore be easily implemented into other types of projects.
FDD (Feature Driven Development)	The FDD approach is based on the iterative and incremental development but it does not cover the entire software development process, but rather focuses on the design and building phases. FDD consists of five chronological processes: Develop an Overall Model, Build a Features List, Plan by Feature, Design by Feature, and Build by Feature. The last two processes run in an iterative cycle, and therefore are changes to product requirements and business needs possible even late in the overall process. FDD consists of a set of 'best practices' and encourages that all practices available should be used to get the most benefit of the method as no single practice dominates the whole process. In opposition to XP, FDD does not provide concrete guidance in respect to specific development methods. It is mostly a management-supporting tool that suggests a specific framing of the process as well as iterative development in a certain way. FDD proposes fast iterative cycles between one and three weeks with focus on only one or few features at a time. As the name also suggests, FDD is based on the precondition that the work-product can be split into more or less independent parts, which is most often possible in software projects.

AUP (Agile Unified Process)	AUP is a simplified version of the Rational Unified Process (RUP). It is based on the same iterative methodology as RUP but it doesn't have the heavy emphasis on tools and artifacts. It describes a simple, easy-to-understand approach to developing software using agile techniques. AUP is based on the following principles: Team empowerment, simplicity, agility, focus on high-value activities, tool independence, can be tailored to meet your needs.
DSDM (Dynamic Systems Development Method)	DSDM is a framework similar to Scrum, however, it is much more structured than Scrum. The fundamental idea behind DSDM is to manage the output and results, rather than inputs and activities; in other words, instead of fixing the amount of functionality in a product, and then adjusting time and resources to reach that functionality, it is preferred to fix time and resources, and then adjust the amount of functionality accordingly. At the heart of DSDM, there is facilitated workshops, prioritization, timeboxing and prototyping. DSDM relies on interactivity between the project team, future end users and higher management and it is based on the following 9 principles: • Active user involvement is imperative. • The team must be empowered to make decisions. • The focus is on frequent delivery of products. • Fitness for business purpose is the essential criterion for acceptance of deliverables. • Iterative and incremental delivery is necessary to converge on accurate business solution. • All changes during development are reversible. • Requirements are baselined at a high level • Testing is integrated throughout the lifecycle. • Collaboration and co-operation between stakeholders are essential

11. APPENDIX C: COMPARISON OF COMMONLY CITED AGILE METHODS IN THE LITERATURE

Criteria	XP	Scrum	FDD	AUP	DSDM
Project Size	Small, medium	Small, medium and scalable for large	Small, medium and large (business projects/applications)	Large and complex projects	Small and large projects (business applications)
Team Size	< 10	< 10 and multiple teams	No limit- scalable from small to large teams	Not mentioned	Min. 2, max 6 (multiple teams)
Requirements Management	User stories	User stories	Hierarchical product features & use cases	Agile Model-driven development	Prioritized requirements list
Focus on Customer value	Yes	Yes	Limited	Limited	Yes
Respect for people (Team empowerment)	Yes	Yes	Not defined	Yes	Yes
Continuous improvement emphasis	Yes	Yes	Optional	Optional	Yes
Development style	Iterative, rapid	Iterative, rapid	Iterative design and construction	Iterative and rapid development - distributed development	Iterative, rapid development and cooperative
Code style	Clean & simple	Not specified	Not specified	Not mentioned	Not mentioned
Architectural Design Approach	Not specified	Not specified	UML	UML	UML
Test-Driven Development	Yes	Optional	Optional	Optional	Optional
Code refactoring	Yes	Optional	Optional	Optional	Optional
Pair Programming	Yes	Optional	Optional	Optional	Optional
Technology Environment	Quick feedback	Not specified	Not specified	Not mentioned	Not mentioned
Planning & Estimation Approach	Rolling wave with very limited upfront planning	Product backlog, release plans	More emphasis on upfront planning	More emphasis on upfront planning	High-level scope & requirements baselined upfront
Risk Management	Not specified	Not specified	Not specified	Not specified	Upfront feasibility & business analysis ongoing Risk Management
Physical Environment	co-located teams and distributed teams (limited interaction)	Not specified	Not specified but extensible to distributed teams (multiple sites, time zones)	co-located and distributed teams	Not mentioned but extensible to distributed teams (multiple sites, time zones)
Business culture	Collaborative and cooperative	Not specified	Not specified	Not specified	Collaborative and cooperative

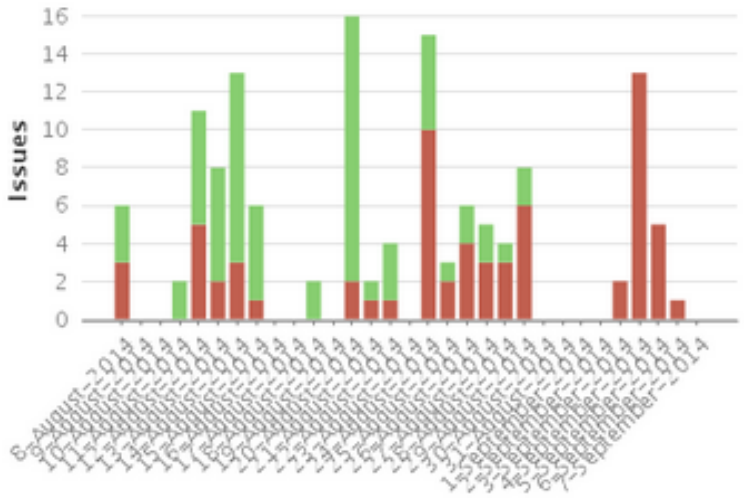
12. APPENDIX D: MPIC PROJECT ACTIVITIES AND AGILE METHODS THAT CAN BE USED

PHASE	PROJECT ACTIVITIES AND AGILE METHODS USED
MPIC Project: Product Definition - Requirements Elicitation - Requirements Analysis - Requirements Validation	**XP** - A cross-functional team containing all relevant stakeholders are put together and a close collaboration and open communication between them are established - Developers and customer collaboratively work on the requirements process - User requirements are captured in User Stories. - User requirements are not baselined and the changes to requirements are accepted after the project started
MPIC Project: EDA Tool Software Development & new IC Development model - Design - Implementation - Testing	**XP** - Constant collaboration and active stakeholder participation and feedback mechanisms are used. - Unit Tests are written before development (Test Driven Development) - Refactoring, continuous testing, frequent Integration - Short 3-week Iterations - Test all code before putting to production. **SCRUM** - Weekly meetings are held to discuss the progress of feature and meeting minutes are captured. - A list of all features are captured and an overall domain model is created - The design is decomposed into features that can be delivered in 3-week iterations. - Each feature is built by the feature owner; continuous testing, frequent Integration - Collaboration and active stakeholder participation

Proposed Dashboard for the OPIC project

+ Add Gadget Edit Layout Tools

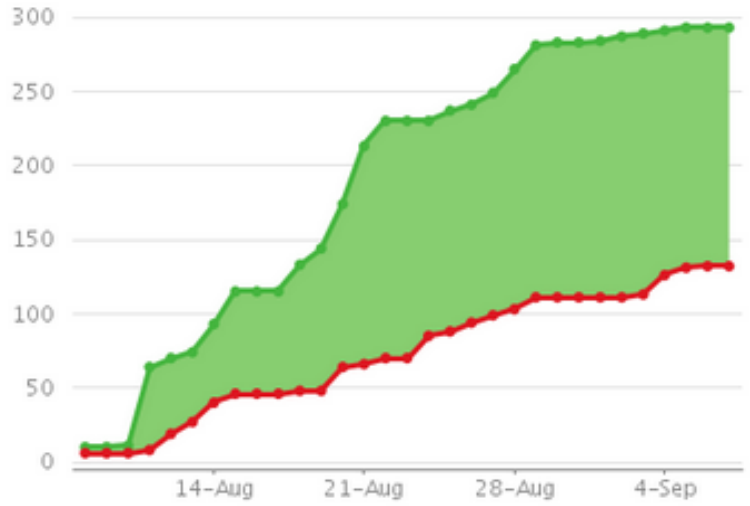
Recently Created Chart: 5582-Elections Canada MPE-EPAS



Total Issues: 132

Period: last 30 days (grouped Daily)

Created vs. Resolved Chart: 5582-Elections Canada MPE-EPAS



Issues: 132 created and 293 resolved

Period: last 30 days (grouped Daily)

Assigned to Me

T	Key	P	Summary
	ECM-357	↑	ECM-128 / Response from EC/BC regarding fields that were sent over DUE 8/14
	ECM-439	↑	PEU to Provide list of Sub-Sub-Topics
	ECM-445	↑	ECM-431 / EC to provide ECSN SME Users

1–3 of 3

Issue Statistics

STATISTICS: 5582-ELECTIONS CANADA MPE-EPAS (ASSIGNEE)

Andrew Newell	2	1%
Brad Bannwart	17	10%
Cynthia Ceil	12	7%
David Alexander	19	12%
David Buffington	6	4%
Ebru Dalbudak	3	2%
Gary Lutz	3	2%
John Chmaj	1	1%

14. APPENDIX F: PAPERS PUBLISHED TO DATE

No	Title and Summary	Published at	Abstract
1	***"On the Use of Agile Methods for IC Design"*** • Explores the use of the iterative and incremental approach of agile methods in IC development projects and proposes a development framework	*28th International Conference on Computer Applications in Industry and Engineering (CAINE 2015)*, San Diego, California, October 12-14, 2015	With the increased level of complexity of Integrated Circuit (IC) designs and market pressures to produce designs quickly, IC companies need to introduce efficiencies into their development process. This paper explores the use of the iterative and incremental approach of agile methods in IC development projects. A development framework is proposed.
2	***"Agile methodology for porting analog and mixed-signal circuits between different technology nodes"*** • Introduces an innovative Electronic Design Automation (EDA) tool that will allow the replication of an existing layout in different technology nodes by automatically porting analog and mixed-signal circuits • Suggests the use of iterative and incremental approach of agile methods by presenting a case study on an EDA tool development project	IEEE International IOT, Electronics and Mechatronics (*IEMTRONICS 2020)* Conference, Vancouver, Canada, September 9-12, 2020	With the growing need for increased capacity, speed and capabilities, the complexity of Integrated Circuit (IC) designs continue to increase. With the increased level of complexity and market pressures to deliver better results faster and cheaper, IC companies are forced to innovate and introduce efficiencies into their development process. To improve IC design productivity and time-to-market, this paper introduces an innovative Electronic Design Automation (EDA) tool that will allow the replication of an existing layout in different technology nodes by automatically porting analog and mixed-signal circuits. Furthermore, to better manage the complexity of IC development projects, the paper suggests the use of iterative and incremental approach of agile methods by presenting a case study on an EDA tool development project.
3	***"Application of the "Methodology Assessment Framework (MAF) on an EDA Tool Development Project"*** • Presents a case study of the application of the Methodology Assessment Framework (MAF) as an evaluation tool to choose the best suited Software Development Methodology (SDM) for the EDA tool development project	*American Journal of Science & Engineering (AJSE), Vol-1, Issue-4, December 2020.*	With the increased level of complexity and market pressures to deliver better results faster and cheaper, Integrated Circuit (IC) companies are looking to innovate and introduce efficiencies into their development process. To improve IC design productivity and time-to-market, this paper introduces an innovative Electronic Design Automation (EDA) tool in order to

			replicate an existing layout in different technology nodes by automatically porting analog and mixed signal circuits. Furthermore, the paper presents a case study of the application of the Methodology Assessment Framework (MAF) as an evaluation tool to choose the best suited Software Development Methodology (SDM) for the EDA tool development project.
4	***"Methodology Assessment Framework (MAF)"*** • Introduces a new tool that is based on seven decision factors to help project managers and decision makers to choose the best suited Systems Development Methodology(SDM) for their project, whether it be an agile methodology, plan-driven methodology or a hybrid of the two	*Journal Of Theoretical And Applied Information Technology', Vol 99, June 2021*	This article introduces the Methodology Assessment Framework (MAF) as an evaluation tool to help project managers and decision makers choose the best suited SDM for their project whether it be an agile, plan-driven methodology or a hybrid of the two. This tool is based on seven decision factors, which are outcomes, scope, CYNEFIN (complexity), constituents, agile principles, team knowledge & experience, and organization capability & maturity. The paper explains each of the seven factors that MAF uses along with their assessment metrics to appraise a given project and based on the evaluation results, suggests whether the project should be run using an agile or plan-driven methodology.

15.REFERENCES

Abrahamsson, P., J. Warsta, et al. (2003). New directions on agile methods: a comparative analysis. 25th International Conference on Software Engineering, 2003. Proceedings., Ieee.

Alexander, M. (2018) How to pick the best project management methodology for success.

Allen, W. (2019) More about outcomes – why they are important … and elusive!

Alqudah, M. K. and R. Razali (2017). "Key factors for selecting an Agile method: A systematic literature review." International Journal on Advanced Science, Engineering and Information Technology **7**(2): 526-537.

Ambler, S. (2014). Just Barely Good Enough Models and Documents: An Agile Best Practice.

Ambler, S. (2018). "Active Stakeholder Participation." from http://agilemodeling.com/essays/activeStakeholderParticipation.htm.

Ambysoft (2013) 2013 IT Project Success Rates Survey Results.

Apke, L. (2020, 16 August, 2016). "Agile Principles: Why you need self-organizing teams." Retrieved March 26, 2020, 2020, from http://www.agile-doctor.com/2016/08/16/why-you-need-self-organizing-teams/.

Batterywala, S. H., S. Bhattacharya, et al. (2008). Cell swapping based migration methodology for analog and custom layouts. Quality Electronic Design, 2008. ISQED 2008. 9th International Symposium on, IEEE.

Bertacco, V. (2003). Achieving scalable hardware verification with symbolic simulation, Stanford University.

Birnbaum, M. D. (2004). "Essential Electronic Design Automation (EDA)." Prentice Hall PTR/Pearson Education.

Boehm, B. and R. Turner (2004). Balancing Agility and Discipline: A Guide for the Perplexed. Boston, MA, Addison-Wesley.

Bryant, R. E., K.-T. Cheng, et al. (2001). "Limitations and challenges of computer-aided design technology for CMOS VLSI." Proceedings of the IEEE **89**(3): 341-365.

Cadence (2008). "Implementing the fastest path from concept to consumer for advanced-node ICS." http://www.cadence.com/rl/resources/white_papers/advancednode_wp.pdf **Cadence.com**.

Choi, Y., I. Chun, et al. (2004). A technology migration method using rectangle-based layout conversion. Advanced System Integrated Circuits 2004. Proceedings of 2004 IEEE Asia-Pacific Conference on, IEEE.

CMS (2005). Selecting a development approach.

Cockburn, A. and J. Highsmith (2001). "Agile software development, the people factor." Computer **34**(11): 131-133.

Darnall, R. and J. Preston (2010). Project management from simple to complex.

Datta, S. (2006). Agility measurement index: a metric for the crossroads of software development methodologies. Proceedings of the 44th annual Southeast regional conference.

De Lucia, A. and A. Qusef (2010). "Requirements engineering in agile software development." Journal of emerging technologies in web intelligence **2**(3): 212-220.

Dingsøyr, T., S. Nerur, et al. (2012). A decade of agile methodologies: Towards explaining agile software development, Elsevier.

Dornelas, H., A. Schmidt, et al. "New Technology Migration Methodology for Analog IC Design."

Dybå, T. and T. Dingsøyr (2008). "Empirical studies of agile software development: A systematic review." Information and software technology **50**(9-10): 833-859.

Eberlein, A. and J. Leite (2002). Agile requirements definition: A view from requirements engineering. Proceedings of the International Workshop on Time-Constrained Requirements Engineering (TCRE'02).

EPMC Introduction to Agile.

Eriksson, U. (2016). The Comprehensive Guide to Making Agile and Waterfall methodologies get along, ReQtest.

Fowler, M. (2020, August 26, 2014). "Maturity Model." Retrieved March 29, 2020, from https://martinfowler.com/bliki/MaturityModel.html.

Fowler, M. and J. Highsmith (2001). "The agile manifesto." Software Development **9**(8): 28-35.

Fowler, M. and J. Highsmith, A. (2001) Agile Manifesto.

Francken, K. and G. Gielen (1999). Methodology for analog technology porting including performance tuning. Circuits and Systems, 1999. ISCAS'99. Proceedings of the 1999 IEEE International Symposium on, IEEE.

Fry, C. and S. Greene (2007). ""Large Scale Agile Transformation in an On-Demand World"." Proceedings of Agile 2007 **Washington, DC: IEEE Computer Society**: 136-142.

Fu, D.-S., Y.-Z. Chaung, et al. (2009). Topology-driven cell layout migration with collinear constraints.

Computer Design, 2009. ICCD 2009. IEEE International Conference on, IEEE.

Geraldi, J., H. Maylor, et al. (2011). "Now, let's make it really complex (complicated)." International Journal of Operations & Production Management.

Geras, A., M. Smith, et al. (2006). Configuring hybrid agile-traditional software processes. International Conference on Extreme Programming and Agile Processes in Software Engineering, Springer.

Griffiths, M. (2013). "Agile suitability filters." Retrieved January.

Griffiths, M. (2018). PMI-ACP Exam Prep, RMC Pubns Inc. .

Highsmith, J. (2004). "Agile Project Management:Creating Innovative Products."

Highsmith, J., A. (2002) What is Agile Software Development? CROSSTALK The Journal of Defense Software Engineering

Highsmith, J., A. (2004). Agile Project Management: Creating Innovative Products, Pearson Education Inc.

jain, m. (1999) Implementing a Design Reuse Strategy Using Hard IP.

Jansen, D. (2003). "The electronic design automation handbook." Springer Science and Business Media.

Johnson, N. (2013) Agile hardware development– nonsense or necessity. EE Times

Jones, C. (2013). Evaluating agile and scrum with other software methodologies, InfoQ.

Kar, P. K. and S. K. Roy (1999). TECHMIG: A layout tool for technology migration. VLSI Design, 1999. Proceedings. Twelfth International Conference On, IEEE.

Kerzner, H. (2017). Project management: a systems approach to planning, scheduling, and controlling, John Wiley & Sons.

Keystone (2018). "Impact, Planning, Assessment and Learning(IPAL) Guide 3 Learning with Constituents." from https://keystoneaccountability.org/wp-content/uploads/files/3%20Learning%20with%20constituents_0.pdf.

Kitchenham, B. (2004). "Procedures for performing systematic reviews." Keele, UK, Keele University **33**(2004): 1-26.

Kostigoff, S. (2003) Methodology:: Development Models.

Lakhssassi, A., Y. Fouzar, et al. (2016). Design Methodology for Porting Analog and Mixed-Signal Circuits between different technology nodes.

Leise, C. and S. W. Beyerlein "2.3. 7 Learning Processes through the Use of Methodologies."

Macintosh, M. (2014). "Analog Hard IP made portable." Chip Design Magazine.

Mansor, Z., S. Yahya, et al. (2011). "Success Determinants in Agile Software Development Methodology." Procedia Engineering.

McConnell, S. (1996). Rapid Development: Taming Wild Software Schedules, Microsoft Press.

Mersino, A. (2018) Project Success Rates: Agile versus Waterfall.

Micic, L. (2017). Agile methodology selection criteria: IT start-up case study. IOP Conference Series: Materials Science and Engineering, IOP Publishing.

Mikkelsen, M. (2018). "In search of project leadership principles for navigating the complexity of IT projects."

Mishra, D. and A. Mishra (2011). "Complex software project development: Agile methods adoption." Journal of Software Maintenance and Evolution **23**(8): 549-564.

Moretti, G. (2014). "An EDA view of semiconductor manufacturing " Systems Design Engineering.

Morien, R. (2014). "An Agile Software Project Management Manifesto - A Reference Disciplines Framework for Agile Development." Int. J. Advance Soft Compu. Appl **6**(1).

Mukundan, S. (2013) Physical Design Flow: Physical Verification.

Mullaly, M. (2006). "Longitudinal analysis of project management maturity." Project Management Journal **37**(3): 62-73.

Mullaly, M. and J. L. Thomas (2009). "Exploring the dynamics of value and fit: Insights from project management." Project Management Journal **40**(1): 124-135.

MunEDA (2014) Automated Design Migration & IP Porting Flow for Custom Circuits.

Paetsch, F., A. Eberlein, et al. (2003). Requirements engineering and agile software development. Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on, IEEE.

Parsons, D., H. Ryu, et al. (2007). "The Impact of Methods and Techniques on Outcomes from Agile Software Development Projects." IFIP International Federation for Information Processing **235**: 235-249

Payne, D. (2016) Three Steps for Custom IC Design Migration and Optimization.

PMI (2017). Agile Practice Guide, PMI.

PMI (2017). A guide to the Project Management Body of Knowledge - Sixth Edition, Project Management Institute.

Qian, L., Z. Bi, et al. (2015). "Automated technology migration methodology for mixed-signal circuit based on

multistart optimization framework." IEEE Transactions on Very Large Scale Integration (VLSI) Systems **23**(11): 2595-2605.

Qumer, A. and B. Henderson-Sellers (2006). Comparative evaluation of XP and Scrum using the 4D Analytical Tool (4-DAT). Proceedings of the European and Mediterranean conference on information systems.

Rakitin, S. R. (2001). "Maifesto Elicits Cynicism." Computer **December**.

Rivera, M. (2020) Agile vs. Scrum for beginners: Everything you need to know.

Royce, W. W. (1987). Managing the development of large software systems: concepts and techniques. Proceedings of the 9th international conference on Software Engineering, IEEE Computer Society Press.

Sangiovanni-Vincentelli, A. (2003). "The tides of EDA." IEEE Design & Test of Computers **20**(6): 59-75.

Saunders, R. G. (1992). "Project management: a systems perspective." International Journal of Project Management **10**(3): 153-159.

Scheible, J. and J. Lienig (2015). Automation of analog IC layout: challenges and solutions. Proceedings of the 2015 Symposium on International Symposium on Physical Design, ACM.

Schwaber K., S. J. (2017). "The Scrum Guide - The Definitive Guide to Scrum: the rules of the game."

Shaphir, E., R. Y. Pinter, et al. (2015). "Efficient cell-based migration of VLSI layout." Optimization and Engineering **16**(1): 203-223.

Silveira, F., D. Flandre, et al. (1996). "A g/sub m//I/sub D/based methodology for the design of CMOS analog circuits and its application to the synthesis of a silicon-on-insulator micropower OTA." IEEE journal of solid-state circuits **31**(9): 1314-1319.

Smith, P., D. Apple, et al. (2006). "Methodology for creating methodologies." Faculty Guidebook. Lisle, Illinois: Pacific Crest: 371-374.

Sobe, U., A. Graupner, et al. (2009). Analog ip porting by topology conversion and optimization. IP-ESC09 Conference.

Sterman, J. D. (2002). "All models are wrong: Reflections on becoming a systems scientist." System Dynamics Review **18**(4): 501-531.

Strode, D. (2006). Agile methods: a comparative analysis. Proceedings of the 19th annual conference of the national advisory committee on computing qualifications, NACCQ.

Systems, N. M. (2000). "Measuring IC and ASIC Design Productivity." (May).

Taromirad, M. and R. Ramsin (2008). An appraisal of existing evaluation frameworks for agile methodologies. 15th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ecbs 2008), IEEE.

Taromirad, M. and R. Ramsin (2008). Cefam: Comprehensive evaluation framework for agile methodologies. Software Engineering Workshop, 2008. SEW'08. 32nd Annual IEEE, IEEE.

van Casteren, W. (2017). "The Waterfall Model and Agile Methodologies: A comparison by project characteristics."

VLSIExpert (2012) VLSI Concepts.

Wang, L., Y. Chang, et al. (2009). "Electronic Design Automation: Synthesis, verification, and test."

Wikipedia (2020). Capability Maturity Model. Wikipedia.

Wolf, W. (2002). Modern VLSI design: system-on-chip design, Pearson Education.

Xia, W. and G. Lee (2005). "Complexity of information systems development projects: Conceptualization and measurement development." Journal of Management Information Systems **22**(1): 45-83.

Zelnik, C. (2002) Layout compaction accelerates SoC design through hard IP reuse.

Zhu, J., F. Fang, et al. (2005). "Calligrapher: a new layout-migration engine for hard intellectual property libraries." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **24**(9): 1347-1361.