International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:1, No:10, 2007

The Application of Six Sigma to Integration of Computer Based Systems

Zenon Chaczko, Essam Rahali, and Rizwan Tariq

***Abstract*—**This paper introduces a process for the module level integration of computer based systems. It is based on the Six Sigma Process Improvement Model, where the goal of the process is to improve the overall quality of the system under development. We also present a conceptual framework that shows how this process can be implemented as an integration solution. Finally, we provide a partial implementation of key components in the conceptual framework.

***Keywords*—**Software Quality, Six Sigma, System Integration, 3SI Process, 3SI Conceptual Framework.

I. Introduction

IN computer systems development, there is a recurring problem of projects being late, over budget and failing to satisfy customer requirements. The amalgamation of such problems translates to poor quality. In large software development projects "Of those that are completed, about two thirds experience schedule delays and cost overruns that may approach 100 percent. Roughly the same number are plagued by low reliability and quality problems in the first year of deployment" [1]. A history of failed software projects shows there is tremendous room for improvement in this discipline. There are many detailed case studies that document the failure of failed software projects [2].

A second prevalent problem in software systems development is with the integration of modules. Integration is the 4th stage of the software development lifecycle, it is an area that lacks process and in many cases is treated as an after-thought. A common approach taken to integration is the Big Bang approach. The entire system is integrated all at once. This is analogous to 'shoving' the system together and hoping for the best. With this approach, there are many opportunities for defects to be introduced into the system. Furthermore, it also becomes very difficult to identify the source of defects [3]. Integration and testing plays a critical role in establishing system functionality as well as detecting and removing defects. However, there is limited work done in the way of software integration standards. This lack of process in software integration negatively influences the overall quality of the system.

Authors are with Faculty of Engineering, University of Technology Sydney, NSW 2007, Australia (e-mails: zenon.chaczko@uts.edu.au, Essam.Rahali@uts.edu.au, Rizwan.Tariq@uts.edu.au).

There is a need for clearly defined processes that provides the system development team with a blueprint for building the individual modules into subsystems, and then integrating these subsystems to form a complete system.

Another important issue that needs to be considered is the role of human factors in computer based systems development. Human cognitive processes play a very important role in producing high quality computer systems. Currently, the quality of human skills and how they affect the quality of systems being developed is not well researched. The quality of human skills needs to be measured. In particular, defects directly attributed or resulting from human factors need to be weeded out. We are then equipped with information to measurements the affect of human skills on the quality of the system being integrated.

We propose to address these problems by presenting a single integrated solution. We present a system integration process based on the Six Sigma Model. The objective of the Six Sigma System Integration (3SI) process is to improve the quality of the system by providing the system (hardware and software) development team with a road map to aid the integration effort.

The Six Sigma process improvement model was conceived at Motorola to improve the quality of manufacturing processes. Since its conception at Motorola, it has been adopted in a number of industries including finance, healthcare and product development. The Six Sigma model uses a five-step problem solving methodology to improve an organizations development process. These steps are Define, Measure, Analyse, Improve and Control (DMAIC). In Six Sigma variation is seen as the cause of defects both in the process and the product. A defective product is a result of variation that exists between a target value specified by the customer and the actual output value displayed by a product. The Six Sigma model focuses on reducing this variation to a degree such that six sigma's of variation fit between the target and the customer's specification limits [4]. The Six Sigma is a proven quality improvement methodology that has achieved widespread success in many different disciplines. It encapsulates core concepts that are aligned with our objective of producing quality system. Furthermore, taking an abstract view of the System Development Lifecycle and DMAIC, many common themes, ideas and processes emerge between the two.

This paper has been divided into three sections. In the first we provide an overview of the 3SI process. In the second section we present a system integration solution in the form of

a conceptual framework. This solution is a based on the 3SI process. In the final section we discuss the implementation of the 3SI process.

II. The 3SI Process

The Six Sigma System Integration (3SI) process draws on key processes and concepts from system engineering and Six Sigma. These have been incorporated into a logical and cohesive series of steps designed to improve the quality of system systems. In the 3SI, integration is treated as a complete process, rather than a single task of piecing the system together. It treats integration as a key contributor to the overall quality of the system. It emphasises the use of customer requirements as the driver of all integration activities. This integration process places a specific focus on improving the reliability of the system. The integrators need to carry out a system reliability assessment of the system, forcing them to consider the customer's reliability requirements from the outset. The 3SI process is a crossbreed between system development and manufacturing. It shows that the Six Sigma DMAIC methodology can be adopted in system development. The 3SI Process consists of five phases, and each phase contains a number of tasks that need to be completed by the integrators.

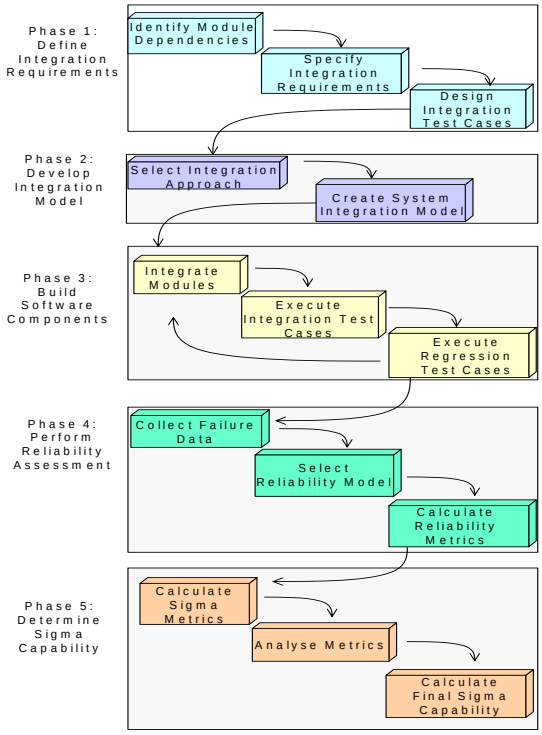

Fig. 1 An overview of the 3SI process

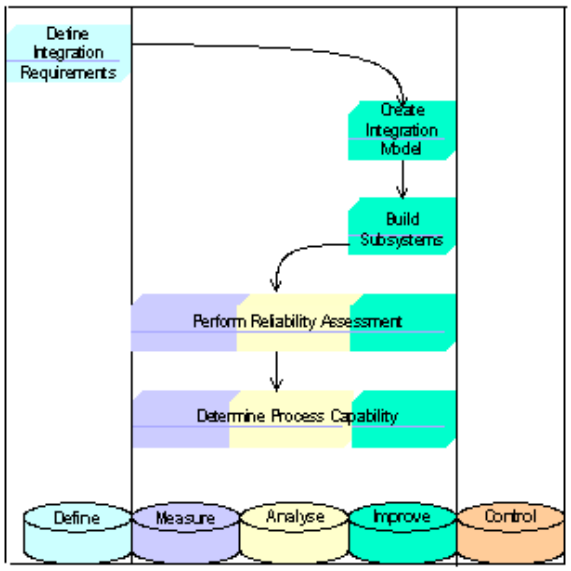

Fig. 2 Mapping between DMAIC and the 3SI process

A. Process Mapping

The Six Sigma DMAIC methodology cannot be adopted in system integration in its original form. The reason for this is Six Sigma was developed for application in the manufacturing industry. Hence it does not have any focus on system systems development. For DMAIC to make sense in system development, the most applicable concepts have been taken from each phase and customised to become system focused.

The structure of the 3SI process is significantly different to DMAIC. The reason for this is that DMAIC is focused purely on process improvement however 3SI is not a process improvement model; it is a system integration solution. Fig. 4 shows the different phases from DMAIC that have been used to build the 3SI process. It shows an alignment between DMAIC and the phases of the 3SI process. This alignment shows the foundation of each phased in the 3SI process.

The first phase "Define Integration Requirements" of the 3SI process is based on the Define phase from DMAIC. Key activities that have been adopted include identifying the needs of the customers, specifying a set of requirements and setting quality targets [4]. The second and third phases are based on the DMAIC Improve phase. Key activities that have been incorporated include process optimisation, design of solutions to remove defects, identification of the most suitable solution and then implementation [5]. The fourth and fifth phases are based on three phases of DMAIC. These are the Measure, Analyse and Improve phases. Key activities that have been incorporated include collecting process data to verify the presence of defects, using process data to calculate process capability and using this to verify whether Six Sigma capability levels have been achieved [4]. There is no direct mapping from 3SI to the DMAIC Control phase because managerial tasks are executed throughout the entire 3SI process to ensure successful completion of system integration.

B. Phase 1: Define Integration Requirements

In this phase the integrators need to define a clear, concise, unambiguous, and testable set of "integration requirements" [6]. By specifying a set of integration requirements, it ensures that integration activities are geared towards satisfying customer needs and the system is not composed together using a Big Bang approach. In this phase the integrators will be required to execute three different tasks. The first two tasks focus on defining the integration requirements, and the third tasks looks at how these requirements can be tested.

C. Phase 2: Develop Integration Model

The purpose of phase two is to design an integration model for the system. The integration model will be a blueprint that shows the modules that need to be integrated, and the order in which they need to be integrated. This phase consist of only two tasks, both targeted at eliciting two key pieces of information used to form the integration model.

D. Build System Subsystems

In this phase, the individual modules are integrated to form subsystems, and subsystems are integrated to produce the final software system. There are three tasks in this phase, and all three tasks are executed each time two modules are integrated to satisfy the integration requirements. It is a cycle that continues until all three tasks have been executed for every pair of modules in the system.

E. Perform Reliability Assessment

One of the key reasons we are adopting Six Sigma in system integration is to improve the reliability of the system. By doing this we work towards improving the overall system quality. This phase focuses on collecting failure data, and then using this data to calculate reliability metrics.

F. Determine Sigma Capability

In this phase, we determine the Sigma capability of our process. Calculating the sigma capability gives us information about the quality of the process and the quality of the product. Firstly, it indicates the extent to which the 3SI process is capable of satisfying the integration requirements. Although in this phase we are calculating the process capability, we are using this information to assess the quality of the system. Based on the assessment completed in this phase the integrators can see whether the 3SI process is at the Six Sigma capability level.

III. The 3SI Conceptual Framework

In the preceding section we introduced a Six Sigma based system integration process. For this process to be adopted during the Integration phase of the system development lifecycle, it needs to be implemented.

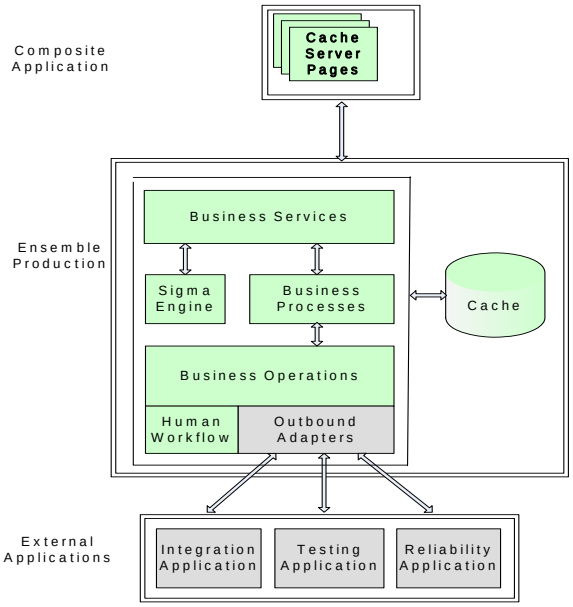

Fig. 3 The 3SI conceptual framework

The 3SI Conceptual Framework is an integration solution. It represents a high-level blueprint that shows how the 3SI process can be implemented with CASE tools. It is presented as a model made up of a series of components. These components interact with one another to execute a specific task in the 3SI process. We have adopted CASE tools to reduce the complexity and time required to execute tasks in the 3SI Process. The 3SI conceptual framework can be viewed as having three 'layers'.

1) Composite Application

This is a web application that is used by the integrators to communicate with the Ensemble [9], [11] production. Integration with the 3SI process begins at the composite application. The integrators use it to invoke or trigger the 3SI process.

2) Ensemble Production

The 3SI process has been specified as an Ensemble production. The Ensemble production accepts and directs requests made through the composite application. Ensemble controls the execution of every task in each phase of the 3SI Process.

3) External Applications

A number of external applications are required to execute specific tasks in the 3SI process. At present we have not identified a suitable Integration application to automate the task of joining two software modules together. However, we have still included it in our conceptual framework for completeness. The consequences of this are that the integrators are required to manually join modules when they reach Phase 3 of the 3SI process. The testing application we have selected is Mercury Test Director. It will be responsible for executing tasks two and three in Phase 3 of the 3SI process. Test Director will execute automated integration and

regression tests [7]. The Computer-Aided System Reliability Estimation (CASRE) tool will be used to perform Reliability Assessments of the system based on failure data collected by the integrators [8].

In Fig. 3, components that have been shaded in green are part of the current implementation. Those components not a part of the current implementation have been shaded in grey. At present, the Ensemble production does not communicate with any external applications using outbound adapters. However, both the adapters and the external applications have still been included for the purpose of presenting a complete integration solution.

IV. Implementing the 3SI Process

The 3SI Process has been partially implemented to demonstrate that it is not just a theoretical concept. Rather, it has basis as an executable integration solution. At present we focused on laying the foundation for future work by presenting key concepts and demonstrating their application at an abstract level. As discussed earlier, in the current implementation, the Ensemble production does not communicate with external applications.

A. Composite Application

The composite application consists of eight web pages created using HTML and Cache Object Script. It is used to invoke the 3SI Ensemble production using Cache Server Pages. Each Cache Server Page is designed to invoke a specific phase of the 3SI process. Fig. 3 depicts the structure of the composite application. There are eight web pages, with five of these are dedicated to invoking each of the business services that exist in the 3SI Production. The top-level tier represents the home page used to access every other page in the composite application. The middle-level tier represents the CS Pages used to invoke 3SI Business Services. The bottom-level tier represents the actual classes that have been written for each business service. Within the Ensemble production, each business service component has been defined as a class.

B. Ensemble Production

To create an executable process with Ensemble we define a Production. The production consists of Business Services, Business Processes, and Business Operations that are all integrated. There are also a number of additional elements that need to be created to implement the 3SI process. These include messages, human workflow, and data management. Fig. 4 is a model of the 3SI Ensemble production. It shows all of the business services, business processes and business operations that have been created to implement the 3SI process. The business services are represented as inward arrows marked with a yellow diamond. The business processes are marked with a yellow circle. The business operations are represented as outward arrows marked with a yellow diamond.

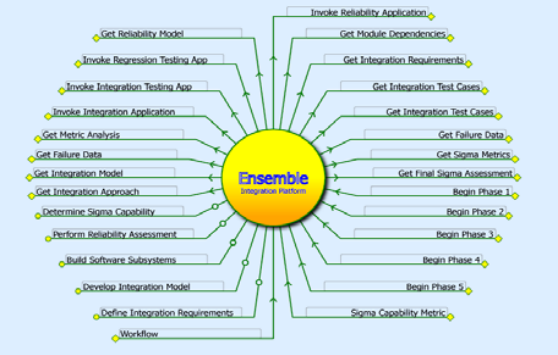

Fig. 4 Overview of the 3SI Ensemble production

1) Business Services

The role of the Business Service is to accept requests from the composite application, and invoke the relevant business process to service the request [9]. In the 3SI Ensemble Production there are five different business services. There is a business service associated with every single business process. Each business service is a Class definition created using Cache Object Script.

2) Business Processes

In the Ensemble Production, we have taken each phase of the 3SI Process and described it as a separate business process. We have five business processes, and each process corresponds to a phase in the 3SI Process. Each process is executed independent of the other processes.

The role of the 3SI Process is to dictate which tasks need to be completed and the order in which they must be completed to satisfy a request [10] The actual execution of the task is carried out by the business operations.

The 3SI process has been embedded into the Ensemble production using the Business Process Language (BPL). Using BPL, the process is transformed from an English description into an XML class that can be executed by Ensemble.

3) Business Operations

The role of the business operation is to execute tasks that have been specified in the 3SI process [9]. Every single task in the 3SI business process is linked to a business operation. Each business operation class has been written using Cache Object Script and contains an XML message map.

4) Human Workflow

There are many tasks in the 3SI Process that cannot be automated and have to be completed by the integrators. Ensemble allows us to make humans a part of process execution. This is done using standard workflow operations from the Ensemble Library. Instead of executing a standard business operation that has been custom written, a task that requires human intervention needs to be completed uses a workflow operation. Prior to using any workflow operations, we are required to specify users and roles [11].

5) Messages

All communication between business services, business processes and business operations takes place via messages.

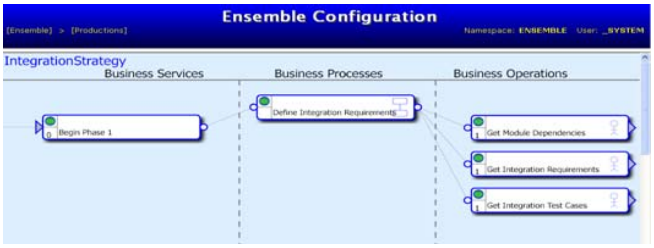

Fig. 5 Implementation model for Phase 1 of the 3SI process

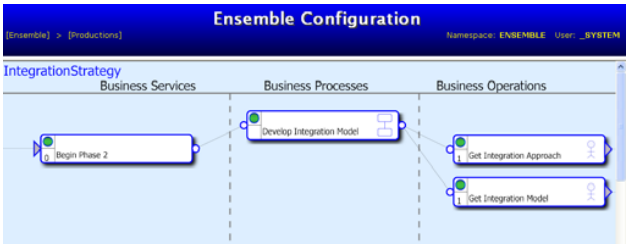

Fig. 6 Implementation model for Phase 2 of the 3SI process

There are two types of messages that can be sent between Ensemble components, these are requests and responses [9].

6) Data Management

The Ensemble integration platform is built on a full-scale Post-relational database known as Cache (Refer to Fig. 3). In this project, Cache has two main purposes. Firstly, it is used to store the class definitions we have created to implement the components in our Ensemble production. The second source of data is produced when the 3SI Process is executed. During each phase of the 3SI process, data is produced either by the integrators or by external applications.

C. Implementation Model

Each phase of the 3SI process has been implemented using business services, business processes and business operations. Phase 1 'Define Integration Requirements' has been implemented using one business service, the business process definition and three business operations. Figure 5 shows the implementation model phase 5. The connections between components mean that the component on the left side of the connection makes a request to the component on the right side. The first task, "Identify Module Dependencies" is completed by invoking the operation "Get Module Dependencies'. The second task, is completed by invoking the operation "Get Integration Requirements". The third and final task is completed by invoking the operation 'Get Integration Test Cases'. All of these business operations are Workflow operations. All workflow operations are marked with a human stick figure in the implementation model.

Phase 2 of the 3SI process is implemented using one business service, the business process definition and two business operations. The process definition consists of two tasks. The first task, "Select integration Approach" is completed by invoking the operation "Get Integration Approach". The second task, "Create System Integration Model" is completed by invoking the operation "Get Integration Model". Both of these operations are Workflow operations. The configuration manager represents workflow operations by marking them with a human stick figure. Phase 3 of the 3SI process is implemented using one business service, the business process definition and three business operations. The first two phases of the 3SI process are implemented using workflow operations only. Phase 4 of the 3SI process is implemented using one business service, the business process definition and two business operations.

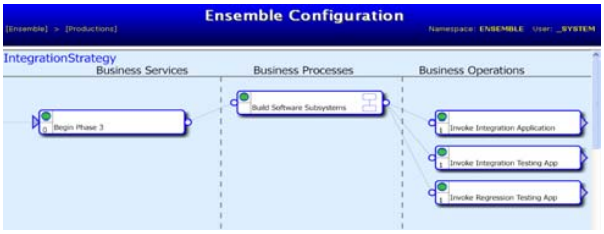

Fig. 7 Implementation model for Phase 3 of the 3SI process

However, phase three does not use any Workflow Operations. Every single operation in this phase has to be custom written. The process definition for phase three consists of three tasks. Each task needs to be completed by invoking three different external applications. The first operation to be executed is "Invoke Integration Application". The second and third business operations are required to invoke the Mercury Test Director via a pair of outbound adapters.

The process definition consists of three tasks. These tasks are implemented using a combination of Workflow and Custom business operations. The first task "Collect Failure Data" is implemented as a Workflow Operation. However, once an integrator has accepted responsibility for this task to complete this task, the integrators need to use the Mercury Test Director. Hence, to complete this task, a combination of Workflow and Custom business operations are used. Like the first task in this phase, second task "Select Reliability Model" is also implemented as a combination of Workflow and Custom business operations. It requires the cognitive processes of the integrators to interpret the results produced by CASRE and decide which reliability is most appropriate. The third task "Calculate Reliability Metrics" is implemented as a pure Custom business operation.

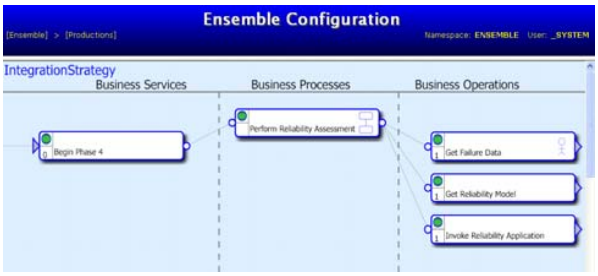

Fig. 8 Implementation model for Phase 4 of the 3SI process

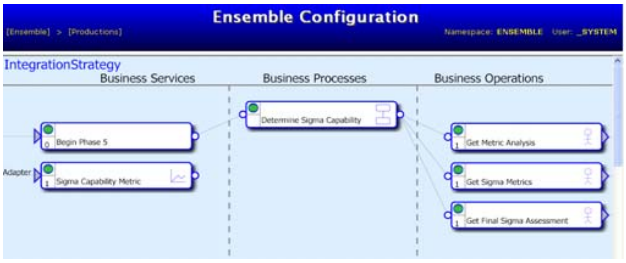

Fig. 9 Implementation model for Phase 5 of the 3SI process

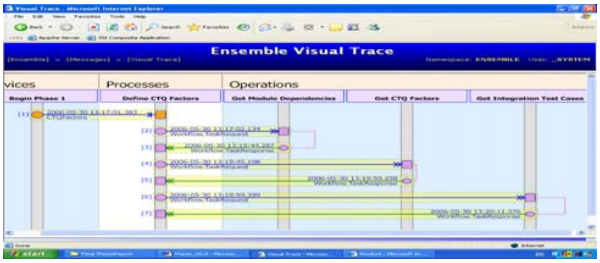

Fig. 10 Testing Phase 1of the 3SI process with Ensemble

Initially, we decided to create a Sigma Engine, this was a business service that is capable of retrieving data from the Cache post-relational database, and then calculating the 3SI process capability using a specific formula. However, based on further statistical studies we found that the initial mathematical definition that was being used was perhaps somewhat simplistic. We have changed our definition, and found that the Ensemble Business Metric portal is not capable of calculating the Sigma capability for system processes. Due to limitations of Ensemble, every single task in this phase has been implemented as a pure Workflow operation that relies on the integrators.

V. Testing the 3SI Ensemble Production

The 3SI Ensemble production has been tested by running through a number of usage scenarios. In a usage scenario, we execute each task in that particular phase of the 3SI process. If we consider Phase 1 'Define Integration Requirements', the usage scenario would require us to execute all three tasks in this phase, and then trace the behaviour of these tasks.

The Ensemble testing portal is used to view the status of every single interaction that occurs between business services, business processes, and business operations in Phase 1. The contents of every single message sent during process execution are viewed. It is important to trace every single message to identify any run-time errors that may have occurred.

Fig. 10 shows the message browser contents for the process "Define Integrations Requirements". The first task "Identify Module Dependencies" is executed by invoking the operation "Get module Dependencies" is invoked. Once this business operation has finished executing it returns a response to the business process. These sequences of events are repeated for each task in the process. The execution of each task is captured in the pair of request and response messages for the task. A purple arrow connects each pair.

From the message browser contents, it can be seen that every single task has been executed, and Phase 1 of the 3SI process has been completed without any errors.

Using the Ensemble testing portal, in particular the Message Browser, we can determine whether our composite application, business services, business processes, and business operations have been successfully integrated. This form of testing has been applied to every single phase of the 3SI process.

VI. Conclusion

The application of Six Sigma in system is quiet controversial, as such it has not received widespread acceptance in the professional community. There have been many arguments refuting the application of Six Sigma to system development. During the course of this project, we did not find any research or projects where Six Sigma has been adopted in system integration. Due to the inherent similarities between manufacturing and system integration, we believe that Six Sigma is most adoptable during the integration and testing phase of the system development life cycle. Furthermore, there is also a strong correlation between the Six Sigma and the DMAIC methodology.

We have designed a Six Sigma System Integration (3SI) process whose foundations can be traced to the Six Sigma DMAIC model. We believe we have shown that the Six Sigma model in its original form is not directly applicable to system integration. However, this model can be customised and adopted in system integration to produce high quality systems.

References

[1] Jones, C., Patterns of large software systems: Failure and success. Computer, 1995. 28(3): p. 86-87.
[2] Flowers, S., Software Failure: Management Failure. 1996, West Sussex: John Wiley & Sons.
[3] Behforooz, A. and F.J. Hudson, Software Engineering Fundamentals. 1996, New York: Oxford University Press, Inc.
[4] Pande, P.S., R.P. Neuman, and R.R. Cavanagh, The Six Sigma way: how GE, Motorola, and other top companies are honing their performance. 2000, New York: McGraw-Hill.
[5] Pande, P. and L. Holpp, What is six sigma? 2002, New York: McGraw-Hill.
[6] Bluescribe. What is an integration blueprint made up of? [Internet] 2005 [cited 2006 23/04]; Available from: http://www.bluescribe.com/Blue/bluescribe_offering.cfm.
[7] Mercury. Mercury Test Director. [Internet] 2006 [cited 2006 30/03]; Available from: http://www.mercury.com/us/products/quality-center/testdirector/.
[8] Stark, G., Appendix A: Software Reliability Tools, in Handbook of Software Reliability Engineering, M.R. Lyu, Editor. 1996.
[9] InterSystems, Developing Ensemble Productions. 2004, Cambridge: InterSystems.
[10] InterSystems, Ensemble Business Process Language Reference. 2004, Cambridge: InterSystems.
[11] InterSystems, Using Workflow with Ensemble. 2004, Cambridge: InterSystems.