



Article

Smart and Agile Manufacturing Framework, A Case Study for Automotive Industry

Gullelala Jadoon ¹, Ikram Ud Din ¹, Ahmad Almogren ^{2,*} and Hisham Almajed ²

¹ Department of Information Technology, The University of Haripur, Haripur 22620, Pakistan; gullelalajadoon@gmail.com (G.J.); ikramuddin205@yahoo.com (I.U.D.)

² Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11633, Saudi Arabia; 438105079@student.ksu.edu.sa

* Correspondence: aalmogren@ksu.edu.sa

Received: 28 September 2020; Accepted: 30 October 2020; Published: 3 November 2020



Abstract: Smartness and agility are two quality measures that are pragmatic to achieve a flexible, maintainable, and adaptable system in any business. The automotive industry also requires an enhanced performance matrix and refinement in the development strategies for manufacturing. The current development models used in automotive manufacturing are not optimal enough; thus, the overall expenditure is not properly managed. Therefore, it is essential to come up with flexible, agile techniques incorporating traceability methods. It overcomes the traditional manufacturing approaches that are usually inflexible, costly, and lack timely customer feedback. The article focuses on significant Requirements Management (RM) activities, including traceability mechanism, smart manufacturing process, and performance evaluation of the proposed methods in the automotive domain. We propose a manufacturing framework that follows smart agile principles along with proper traceability management. Our proposed approach overcomes the complexities generated by traditional manufacturing processes in automotive industries. It gives an insight into the future manufacturing processes in the automotive industries.

Keywords: automotive industry; agile manufacturing; smart requirements management; traceability

1. Introduction

Managing requirements is a critical concern in any innovative vehicle manufacturing from initial planning to the final stages. In such cases, these requirements need to be fulfilled according to the needs, aesthetics, and satisfaction level of the customers. Some quality attributes are also managed to produce the desired results. For this purpose, we use smart requirements gathering and managing tools to provide automated solutions to such situations. In agile, the management of requirements is an independent activity that is incorporated at all the developmental stages of the product. Different frameworks and solutions are provided for better management, without the misuse of resources [1,2]. Modeling system requirements and manufacturing transformations are also vital for such systems. Using big data analytics, we can derive better statistical results for decision making and optimization [3]. The evaluation of such methodologies provides an insight into the organization to identify the needs of its valued customers. In this way, organizations choose smart methodologies along with the deployed methods to make their work easier and satisfactory. Such a blend of techniques provides better developmental strategies because the requirements are well managed. Past studies have proven the effectiveness of such techniques [4].

Management of requirements involves controlling different factors affecting the nature of these requirements. These factors include diversification of models, data analysis, dependency, customer satisfaction, modularization, and usability. A primary concern is to balance all the stated

attributes from baseline activities until the final product [5]. Requirements management is a part of the capability maturity model level two Key Process Area for better project management. This level is called Repeatable or Managed because activities are initiated for better management of the project's developmental activities. As far as project management is concerned, organizations at this level strive for the better quality of their products [6].

The capability maturity model provides a guideline to the automotive companies to improve their processes and products. It gives the insight to innovate the manufacturing activities to gain desired results [7]. The heterogeneity and diversification of modern vehicles is a challenge for automotive manufacturers. Online interactive software systems require less response time compared to vehicular hardware systems, and these factors are balanced using CMMI-DEV (Capability Maturity Model for development). Thus, we can improve the safety and security processes of an organization globally using such smart standards [8,9]. Requirements traceability is an essential concern in smart requirements management because this process involves a bidirectional trace record of all the activities performed in the manufacturing process. This activity covers all the phases of development, from requirements management to maintenance of the automotive system. A large number of manual and automated tools are available for requirements management. The manual means are more effort consuming and costly, while the automated methodologies are efficient in terms of resources [10–12].

Smart requirement management is one of the fundamental project management concerns which provides a strong base for the upcoming development activities. Managing requirements involves several specific practices such as understanding requirements, obtaining commitment, managing changeability of requirements, managing inconsistencies among requirements, and maintaining bidirectional traceability of requirements for change management and other quality assurance activities. For any business organization, it is exceptionally vital to intelligently manage high volumes of data and deliver sophisticated business solutions meeting the explicit practices required by the particular development level of CMMI. Such a large amount of data enables an organization to adopt data-driven strategies and shift to a smart manufacturing processes [13,14]. Presently, Model-Driven Development is turning into a rising methodology for modeling automotive in the best proficient way. In CMMI level two—i.e., Repeatable or Managed—we need to keep up the bidirectional trace of the automotive system that is being developed, for better compliance of user needs. Maintaining a bidirectional trace in such Meta models is a big trial for the engineers, particularly at evaluation and support stages. In this study, we emphasize the latest development and traceability studies for automotive manufacturing and compare their performance. This study will provide an indication to the manufacturers to select the most suitable traceability methodology for their product. The main stages involved in the auto manufacturing process are depicted in Figure 1. These stages are then refined with the core agile manufacturing standards to achieve the best results.

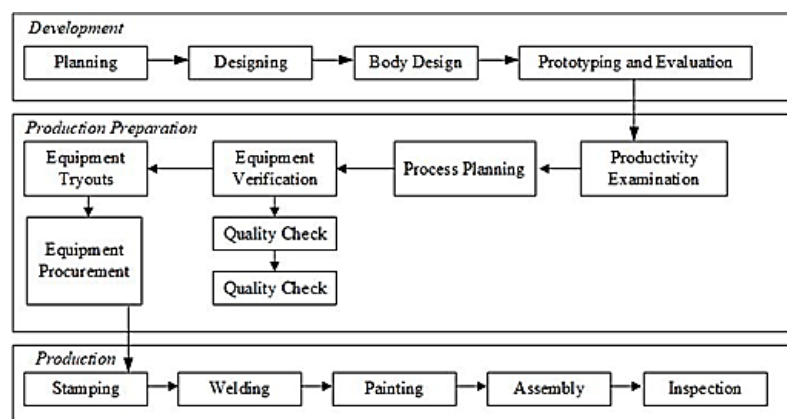


Figure 1. The existing automobile manufacturing process.

In Figure 1, the process comprises three major stages—namely, the development stage, the production preparation stage, and finally the production stage. In the first stage, the major activities involved are planning the whole automobile system, designing different prototypes for the system, and after finalizing the basic design, the body design of the vehicle is designed. At the end of this phase, the final prototype of a vehicle is prepared. The top management further evaluates this prototype.

The second stage is about production preparation, where productivity examination takes place. Afterward, the whole manufacturing process is planned. The equipment used in the automobile is sent for verification using high-quality standards. After several equipment tryouts, the procurement of equipment takes place.

Next is the actual vehicle production phase, where stamping of the metal sheets takes place to form a net shape of the vehicle. Welding is also used along with stamping to design the outer body of the automobile. After painting the body parts, its assembly is done, and the final inspection process is started in which the automobile is fully tested according to the global manufacturing standards.

Requirement Traceability

Requirement traceability is the ability of the system to trace and link each outcome of any development stage back to its requirements specification as well as in the forward direction to the final product. Traceability is vital in any project both from the developmental perspective and validation perspective. The stages involved in the traceability process are defined in Figure 2.

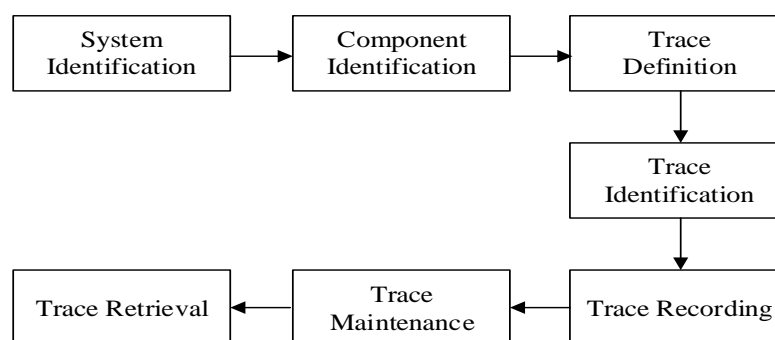


Figure 2. Traceability stages.

In Figure 2, the first stage is system identification, which identifies a system under construction. The automotive system under development needs to be properly analyzed following the requirements specifications document. As a result, we can get information about the scope, sections, and components of the system [15]. The process of traceability needs to be started as the requirements process starts. All the activities followed by the requirements engineering process need to be mapped to record the trace information. In the components identification stage, after identification of the automotive system, it is modularized into manageable compartments called components. Each component is developed based on the priority of requirements. Another follows a component, so the trace continues, and all the information is recorded against each component [16]. In this regard, the components-based traceability approach provides an efficient mechanism to record component traces. In the case of model-oriented development, each metamodel is treated as a component, and the trace is recorded accordingly.

In the Trace Definition stage, the trace process is defined using any tool, methodology, or framework. This methodology is much dependent on the nature of the product. In the case of small systems, manual traces are recorded. As the system becomes large and complex, different traceability tools are used to provide automated traceability of the system [17]. Each trace id is a bidirectional link to all the elements and artifacts representing the unique functionality of the automobile under development. The traceability link is integrated throughout the system to refer to the required functionality. The specific identity of the trace makes it possible to refer to the particular requirement

at each manufacturing stage [18]. In the case of Quality Assurance activities, these bidirectional links are used to record the effect of vehicle alterations after regression testing is carried out. At the post-implementation phase, whenever any component or module is changed, the change is also recorded to measure its ripple effect. All these situations demand proper identification of the trace.

In the Trace Recording stage, each trace entity is recorded by its trace id and is placed in a table by a sequence of its requirement specification. These links are also maintained in any database or Excel sheet manually [19]. The automated tools record the trace information in a tabular form which makes it easy to retrieve it back using any query language [20]. In the Trace Maintenance stage, the maintenance of traceability links is a crucial requirement management concern. Rapid changes in the specifications result in the degradation of trace links. Similarly, the regression testing records the effects of changes made to the system, and after each test, the trace information also needs to be updated to avoid any loss of trace information [21,22].

Finally, in the Trace Retrieval stage, the trace information is retrieved to get statistics about the completion of the automotive components [23]. In testing, this information is retrieved to record the effect of defects in any component, and when it is resolved, the trace information is used to analyze the effect of changes in other modules and subsystems. This trace information is retrieved using a query or is manually retrieved through tables. Our work is based on analyzing requirements management strategies at various manufacturing stages, the related work, and its performance comparison. Thus, research questions that arise are as follows:

1. What are the Major RM activities carried out by automotive companies?
2. What are the different traceability mechanisms in automotive companies?
3. What is the Performance evaluation of current traceability methods in the automotive domain?

The rest of the article is divided into different sections. Related literature is discussed in the Related Work as Section 2, Manufacturing Agile Framework is proposed for the automotive industry in Section 3, and Performance Comparison is carried out in Section 4.

2. Related Work

Different traceability techniques are discussed in literature, which use various methodologies to maintain bidirectional traceability links. A traceability metamodel is designed for railway control system (RCS) implemented in V-Model of system development. This methodology is efficient in maintaining traceability at the component level, i.e., it supports modularization and independence of system components. Links of traceability are maintained using the UPPAL tool [24,25]. The main challenge in implementing this methodology is updating traceability links of requirements when alteration occurs in any component. Energy systems are also switching towards economical and smart monitoring systems to automate functionality and derive voluminous data for decision making.

Another technique that is very efficient in recovering the missing traceability links is refactoring. This method involves making the code artifacts semantically and syntactically aligned, which makes it more understandable and hence improves the traceability links because of code refinements [26]. Thus, we can relate this technique to the hardware components as well. As the automotive systems become more sophisticated day by day, the manual traceability mechanisms also need to be improved for dealing with complex systems. Model-based development also plays a significant role in the latest system development where trace information about different metamodels are maintained, and this trace is used to localize different hardware transformations. It is very crucial to maintain this trace information when different metamodels are integrated [27].

The current product manufacturing trend mostly follows smart agile methods rather than traditional plan-driven approaches like a waterfall [28,29]. These methods include Rational Unified Process, Extreme Programming, Scrum and Feature Driven Development, etc. Among all these models, Extreme Programming is most popular in most of the small and medium companies. The main reason behind this is its manufacturing flexibility for both the clients and the developers.

Due to the quick feedback from clients, rapid changes take place in the manufacturing system [30]. To record these changes, proper traceability is needed, and after the alteration process is completed, the trace also needs to be updated. The SWEBOK also supports this view. These traceability procedures used by different organizations vary from simple matrices, formulas, and tools, to UML models that are mapped to draw traces of outcomes. However, all these methods do not provide surety to obtain the full trace of the system at all the phases of software development [31].

Model-driven development is gaining more importance over the past decade, because of its variation in the manufacturing of different kinds of systems. Researchers have mainly focused on its related features and characteristics. Enormous techniques and methodologies are being implemented to analyze the results in real-time automotive systems. One such technique is the use of soft goal trees for traceability management. Following this method, the trace information is maintained at the baseline and is updated as the changes are made to the system. For this purpose, regression testing is performed to analyze the effect of these changes. The main problem faced in utilizing this technique is the time delay in generating a response because of the large number of model constituents [32,33]. For improving the requirements management, manufacturing, and maintenance tasks, these activities are linked to each other through text retrieval techniques. As different metamodels are also involved in system development, we need to follow the nontextual techniques as well. In this way, information retrieval is enhanced by using both techniques together [34].

The need for traceability arises as the system becomes complex to keep a record of all the functional points. Later on, at the phase of evaluation, we need to perform a variety of regression tests that result in major or minor changes in the automotive. These changes need to be tracked bidirectionally in order to measure the amount of change that the system undergoes. Similarly, the maintenance activities also need to be traced in order to integrate the components. In today's technological environment, where we are dealing with safety and security-critical vehicles, the need for traceability becomes more vital. Most enterprises fail to implement the complete traceability due to immaturity in their manufacturing processes [35,36]. A large number of studies have been carried out to analyze the latest challenges faced in recording and maintaining traceability. The tools and methods have been tested, and their performances analyzed by researchers and experts [37–39].

As the automated software tools are replacing the standard methodologies, the traceability automation also comes with a price. The storage and maintenance of trace links are crucial and challenging too, especially in the case of large applications. Some automated techniques have been implemented at the design level, but they are not implementation-based [40,41]. Managing changes in the automotive industry is very crucial because the clients always want their feedback to be acknowledged. In case of changes, regression testing is performed to analyze the effects of changes on other components. These changes demand constant updates in the trace record. Several manual and automated tools are used to keep a record of such changes [42,43]. Among all the methods, the link maintenance technique is efficient in maintaining the trace record when changes are incorporated [44]. In the case of software systems, a leading trend in software engineering is the global software development (GSD), where the development process is distributed across different locations and communities. Experts from different geographical locations are available to provide their expertise and suggestions in the construction and innovation of a system [45]. In such a developmental environment, the traceability needs to be globally updated [46]. Model-based traceability is replacing the conventional development approach nowadays. The maintenance of trace in such metamodels is essential and challenging too [47]. The transformations require proper record-keeping of the traces, so that the changes can be localized easily [48,49].

3. Proposed Manufacturing Framework

We have proposed an agile manufacturing framework that follows a trace matrix to record every single change and every testing activity related to the functions of the automobile and is defined in Figure 3.

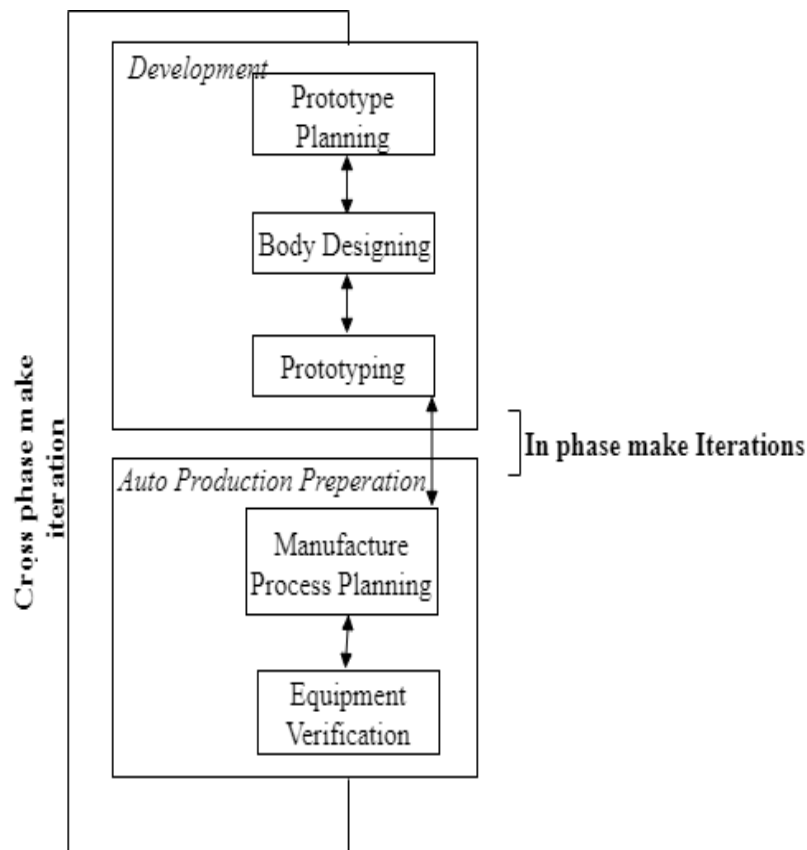


Figure 3. Proposed agile manufacturing framework.

In our proposed framework, we have two main phases of manufacturing—the development phase and the production preparation phase on which the principles of agility are applied. In the first phase, we have three subactivities. At the planning stage, the functionality, feasibility, and resources of the whole auto system are planned. After that, the design is carried out, which includes body design models; at the end of designing, different prototypes are prepared to determine the final model of the automobile. All three activities are carried out incrementally, and these steps can be iterated in order to refine the final model. This iteration is called in-phase iteration, as shown in the figure below.

The second phase of production preparation is comprised of two incremental activities. The first one is the process planning in which the manufacturing process is planned, and then the testing and verification of equipment are carried out. These activities are also interleaved and can be iterated for further refinement. The activities of both phases can also be iterated using cross-phase iteration.

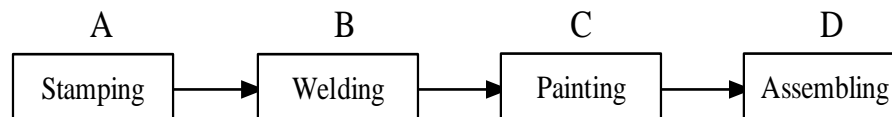
This means that once the whole manufacturing process is finalized and the production equipment is verified, we can reconsider the whole process once again for further improvement. This happens in the cases when the top management, quality insurance team, or any customer representative suggests any enhancement in the system. Table 1 is the trace matrix of the proposed framework, which shows different functions of the vehicle and how these functions are traced using a specific trace id whenever any improvement is suggested.

In this trace matrix, we have randomly selected five functionalities of any arbitrary automobile. Each requirement is assigned a unique requirement ID and a unique trace ID. In the case of analysis or quality assurance activity, the status of each requirement is clearly shown, and a proper description is given along with each requirement. In this way, each functionality is traced throughout the manufacturing process, and its current status is also shown. This information is used to find the current status of each function so that the effort and resources can be estimated in a better way.

Table 1. Trace Matrix.

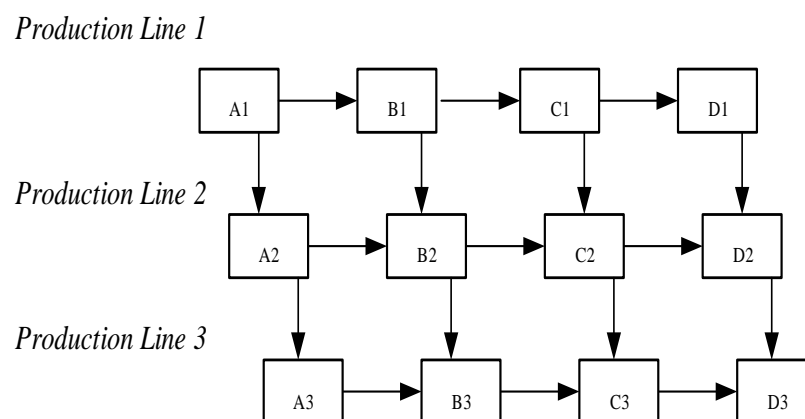
Requirement ID	Trace Scenario	Functionality	Trace ID	Status	Description
R1	Function 1	Fog lamps Assessment	T1	Pass	Done
R2	Function 2	Battery Inspection	T2	Fail	Leakage
R3	Function 3	Electrical Wiring	T3	Fail	Pending
R4	Function 4	Doors Installation	T4	Fail	Nonfixed
R5	Function 5	Engine Assembly	T5	Pass	Done

After finalizing the design and production preparation, the next step is the actual manufacturing process. At this stage, we have finalized the vehicle's design and body, and we need to start the manufacturing of the automobile as shown in Figure 4. The stages involved in hardware manufacturing include the stamping of the metal sheets. The sheets are molded according to the required design and then pressed using heavy machinery. The next step involves the welding of these metal sheets to design the body of the automobile using various welding mechanisms.

**Figure 4.** Automobile Manufacturing Process.

After body design, its painting is carried out using automotive paints. These paints are body protective and also used for decoration. In the end, the automobile components are assembled. Components such as the gearbox, engine, radiator, battery, steering wheel, lights, windshield, bumper, doors, bonnet, fender, and tires, etc., are assembled to form a completely working automobile.

This mass manufacturing process would be more efficient if it follows the agile, incremental manufacturing process. The below figure represents the four necessary manufacturing activities as A, B, C, and D, whereas the numbering 1, 2, 3 represent the first product, other product, and the third product, respectively. The manufacturing process is pipelined in such a way that after completing the first activity of stamping on product one, the other product is brought for stamping, and after that the third one. Similarly, the next three activities are carried out in the same way and shown in Figure 5.

**Figure 5.** The agile incremental manufacturing process.

4. Performance Comparison

The performance of the proposed mechanism can be associated with the past deployed techniques for performance comparison. In a traceability study conducted by Chhabra et al. [34], an automated traceability mechanism is suggested, which is based on textual links. This mechanism provides

text-based trace links for system artifacts that are not very effective for auto development but can aid in its prototyping. Reviews and inspections of activities by experts are also carried out to trace maintenance and testing activities, as proposed by Samalikova et al. [50]. These inspections are helpful in identifying any manufacturing defects. Another bidirectional methodology has been implemented using mappings between system artifacts and the software source code [51]. Thus, we can apply mappings to our automotive metamodels for proper trace. Due to the wide application of agile development, much work has been done for the management of requirements in an agile environment. The involvement and rapid feedback of the customers require a constant update of system artifacts. Different agile traceability frameworks have been suggested for such issues that provide traceability in both directions, which is much useful for agile-based automotive development [52,53].

The overall requirements management is monitored using different tools such as JIRA, UPPAL, etc., which provide bidirectional traceability of the source code based on an automated mechanism. A large number of frameworks and models such as LEGO, QFD (Quality Function Deployment, Kano's model, ISO 9001 audit model, and VTML (Visual Trace Modeling Language), etc., have been proposed for effective traceability management of systems. For sufficient testing, soft goal trees are used to maintain trace links when regression testing is carried out to test the effect of changes on other artifacts. In the case of automotive model-based development, model-driven traceability is carried out using trace links. In such cases, both manual and automated traceability mechanisms are used. Currently, studies have been carried out to trace activities in service-oriented architecture following the model-based approach [54].

In the proposed framework, we have designed an agile approach for automotive manufacturing process. The agility factor of the process enables the manufacturers to reconsider the improvements once they are finalized. Customer satisfaction is the key to success in agile processes and any change is welcomed in the proposed system. The trace matrix provided by the framework is used to record every update related to each requirement. In case of verification activities, all the changes that take place are logged and can be traced bidirectionally. This framework is also effective for the development of customized automotive. In order to assess the performance of this methodology, we have used the MATLAB simulation software. Using the MATLAB fuzzy model, we have simulated Function five of the Trace Matrix, which is 'Engine Assembly' represented by Requirement ID (R5) and Trace ID (T5), and is shown above in Table 1.

We used a fuzzy scale to find out the effect of cost and time fluctuations of Engine Assembly on the production. The cost is classified as less, moderate, high, and very high. Similarly, the time is classified as extremely less, very less, less, and very long. The production type is distributed as moderately agile, agile, less agile, traditional, and less traditional. The fuzzy rules are generated accordingly to measure the effect of cost and time on the production type. The cost, production type, and time scale are supposed to be based on continuous values. The engine assembly time is assumed with a range of 0–60 min while the assembly cost from 0–2000 dollars. The production type ranges from 1–5, showing higher agility and traditional approach on both the extremes.

A two-dimensional result in the above Figure 6 is attained that shows the relationship between engine assembly cost and the product type. It can be seen that engine assembly cost declines with time using an agile approach. For the lesser cost, the production type represents the higher values of agile development. As the cost increases, the production type goes towards the lower values of traditional development. This result clearly shows that agile manufacturing supports efficient production with the least cost possible.

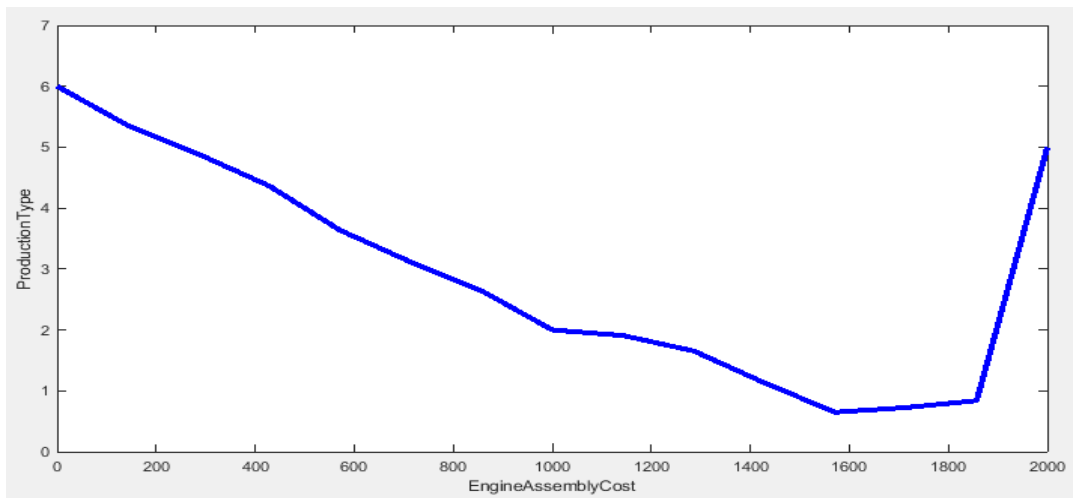


Figure 6. The relation between Engine Assembly Cost and Production Type.

The attained 3D surface in Figure 7 below shows the impact between two input variables, cost and time, on the output production. In this surface graph, agile results are attained based on our defined rules. Less the assembly cost and time, the production type moves towards the higher values of agility (yellow).

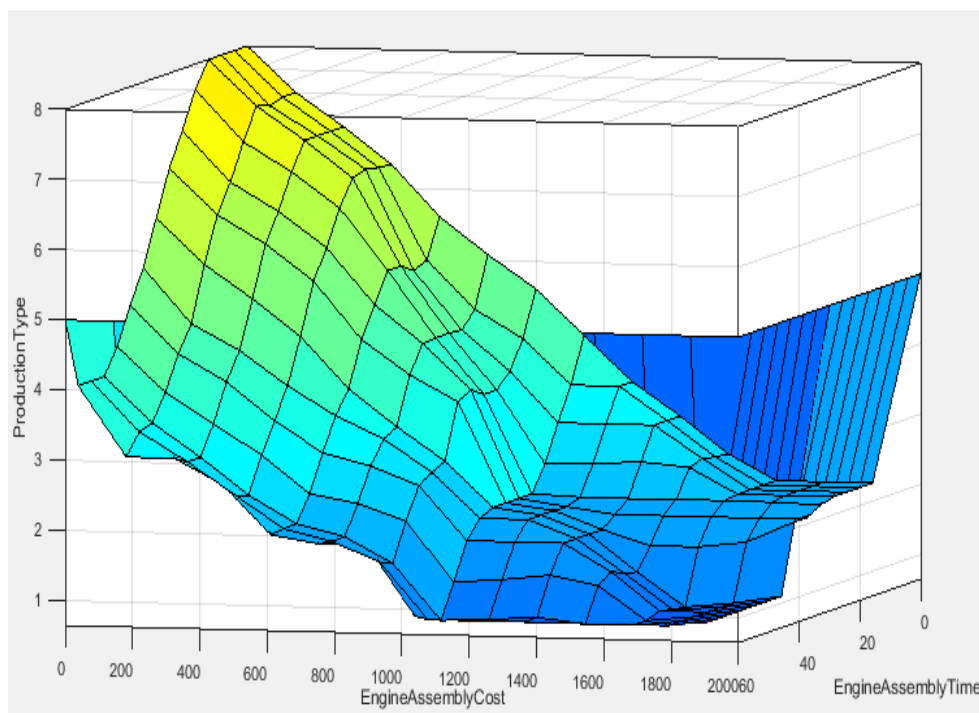


Figure 7. Surface Graph.

For higher cost and more significant manufacturing time, the production type shows the minimum graph of traditional development (blue).

In Figure 8 below, the 3D line graph shows the relation between the assembly time of automobile engine and the total cost involved in the assembly using the scale of moderately agile, agile, less agile, traditional, and less traditional manufacturing process.

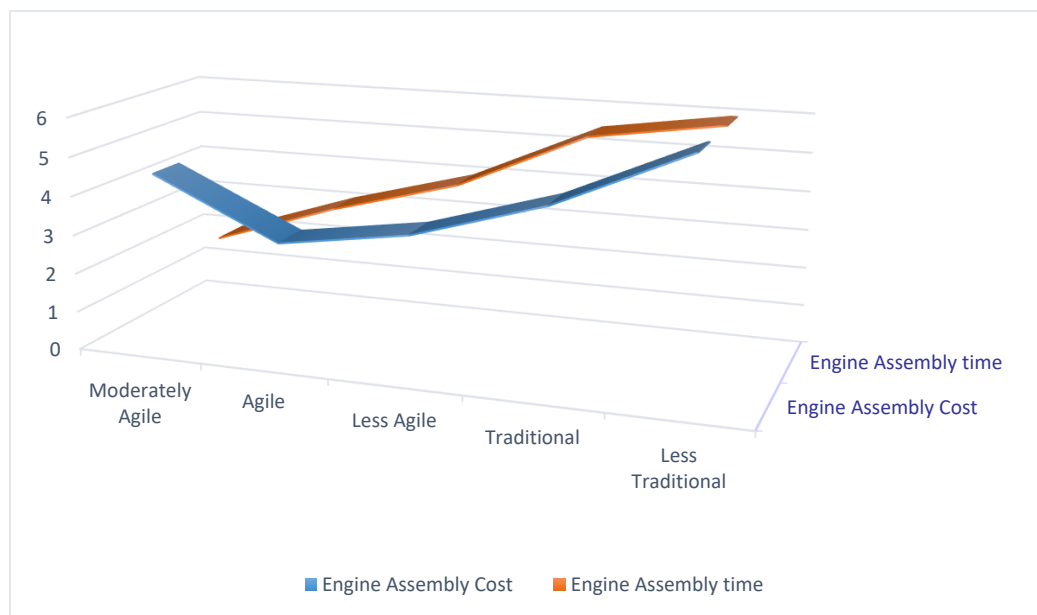


Figure 8. 3D Line Graph.

The results clearly show that the agile process involves less time and cost compared to the traditional assembly process.

5. Conclusions and Future Work

The previous manufacturing and traceability studies consist of source code trace, which is better merely for software-based systems where the primary artifact is the code. Using the proposed framework, the product at each manufacturing stage can be refined through in-phase and cross-phase iterations. Moreover, such iterations during manufacturing stages reduce cost and time in the production of vehicles in the automotive sector. Any changes or revisions are properly recorded using the trace matrix. These results show the validity of the proposed framework concerning prototyping, customer feedback, traceability, cost, and time, which are the main agile features all through automotive manufacturing. The limitation lies in initiating the application of agility to this industry and driving the researchers to develop more specific agile models. In the future, we can extend this research by manipulating the results of applying agile principles to the automotive industry and recording improvements in the manufacturing processes.

Author Contributions: Conceptualization, G.J., I.U.D. and A.A.; methodology, I.U.D. and A.A.; software, G.J.; validation, A.A.; formal analysis, I.U.D. and H.A.; investigation, A.A.; resources, A.A.; data curation, H.A.; writing—original draft preparation, G.J.; writing—review and editing, I.U.D. and A.A.; visualization, H.A.; supervision, I.U.D.; project administration, A.A.; funding acquisition, A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by the Deputyship for Research & Innovation, “Ministry of Education” in Saudi Arabia, project number IFKSURG-1437-035.

Acknowledgments: The authors extend their appreciation to the Deputyship for Research & Innovation, “Ministry of Education” in Saudi Arabia for funding this research work through the project number IFKSURG-1437-035.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Huikkola, T.; Kohtamäki, M. Agile new solution development in manufacturing companies. *Technol. Innov. Manag. Rev.* **2020**, *10*, 16–23. [[CrossRef](#)]
2. Gunasekaran, A.; Yusuf, Y.Y.; Adeleye, E.O.; Papadopoulos, T.; Kovvuri, D.; Geyi, D.G. Agile manufacturing: An evolutionary review of practices. *Int. J. Prod. Res.* **2019**, *57*, 5154–5174. [[CrossRef](#)]
3. Gunasekaran, A.; Yusuf, Y.Y.; Adeleye, E.O.; Papadopoulos, T. Agile manufacturing practices: The role of big data and business analytics with multiple case studies. *Int. J. Prod. Res.* **2018**, *56*, 385–397. [[CrossRef](#)]
4. Kusiak, A. Smart manufacturing. *Int. J. Prod. Res.* **2018**, *56*, 508–517. [[CrossRef](#)]
5. Song, W. Requirement management for product-service systems: Status review and future trends. *Comput. Ind.* **2017**, *85*, 11–22. [[CrossRef](#)]
6. Niazi, M.; Ali, M. Identifying high perceived value practices of CMMI level 2: An empirical study. *Inf. Softw. Technol.* **2009**, *51*, 1231–1243. [[CrossRef](#)]
7. Buglione, L.; Gresse, C.; Hauck, J.C. The LEGO Maturity & Capability Model Approach Luigi Buglione. In Proceedings of the 5th World Congress for Software Quality, Shanghai, China, November 2011; pp. 1–8.
8. Mittal, S.; Khan, M.A.; Romero, D.; Wuest, T. Smart manufacturing: Characteristics, technologies and enabling factors. *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.* **2019**, *233*, 1342–1361. [[CrossRef](#)]
9. Tuptuk, N.; Hailes, S. Security of smart manufacturing systems. *J. Manuf. Syst.* **2018**, *47*, 93–106. [[CrossRef](#)]
10. Keshta, I.; Niazi, M.; Alshayeb, M. Towards Implementation of Requirements Management Specific Practices (SP1.3 and SP1.4) for Saudi Arabian Small and Medium Sized Software Development Organizations. *IEEE Access* **2017**, *5*, 24162–24183. [[CrossRef](#)]
11. Maro, S.; Steghofer, J.P.; Steghöfer, J.P. Capra: A Configurable and Extendable Traceability Management Tool. In Proceedings of the 2016 IEEE 24th International Requirements Engineering Conference, RE 2016, Beijing, China, 12–16 September 2016; pp. 407–408. [[CrossRef](#)]
12. Bokhari, M.U.; Siddiqui, S.T. Metrics for Requirements Engineering and Automated Requirements Tools. In Proceedings of the 5th National Conference, Galway, Ireland, 9–10 June 2011.
13. Tao, F.; Qi, Q.; Liu, A.; Kusiak, A. Data-driven smart manufacturing. *J. Manuf. Syst.* **2018**, *48*, 157–169. [[CrossRef](#)]
14. Wang, J.; Ma, Y.; Zhang, L.; Gao, R.X.; Wu, D. Deep learning for smart manufacturing: Methods and applications. *J. Manuf. Syst.* **2018**, *48*, 144–156. [[CrossRef](#)]
15. Kang, H.S.; Lee, J.Y.; Choi, S.; Kim, H.; Park, J.H.; Son, J.Y.; Kim, B.H.; Noh, S.D. Smart manufacturing: Past research, present findings, and future directions. *Int. J. Precis. Eng. Manuf. Green Technol.* **2016**, *3*, 111–128. [[CrossRef](#)]
16. Souali, K.; Rahmaoui, O.; Ouzzif, M. An overview of traceability: Definitions and techniques. In Proceedings of the 2016 4th IEEE International Colloquium in Information Science and Technology, CIST, Tangier, Morocco, 24–26 October 2016.
17. Nassar, B.; Scandariato, R. Traceability Metrics as Early Predictors of Software Defects? In Proceedings of the 2017 IEEE International Conference on Software Architecture, ICSA 2017, Gothenburg, Sweden, 3–7 April 2017.
18. Kamalabalan, K.; Uruththirakodeeswaran, T.; Thiyagalilingam, G.; Wijesinghe, D.B.; Perera, I.; Meedeniya, D.; Balasubramani, D. Tool support for traceability of software artefacts. In Proceedings of the 2015 Moratuwa Engineering Research Conference, Moratuwa, Sri Lanka, 7–8 April 2015.
19. Li, Z.; Chen, M.; Huang, L.; Ng, V.; Geng, R. Tracing requirements in software design. In Proceedings of the 2017 International Conference on Software and System Process, ICSSP 2017, Paris, France, 5–7 July 2017.
20. Kleffmann, M.; Rohl, S.; Gruhn, V.; Book, M. Establishing and Navigating Trace Links between Elements of Informal Diagram Sketches. In Proceedings of the 2015 IEEE/ACM 8th International Symposium on Software and Systems Traceability, SST 2015, Florence, Italy, 17 May 2015.
21. Mäder, P.; Egyed, A. Do developers benefit from requirements traceability when evolving and maintaining a software system? *Empir. Softw. Eng.* **2015**, *20*, 413–441. [[CrossRef](#)]
22. Guo, J.; Cheng, J.; Cleland-Huang, J. Semantically Enhanced Software Traceability Using Deep Learning Techniques. In Proceedings of the 2017 IEEE/ACM 39th International Conference on Software Engineering, ICSE 2017, Buenos Aires, Argentina, 20–28 May 2017.
23. Maro, S.; Staron, M.; Steghöfer, J.P. Challenges of establishing traceability in the automotive domain. In *Lecture Notes in Business Information Processing*; Springer: Cham, Switzerland, 2017; Volume 269.

24. Gayer, S.; Herrmann, A.; Keuler, T.; Riebisch, M.; Antonino, P.O. Lightweight Traceability for the Agile Architect. *Computer* **2016**, *49*, 64–71. [[CrossRef](#)]
25. Sango, M. *A Component-Based Model-Driven Approach with Traceability of Concerns: Railway RBC Handover Case Study, Introduction in YRS 2015, Young Researchers Seminar*; ECTRI: Rome, Italy, 2015; pp. 1–15.
26. Mahmoud, A.; Niu, N. Supporting requirements to code traceability through refactoring. *Requir. Eng.* **2014**, *19*, 309–329. [[CrossRef](#)]
27. Aizenbud-Reshef, N.; Nolan, B.T.; Rubin, J.; Shaham-Gafni, Y. Model traceability. *IBM Syst. J.* **2006**, *45*, 515–526. [[CrossRef](#)]
28. Elkins, D.A.; Huang, N.; Alden, J.M. Agile manufacturing systems in the automotive industry. *Int. J. Prod. Econ.* **2004**, *91*, 201–214. [[CrossRef](#)]
29. Fritzsche, A. Implications of agile manufacturing in the automotive industry for order management in the factories-evidence from the practitioner’s perspective. *Procedia CIRP* **2018**, *72*, 369–374. [[CrossRef](#)]
30. Potdar, P.K.; Routroy, S. Analysis of Agile Manufacturing Enablers: A Case Study. *Mater. Today Proc.* **2018**, *5*, 4008–4015. [[CrossRef](#)]
31. Qasaimeh, M.; Abran, A. An Audit Model for ISO 9001 Traceability Requirements in Agile-XP Environments. *J. Softw.* **2013**, *8*, 1556–1567. [[CrossRef](#)]
32. Laghoulouata, Y.; Anwar, A.; Nassar, M.; Coulette, B. A dedicated approach for model composition traceability. *Inf. Softw. Technol.* **2017**, *91*, 142–159. [[CrossRef](#)]
33. Paige, R.F.; Matragkas, N.; Rose, L.M. Evolving models in Model-Driven Engineering: State-of-the-art and future challenges. *J. Syst. Softw.* **2016**, *111*, 272–280. [[CrossRef](#)]
34. Chhabra, J.K. Requirements Traceability through Information Retrieval Using Dynamic Integration of Structural and Co-change Coupling. In Proceedings of the International Conference on Advanced Informatics for Computing Research, Jalandhar, India, 17–18 March 2017; pp. 107–118.
35. Cleland, J.; Berenbach, B.; Clark, S. Best Practices for Automated Traceability. *Computer* **2007**, *40*, 27–35. [[CrossRef](#)]
36. Regan, G.; McCaffery, F.; McDaid, K.; Flood, D. The barriers to traceability and their potential solutions: Towards a reference framework. In Proceedings of the 38th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2012, Cesme, Turkey, 5–8 September 2012.
37. Torkar, R.; Gorschek, T.; Feldt, R.; Svahnberg, M.; Raja, U.A.; Kamran, K. Requirements traceability: A systematic review and industry case study. *Int. J. Softw. Eng. Knowl. Eng.* **2012**, *22*, 385–433. [[CrossRef](#)]
38. Maro, S.; Steghöfer, J.P.; Staron, M. Software traceability in the automotive domain: Challenges and solutions. *J. Syst. Softw.* **2018**, *141*, 85–110. [[CrossRef](#)]
39. Wohlrab, R.; Steghöfer, J.P.; Knauss, E.; Maro, S.; Anjorin, A. Collaborative Traceability Management: Challenges and Opportunities. In Proceedings of the 2016 IEEE 24th International Requirements Engineering Conference, RE 2016, Beijing, China, 12–16 September 2016; pp. 216–225.
40. Frederick, S.; Beier, G.; Figge, A.; Stark, R. Advanced Engineering Informatics Traceability in Systems Engineering—Review of industrial practices, state-of-the-art technologies and new research solutions. *Adv. Eng. Inform.* **2012**, *26*, 924–940.
41. Gotel, O.; Cleland-Huang, J.; Hayes, J.H.; Zisman, A.; Egyed, A.; Grünbacher, P.; Antoniol, G. The quest for Ubiquity: A roadmap for software and systems traceability research. In Proceedings of the 20th IEEE International Requirements Engineering Conference, RE 2012, Chicago, IL, USA, 24–28 September 2012; pp. 71–80.
42. Wen, L.; Tuffley, D.; Dromey, R.G. Formalizing the transition from requirements’ change to design change using an evolutionary traceability model. *Innov. Syst. Softw. Eng.* **2014**, *10*, 181–202. [[CrossRef](#)]
43. Ståhl, D.; Hallén, K.; Bosch, J. Achieving traceability in large scale continuous integration and delivery deployment, usage and validation of the eiffel framework. *Empir. Softw. Eng.* **2017**, *22*, 967–995. [[CrossRef](#)]
44. Kirova, V.; Kirby, N.; Kothari, D.; Childress, G. Effective Requirements Traceability: Models, Tools, and Practices. *Bell Labs Tech. J.* **2008**, *12*, 143–157. [[CrossRef](#)]
45. Tsuchiya, R.; Washizaki, H.; Fukazawa, Y. Interactive Recovery of Requirements Traceability Links Using User Feedback. In Proceedings of the International Conference on Advanced Information Systems Engineering, Stockholm, Sweden, 8–12 June 2015; Springer: Cham, Switzerland, 2015; Volume 9097, pp. 247–262.

46. Rempel, P.; Patrick, M.; Kuschke, T.; Philippow, I. Requirements Traceability across Organizational Boundaries—A Survey and Taxonomy. In Proceedings of the International Working Conference on Requirements Engineering: Foundation for Software Quality, Essen, Germany, 8–11 April 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 125–126.
47. Seibel, A.; Neumann, S.; Giese, H. Dynamic hierarchical mega models: Comprehensive traceability and its efficient maintenance. *Softw. Syst. Model.* **2010**, *9*, 493–528. [[CrossRef](#)]
48. Winkler, S.; von Pilgrim, J. A survey of traceability in requirements engineering and model-driven development. *Softw. Syst. Model.* **2010**, *9*, 529–565. [[CrossRef](#)]
49. Rempel, P.; Mader, P. Preventing Defects: The Impact of Requirements Traceability Completeness on Software Quality. *IEEE Trans. Softw. Eng.* **2017**, *43*, 777–797. [[CrossRef](#)]
50. Samalikova, J.; Kusters, R.J.; Trienekens, J.J.M.; Weijters, A.J.M.M. Process mining support for Capability Maturity Model Integration- based software process assessment, in principle and in practice. *J. Softw. Evol. Process* **2014**, *26*, 714–728. [[CrossRef](#)]
51. Soltan, H.; Mostafa, S. Lean and Agile Performance Framework for Manufacturing Enterprises. *Procedia Manuf.* **2015**, *2*, 476–484. [[CrossRef](#)]
52. Zanatta, A.; Vilain, P. Extending an Agile Method to Support Requirements Management and Development in Conformance to CMMI. *Hifen Uruguaiiana V* **2006**, *30*, 25–31.
53. Tariq, A.; Iftikhar, S. Remapping of CMMI Level-2 KPA's for Development Process Improvement of Software-as-a Service (SaaS) Cloud Environment. In Proceedings of the International IEEE Conference on Open Source Systems & Technologies, Lahore, Pakistan, 18–20 December 2014.
54. Garzas, J.; Paulk, M.C. A case study of software process improvement with CMMI-DEV and Scrum in Spanish companies. *J. Softw. Evol. Process* **2013**, *25*, 1325–1333. [[CrossRef](#)]

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).