

Hybridization of GFT with Life Cycle Models

P. Patchaiammal¹, R. Thirumalaiselvi²

Assistant Professor, Department of Computer Applications, Bharath University, Chennai, India¹

Assistant Professor, Department of Computer Science, Govt. Arts College (Men), (Autonomous), Chennai, India²

ABSTRACT: In software development there is objectivity between quantity, cost and time. Software methodologies are concerned with the process of creating software - not so much in the technical side, but the organizational aspects. Technological change of 21st century systems is intensive with exponential speed, so technology is obsolete before project completion. The main goal of every software developer and every development team is to deliver the highest possible value to employers and customers. A buyer must choose Software Development Company or Agency that fulfills the clients all requirements and develop Best, Robust and Errorless Software for the Client. This study enlightened various SDLC methodologies with the need for hybridization in SDLC and proposed a new methodology GFT (Genetic Fault Taxonomy). By hybridizing Genetic Fault Taxonomy in life cycle models a Software Development Company can develop superlative and Errorless Software for its client and fulfill all the requirements of the user.

KEYWORDS: Genetic Fault Taxonomy(GFT), Software, Agile, Devops, Genetic Gain Algorithm(GGA), Hybridization.

I. INTRODUCTION

In this competitive Internet world, everyone wants to design their Software for its business. This software functionality must solve the user's need. Many users want new functionalities in their Software and Developer must provide this functionality in that software.

Numerous types of SDLC models like Waterfall, Agile, Spiral, Devops are available. Traditional methods are well-suited for predictability whereas Agile and Devops are well-suited for high uncertainty. It comes down to efficiency versus effectiveness. These entire SDLC models follow the steps like requirement gathering and analysis, System analysis, system Design, Coding, Testing and evolution for developing errorless software. All these steps are described in Fig 1.

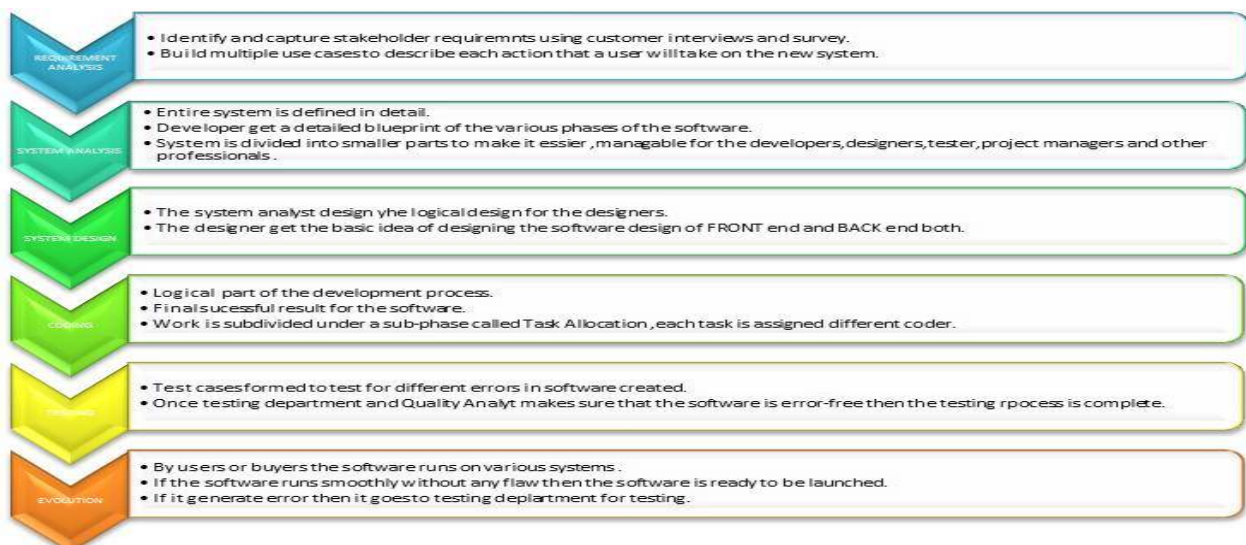


Fig. 1 Steps of SDLC models (Waterfall, Spiral, Agile, etc.)

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 12, December 2016

II. WORK-IN-PROCESS

Operational needs are assessed and the concept of operations is completed, the next step and typically the first task on development projects are to discover, elicit, collect, define and analyze requirements. Requirements cover various aspects of a capability or system like user needs, behavioral, quality, implementation, etc.

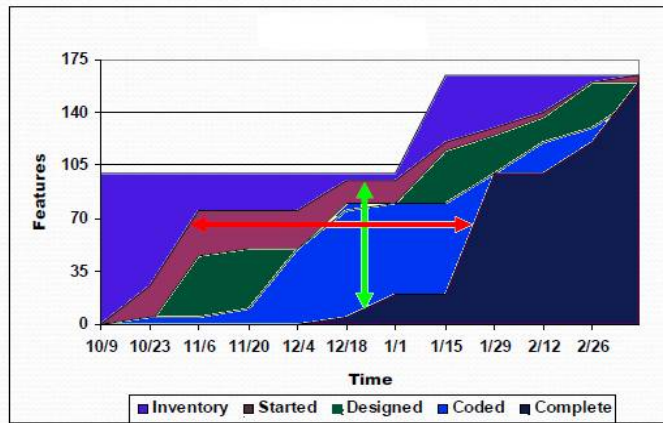


Fig. 2 Work-in-process Traditional models

The above Fig 2 shows the work in process of traditional models like water fall , V-model , incremental , spiral and iterative models. According to the chart, the work initiated at the month of October and is started in the end of October month. There is a bigtime lead between start and complete of the project. The final result to the customer is released them after 2 months. Total time period to complete the software is 5 months. Therefor the work process takes longest lead time to complete.

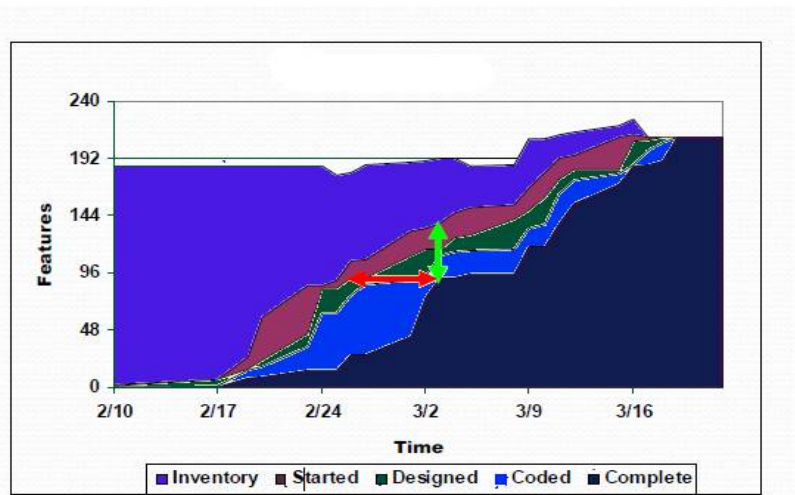


Fig. 3 Work –in-process Agile and Devops models

The above Fig 3 shows the work in process of agile methodologies like XP, Scrum, Kanban and Devops. According to the chart the work initiated at the month of October and simultaneously started. There is a very small time lead to start design and project completion. Total time period to complete is less compared to traditional models. It takes one-and-a-half month period to complete. At regular time interval these methodologies ensures that the capabilities are delivered at regular time intervals.

International Journal of Innovative Research in Science, Engineering and Technology



(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 12, December 2016

III. SDLC METHODOLOGIES

All the SDLC methodologies and its pros and cons are described in the following table 1.

Table 1 SDLC Methodologies

Methodology	Definition	Pros	Cons
<p>1. Water fall</p> 	<p>The waterfall model is a sequential (non-iterative) design process, used in software development processes, in which progress is seen as flowing downwards (like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, production/implementation and maintenance</p>	<ul style="list-style-type: none"> ➤ Simple and easy to understand and use. ➤ Easy to manage due. ➤ Phases are processed and completed one at a time. ➤ Works well for smaller projects where requirements are very well understood. 	<ul style="list-style-type: none"> ➤ Once an application is in the testing stage, it is very difficult to go back and change. ➤ No working software is produced until late during the life cycle. ➤ High amounts of risk and uncertainty. ➤ Not a good model for complex and object-oriented projects. ➤ Poor model for long and ongoing project
<p>2. V model</p> 	<p>The V - model is SDLC model where execution of processes happens in a sequential manner in V-shape. It is also known as Verification and Validation model. V - Model is an extension of the waterfall model and is based on association of a testing phase for each corresponding development stage.</p>	<ul style="list-style-type: none"> ➤ Simple and easy to use. ➤ Saves a lot of time. ➤ Proactive defect tracking. ➤ Avoids downward flow of the defects. ➤ Work well for small projects. 	<ul style="list-style-type: none"> ➤ Very rigid. ➤ Less flexible. ➤ No early prototype of software produced.

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

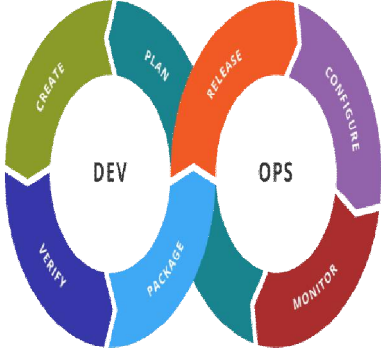
Vol. 5, Issue 12, December 2016

<p>3. Spiral model</p>	<p>The spiral model is a risk-driven process model generator for software projects. Based on the unique risk patterns of a given project, the spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall, or evolutionary prototyping.</p>	<ul style="list-style-type: none"> ➤ The high amount of risk analysis. ➤ Good for large and critical projects. ➤ Strong approval and documentation control. ➤ Additional Functionality can be added at a later date. ➤ Software is produced early 	<ul style="list-style-type: none"> ➤ Can be a costly model to use. ➤ Risk analysis requires highly specific expertise. ➤ Project's success is highly dependent on the risk analysis phase. ➤ Doesn't work well for smaller projects.
<p>4. Agile.</p>	<p>Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations</p>	<ul style="list-style-type: none"> ➤ Customer satisfaction by rapid, continuous delivery. ➤ People and interactions are emphasized ➤ Working software is delivered frequently (weeks rather than months). ➤ Close, daily cooperation between business people and developers. ➤ Continuous attention to technical excellence and good design. ➤ Regular adaptation to changing circumstances. 	<ul style="list-style-type: none"> ➤ In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle. ➤ There is a lack of emphasis on necessary designing and documentation. ➤ The project can easily get taken off track if the customer representative is not clear what the final outcome that they want.

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 12, December 2016

<p>5. devops.</p> 	<p>DevOps (development and operations) is an enterprise software development phrase used to mean a type of agile relationship between Development and IT Operations. The goal of DevOps is to change and improve the relationship by advocating better communication and collaboration between the two business units.</p>	<ul style="list-style-type: none"> ➤ Functional specialization ➤ Consistency. ➤ Economies of scale. ➤ Flexibility to reallocate resources within department. 	<ul style="list-style-type: none"> ➤ Lack of ownership. ➤ Delays caused by handovers. ➤ Tendency to create bottlenecks while scaling.
---	--	--	--

From all the above SDLC methodologies Agile is perhaps today’s the most widely used SDLC model. It uses iterative development as a basis, but uses people-centric viewpoint through user feedbacks rather than planning as the primary control mechanism. The feedback is driven by regular tests and releases of the evolving software. DevOps is an extension of Agile. Agile focuses on helping software developers create better software faster. Now that many corporations are hosting their own software in the “Cloud,” there is a need for these Agile practices to be extended to the hosting (operations) side of things. DevOps focuses on helping teams create, deploy, and host software faster and more resiliently.

It is better not to fixate on any given methodology, because the needs/conditions of the company and project are likely to change regularly, and one needs to be flexible in how to approach managing projects if it is to be successful

IV. TODAY’S ENVIRONMENT

The present stage of the software development is highly unstable, global and domestic markets. Customers are demanding and very difficult to please. Also the project plans cannot cope with this level of volatility.

- ✚ Technology:
Nowadays technologies are economic and market driven, dependent and constant change. It is also growing with complexity. So it has lack of usability and nano technology.
- ✚ Organization:
Current trend of organization is downsizing and restructuring. So it needs merging and budget reduction. The result is adapting new bloated processes.
- ✚ Project:
Each project has vague requirements and volatile specifications with shorter schedules. The budget is also smaller. The project work is more and politically sensitive. Some of the projects are very high risk and uncertainty.
- ✚ Software Created:
The result of the above lead to poor quality with schedule and cost overrun software. In some cases there is no profit at all. Finally market share loss occurs for bad reputation.

V. NEED FOR GENETIC HYBRID MODEL

No single methodology is a silver bullet, so the trick is to determine which methods work for and tune that methodology to suit for individual needs. The above mentioned pros and cons in Table 1 shows that even though agile works better there is a need for new model of project management to cope with high-level of uncertainty and ambiguity.

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 12, December 2016

	use it and love it	use it and hate it	use it and no strong positive or negative feelings	don't use it
Waterfall	56 7.4%	129 17.1%	209 27.7%	361 47.8%
Scrum	390 51.0%	36 4.7%	214 28.0%	125 16.3%
Kanban	206 27.5%	10 1.3%	134 17.9%	400 53.3%
Devops	323 42.7%	19 2.5%	167 22.1%	247 32.7%

Fig. 4 Survey of SDLC methodologies usage

Survey is conducted and also data from global survey is shown in the above fig 4. The methodologies and various point view of usage among the people are described in it. From the survey it is clearly shown that none of the methodology is fully used and none of them is fully outdated. There is a need for strong positive feeling of the methodology. In order to create a positive feeling among the people we created new genetic taxonomy by using GGA(Genetic Gain Algorithm)[5]. Our new genetic taxonomy along with the SDLC methodology will reduce the rework in both traditional and agile methodologies.

VI. NEED FOR GFT HYBRIDIZATION

Genetic Fault taxonomy includes all the faults from previous projects in all the SDLC methodologies. This is stored and training data set is formed by hypothesis. The gain is calculated by using DMM(Decision Making Model). Suggestions are collected by using questionnaires from various teams and are listed in the table 2.

Table 2 Need for Hybridization of GFT with SDLC

S.no.	Team	Suggestion
1	Research	New discoveries, Vague requirement, Simulation, Innovation oriented, Creative solution
2	People centered	Highly-talent people, Cross-functional team, Rich collaboration, Face-to-face interaction, Cohesiveness
3	adaptive	Global threats, Market threats, New customer needs, Changing scope, Change in technology
4	Flexibility	Culture, Attitude, Policies, Processes, Technologies.
5	Customer friendly	Customer interaction, A lot of communication, Customer demos, Customer feedback, Business value focus, Customer satisfactions, Responsive, Sensitivity, Relationships, Contact, Involvement, driven.
6	Quick	New technology, Decision-making, Iterative delivery cycle, Time-to-market Mover capability
7	Disciplined	Lightweight strategy, Lightweight plans, Lightweight lifecycles, Security engineering, Light requirements, Light architecture, Lightweight design. Code reviews.

VII. GENETIC FAULT TAXONOMY MODEL

It is Light, flexible, collaborative and adaptive market –centric project management model. This fault taxonomy can be hybrid with any SDLC methodologies to reduce the rework. This taxonomy includes activities and milestones that have long been part of the fault collection by Goal Question Metrics Approach[5]. Each step within the sequence is completed before moving on to the next step. Faults may be added or modified depending on the scope and type of project or task.

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 12, December 2016

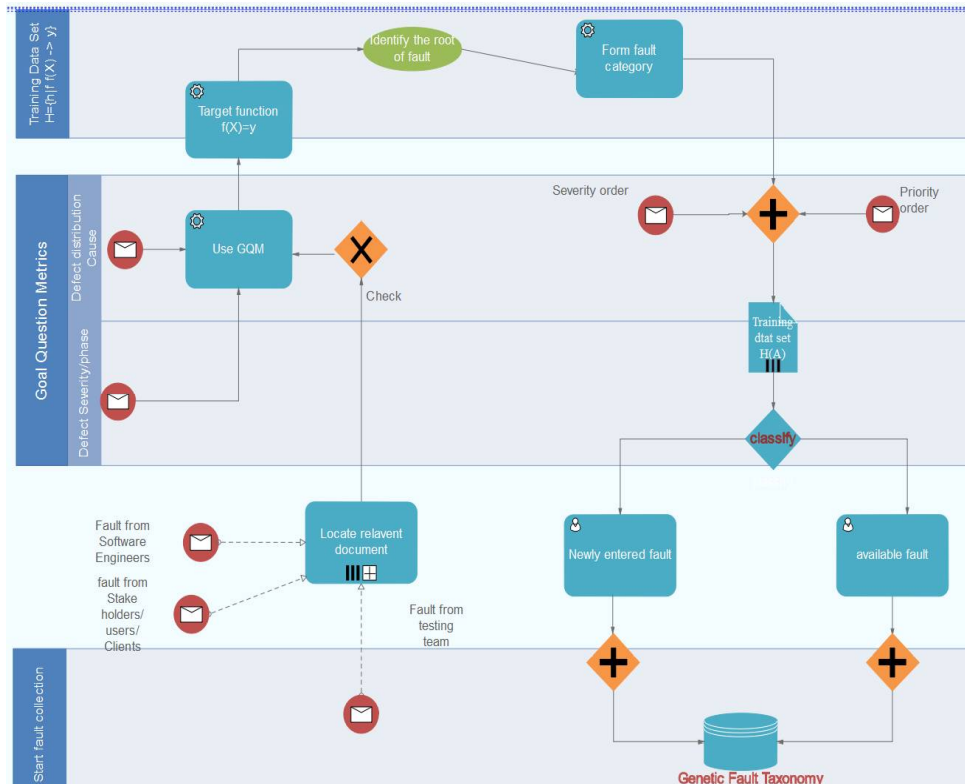


Fig.5 Genetic Fault Taxonomy

Genetic fault taxonomy is described in Fig 5. The taxonomy model is formed by collecting fault by using GQM[5]. The faults localization is done by forming root analysis of the fault. The root of the fault is identified and the Gain value is calculated by GGA[6]. According to the Gain value fault category training data set is formed. The data set is the prioritized by severity of the fault. Finally fault classification is done by DMM[6]. If the fault is deputed the previous fault the fault taxonomy is updated otherwise the fault is added in the Genetic Fault Taxonomy.

VIII. HYBRIDIZATION OF GENETIC FAULT TAXONOMY

The genetic fault taxonomy is combined with any SDLC model to reduce the rework done and also increase the quality. Various SDLC models and phase of GFA is mentioned in Table 3.

Table 3 GFA hybridization with SDLC models:

S.no	SDLC model	GFA fit phase	Rework reduction result
1	Water fall	Requirement Testing Design Unit	Good
2	V model	Requirement Testing Design Unit	Moderate
3	Spiral model	Requirement Testing Design Unit	Good
4	Agile methodologies	From the beginning	Best
5	Devops	Any time	Excellent

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 12, December 2016

A. ADVANTAGE OF HYBRIDIZATION:

- ✚ Flexible process:
It is flexible to manage projects under uncertainty, urgency and a need for unique expertise.
- ✚ Valuable principles:
Used to help project teams in coming to grips with a challenging environment.
- ✚ Manageability:
It managing the flow of human thoughts, emotions, and interactions in a way that produces business value.
- ✚ Reliability:
Rapidly and reliably creating value by engaging customers, continuously learning and adapting.

B. GOALS OF HYBRIDIZATION

This hybridization is formed to focus on improving quality, lead time and reduce rework with initial changes.

- Goal 1: Optimize existing processes rather than change them.
- Goal 2: Deliver high product quality to build stakeholder trust.
- Goal 3: Reduce long lead times and stabilize them.
- Goal 4: achieve sustainable pace work-life balance.
- Goal 5: Provide process slack for process improvement.
- Goal 6: Simplify workload prioritization of customer needs
- Goal 7: Provide transparency into design and operations.
- Goal 8: Strive for process maturity to improve performance.

IX. CONCLUSION

In this paper Genetic Fault Taxonomy is introduced. This model is based on principles for product development flow and mathematical hypothetical theory. GFT can be hybridizing with any SDLC models. The pragmatic principles of this hybridization are quality, reduce reworkflow, and Learn new skills. In future we plan to implement the GFT by forming bug report to effective rework reduction.

REFERENCES

- [1] ArdhenduMandal, S. C. Pal "Identifying the Reasons for Software Project Failure and Some of their Proposed Remedial through BRIDGE Process Models", International Journal of Computer Sciences and Engineering Vol.- 3(1), PP(118-126), E-ISSN: 2347-2693. Feb 2015.
- [2] Pandiyani G. and P. Krishnakumari, "An Efficient Software Defect Prediction Model Using Optimized Tabu Search Branch and Bound Procedure", ARPN Journal of Engineering and Applied Sciences ISSN 1819-6608, January 2015.
- [3] DeepinderKaur, "A Comparative Study of Various Distance Measures for Software fault prediction", International Journal of Computer Trends and Technology (IJCTT) – volume 17 Issue 3, November 2014.
- [4] Parvinder S. Sandhu, Satish Kumar Dhiman, AnmolGoyal, "A Genetic Algorithm Based Classification Approach for Finding Fault Prone Classes", International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol:3, No:12, 2009.
- [5] P.Patchaiammal, R.Thirumalaiselvi "Machine Learning Based Approach for Predicting Fault in Software Engineering by GMFPA: A Survey", International Conference on Computing and Intelligence Systems-Volume: 04, Special Issue: March 2015, Pages: 1358 – 1363.
- [6] P.Patchaiammal, R.Thirumalaiselvi "FAULT PREDICTION TECHNIQUES USING DMM AND GGA", *i-manager's Journal on Software Engineering*, Vol. 9 No. 1 July - September 2014.
- [7] AditiSanyal, Balraj Singh, "A Systematic Literature Survey on Various Techniques for Software Fault Prediction", International Journal of Advanced Research in Computer Science and Software Engineering, January 2014.
- [8] <https://dzone.com/articles/how-774-developers-really-feel-about-agile>
- [9] <https://www.quora.com/Is-DevOps-an-extension-of-Agile-Or-Agile-on-Steroids>
- [10] <http://www.agileweboperations.com/lean-agile-devops-related>
- [11] <https://techbeacon.com/back-waterfall-when-agile-devops-dont-work-well>
- [12] <http://www.informationweek.com/devops/agile-vs-devops-10-ways-theyre-different/d/d-id/1326121>
- [13] https://www.tutorialspoint.com/sdlc/sdlc_v_model.htm
- [14] <https://theagileadmin.com/what-is-devops/>
- [15] https://en.wikipedia.org/wiki/Agile_software_development
- [16] <https://www.sitepoint.com/devops-by-example-tools-pros-and-cons-of-a-devops-culture/>