DOWNLOAD PDF DIFFERENCE BETWEEN WATERFALL MODEL AND ITERATIVE MODEL

Chapter 1 : methodology - Agile Vs Spiral Model for SDLC - Stack Overflow

Answer / parmod. WaterFall Model: This is a flow based model, in which u pass every phase once, and can not go back to that phase again. That is why, it is used rarely now a days.

There are a number of activities done in a sequential order to achieve the end product. Each phase is associated with a deliverable that acts as an input to the subsequent phase of SDLC. Let us take a look at the different phases of the SDLC model: Requirement â€" This phase is the most important one for stakeholders and managers. These requirements determine the users of the system, their main functionalities, the inputs and outputs of the system. The output of this entire process is a functional specification document that explains the system as a whole. Design â€" The input to this phase is the functional specification document from the requirement phase. This phase details out the look of the system. The main output from this stage is software design and the decision of hardware and software requirements. The developers are the main people at work in this phase. In certain SDLC models, the testing and design phase overlap with the implementation phase. Testing â€" This includes both unit as well as system testing. The purpose of testing is to check whether the code has been able to achieve the required functionality as defined in the requirement phase or not. Some of the most popular SDLC models are: It is a classic approach to software development that follows a linear and sequential method to deliver software product. This model has different deliverables from each phase. This model offers the following benefits: It is simple and easy to implement. Since the model follows a linear approach, it becomes easier to manage. Each phase is executed one at a time. This is best utilized for small-sized projects. With advantages come certain disadvantages. Some of them are discussed below: There is a high-risk factor involved. It is not beneficial for big projects. It cannot be used for projects where requirements can change. It is not suited for projects that are complex or that employ OOPS concepts. Output from one phase of SDLC acts as input to the next phase. Requirements are converted into design. Design determines the code that needs to be written to implement it. Testing verifies whether the code satisfies the design and requirement. The main phases of SDLC are: In a waterfall model, one of the most popular SDLC models, each step follows in a sequential manner without overlapping or iterative steps. If you like this article or our site. Please spread the word.

Chapter 2 : An Iterative Waterfall Isnâ€™t Agile

Waterfall Model is a flow based model, in which we pass every phase once, and can not go back to that phase again. Its most eminent drawback is that if there is any change in requirements, we cannot make any changes to the requirement section. Iterative Model is somewhat similar to waterfall model.

This debate revolves around the best software model that developers can make use of. There are various phases that are involved in the software development process. The phases are similar in both waterfall and the V model, and the only thing that has so far been contentious is the approach to which these two models can be achieved by. In the V model, there is a lot of activities which, when plotted together on a schematic diagram, form a V shape. Each phase that is said has a corresponding phase that is involved with testing. This model due to the equal number of testing and development is referred to as the verification and validation model. The verification side deals with the development end while validation deals with the testing phases. Among the activities that verification falls under include requirement analysis where information is gathered from the end user. This information is important in the development of the software documentation. Next up is the system design, which aims to prepare functional design of software. The next thing that follows in line is the architectural design. This is also called the high level design that the interface relationship and the database tables and the dependencies of the tables. The final stage in the development process is coding where the entire project is broken down into small sections for coding which are then merged to create the entire system. The validation side, on the other side, has four stages as in the verification stage. These phases begin with the unit testing, then integration testing, system testing and finally the user acceptance testing where the entire system is evaluated as a whole. The waterfall model is the earliest software development procedure, with its origin coming from the manufacturing and construction industries. The basic concept of this process is that there is a sequential flow of processes which scale down one after the other, as seen in a waterfall. This step leads to the design phase, where most of the software is created and then the implementation phase where the software code is written. The phase that follows is the testing and debugging, leading to the delivery and finally the maintenance phase. The main difference noted between the two models is that the testing activities are performed after the development is over. The V model seems to look like a model that has a given start and end whereas the waterfall model is continuously iterative. The V model differs by being a simultaneous process. From the different software that has been produced in the market, software produced using the V process seem to be lower, as there are numerous testing activities as opposed to the waterfall model that has a single testing phase when the project is complete. It can therefore be said that the use of the V model is preferred whenever there are continuous changes that need to be included. This is for a person or a development that has the client unsettled about the needs of their project, as they keep changing what it is they perceive to seem as ideal. People with fixed requirements that will not change in the development phase of the project should settle for the waterfall model. It is also important to note that changes in the V model are cheap to implement as testing is and development are done simultaneously. This is not the case with the waterfall model, which tends to be a costly affair, as any software defects cannot be noticed until it gets to the testing phase. If you like this article or our site. Please spread the word.

Chapter 3 : Difference Between SDLC and Waterfall Model | Difference Between

In an iterative waterfall model, you might use the iterations more for refinement/elaboration of elements of the overall design in multiple phases of [detailed design / build / test]. In theory, you could plan most of the iterations in advance.

To do this of course is sin since it removes all possibility of review or feedback from previous iterations into the next. Yet I have seen it applied after which their protractors proclaimed high and loud they were now Agile for doing so!! Another definition of iterative waterfall is to have successive waterfalls where the next feeds from the previous. Again no feedback during the cycle itself from design to delivery but only between delivery and design is feedback allowed. This definition seems to match the link provided by DPD. My personal definition of "Waterfall" style and "Agile" style processes lies in how the project is viewed and broken down. Both pretty much cover the same phases and though processes but tackle them in a much different way. If you view the production of a software as a stack with Inception at the top and deployment and support at the bottom. Waterfall will break down the software creation process horizontally and traverse each layer in sequence. For a waterfallist if such thing can be said proceeding to the next layer is impossible without having completely worked out the layer above, its just inconceivable. Typical argument would be that you never build the roof of a house before the foundation and will say things akin to "measure twice but cut once". Agile on the other hand will break down the creation process vertically producing slices of functionality that get added to the product over time. They are able to do this because they have willingly reduced the scope of the target result. To them it is inconceivable to proceed without testing first to see if the idea makes sense. With each iterations they add slivers of functionality and test it against the idea and against the rest of the system. Waterfall is rooted in engineering where the target is a tangible real life product and error is not admissible. They see software as rigid pieces of steel and concrete that must be assembled into a structure. Therefore they will tend to be over cautious planing everything minutely wasting resources on the altar of perfection. For Waterfall to work you need very strong experienced players and a complete trust in them, you need to be extra meticulous at every step of the way in foreseeing what could be and anticipating. Agile is rooted in the scientific process where knowledge is gathered one bit at a time and tested continuously. They understand perfectly that software is a malleable and can be bent and shaped at will. They treat ideas as would a researcher treat a theory, that is it stands as long as it agrees with the tests. Therefore they will tend to continuously reshape their system wasting resources by recreating the same things over and over each time with a slight twist to fit the new variables. For agile to work you need to continuously evaluate your system as a whole and place it against the new reality. Measurements and automation are essential in order to gain the most accurate picture possible of the present such that the next step can be taken as a corrective measure towards the ideal path. Even the ideal path needs to be re-evaluated to make sure that it does indeed points towards the ideal target. Waterfall Build software to last as-is, static, set in concrete. Agile grow software, care and tend for it to an ever changing world. True that software is malleable early on but as it gains size and age it also gains stiffness resisting changes more and more. In my opinion there is no silver bullet and there tends to be to much religious devotion at the detriment of rational reflection. That said of the two extremes Agile seems to be the one that takes into account the soft of software.

Chapter 4 : Iterative and incremental development - Wikipedia

One of the major differences between Agile and Waterfall development methodology is their individual approach towards quality and testing. In the Waterfall model, the "Testing" phase comes after the "Build" phase, but, in the Agile methodology, testing is typically performed concurrently with programming or at least in the same.

Waterfall Model followed in the sequential order, and so project development team only moves to next phase of development or testing if the previous step completed successfully. What is the Agile methodology? Agile methodology is a practice that helps continuous iteration of development and testing in the software development process. In this model, development and testing activities are concurrent, unlike the Waterfall model. This process allows more communication between customers, developers, managers, and testers. Advantages of Waterfall Model: It is one the easiest model to manage. Because of its nature, each phase has specific deliverables and a review process. It works well for smaller size projects where requirements are easily understandable. Faster delivery of the project Process and results are well documented. Easily adaptable method for shifting teams This project management methodology is beneficial to manage dependencies. Advantages of the Agile Model: It is focused client process. So, it makes sure that the client is continuously involved during every stage. Agile teams are extremely motivated and self-organized so it likely to provide a better result from the development projects. Agile software development method assures that quality of the development is maintained The process is completely based on the incremental progress. Therefore, the client and team know exactly what is complete and what is not. This reduces risk in the development process. Limitations of Waterfall Model: It is not an ideal model for a large size project If the requirement is not clear at the beginning, it is a less effective method. Very difficult to move back to makes changes in the previous phases. The testing process starts once development is over. Hence, it has high chances of bugs to be found later in development where they are expensive to fix. Limitations of Agile Model It is not useful method for small development projects. It requires an expert to take important decisions in the meeting. Cost of implementing an agile method is little more compared to other development methodologies. Difference between Agile and Waterfall Model: Agile It separates the project development lifecycle into sprints. Software development process is divided into distinct phases. It follows an incremental approach Waterfall methodology is a sequential design process. Agile methodology is known for its flexibility. Waterfall is a structured software development methodology so most times it can be quite rigid. Agile can be considered as a collection of many different projects. Software development will be completed as one single project. Agile is quite a flexible method which allows changes to be made in the project development requirements even if the initial planning has been completed. There is no scope of changing the requirements once the project development starts. Agile methodology, follow an iterative development approach because of this planning, development, prototyping and other software development phases may appear more than once. All the project development phases like designing, development, testing, etc. Test plan is reviewed after each sprint The test plan is rarely discussed during the test phase. Agile development is a process in which the requirements are expected to change and evolve. The method is ideal for projects which have definite requirements and changes not at all expected. In Agile methodology, testing is performed concurrently with software development. This model shows a project mindset and places its focus completely on accomplishing the project. It may increase stress in fixed-price scenarios. Reduces risk in the firm fixed price contracts by getting risk agreement at the beginning of the process. Prefers small but dedicated teams with a high degree of coordination and synchronization. Products owner with team prepares requirements just about every day during a project. Business analysis prepares requirements before the beginning of the project. Test team can take part in the requirements change without problems. It is difficult for the test to initiate any change in requirements. Description of project details can be altered anytime during the SDLC process. Detail description needs to implement waterfall software development approach. The Agile Team members are interchangeable, as a result, they work faster. There is also no need for project managers because the projects are managed by the entire team In the waterfall method, the process is always straightforward so, project manager plays an essential role during every stage of SDLC. Agile and Waterfall are very different software development methodologies and are good in their respective way. However, there are certain major differences highlighted below - Waterfall model is ideal for projects which have defined requirements, and no changes are expected. On the other hand, Agile is best suited where there is a higher chance of frequent requirement changes. The waterfall is easy to manage, sequential, and rigid method. Agile is very flexible and it possible to make changes in any phase. In Agile process, requirements can change frequently. However, in a waterfall model, it is defined only once by the business analyst.

Chapter 5 : Agile & Waterfall Methodologies â€“ A Side-By-Side Comparison | IT Staffing & Technical Cons

In iterative Model, a small part of our system is implemented and reviewed further for requirements and the process is further repeated till the final product is obtained whereas the spiral model which is a combination of iterative and waterfall model uses a different kind of approach in which the project passes through 4 phases again and again.

Evaluation Phase Advantages The disadvantage of the waterfall model is the advantage of the spiral model. It is a realistic model, which is often used in the development of large software. There is a systematic approach used in the spiral model, which is integrated into the iterative framework. This helps in ensuring there is no problems in the software. Since changes to the software can be made at any point of time in the software development process. Disadvantages In the spiral model , it is important to have a member in the team, who is an expert at risk assessment. Without correct risk assessment, there are chances that the software can utterly be a failure. The client may have to spend a lot of time with the development team to fix the issues that have cropped up in the software. This may at times become like a overhead for the both the client as well as the software development company. This also leads to the over involvement of the customer in the process of software development, which may cause the customer to ignore certain risks and cause harm to the project. Difference Between Waterfall Model and Spiral Model While in the spiral model, the customer is made aware of all the happenings in the software development, in the waterfall model the customer is not involved. This often leads to situations, where the software is not developed according to the needs of the customer. In the spiral model, the customer is involved in the software development process from the word go. This helps in ensuring that the software meets the needs of the customer. In the waterfall model, when the development process shifts to the next stage, there is no going back. This often leads to roadblocks, especially during the coding phase. Many times it is seen that the design of the software looks feasible on paper, however, in the implementation phase it may be difficult to code for the same. However, in the spiral model, since there are different iterations, it is rather easier to change the design and make the software feasible. In the spiral model, one can revisit the different phases of software development, as many times as one wants, during the entire development process. This also helps in back tracking, reversing or revising the process. However, the same is not possible in the waterfall model, which allows no such scope. Often people have the waterfall model or spiral model confusion due to the fact, that the spiral model seems to be a complex model. It can be attributed to the fact that there are many iterations, which go into the model. At the same time, often there is no documentation involved in the spiral model, which makes it difficult to keep a track of the entire process. On the other hand, the waterfall model has sequential progression, along with clear documentation of the entire process. This ensures one has a better hold over the entire process. From the above discussion on spiral model vs waterfall model, it is clear that both the models have their own advantages and shortcomings. While one is stuck between the waterfall model vs spiral model debate, it is best to evaluate the software that is being developed and then decide the right approach. The size of the project and the urgency of the software will have to be taken into consideration for the same. At the same time, the resources available will have an important role to play in the software development process.

Chapter 6 : Difference Between Vmodel and Waterfall Model | Difference Between

Difference between Waterfall and Incremental Model. Different models have been purposed by project managers for the development of software. The choice is made from the purposed model according to the requirements of the software.

An iterative waterfall looks something like this: In one sprint, someone perhaps a business analyst working with a product owner figures out what is to be built. But rather than treating the user stories as short placeholders for future conversations, each user story becomes a mini-specification document, perhaps three to five pages long. Because this takes a full sprint to figure out and document, a second sprint is devoted to designing the user interface for the user story. Sometimes, the team tries to be a little more agile in their minds by starting the design work just a little before the mini-spec for a user story is fully written. Programmers are then handed a pair of documents. No programming can start until these two artifacts are ready. They take an attitude of saying they will build whatever is asked for, but you better tell them exactly what is needed at the start of the sprint. Some organizations then stretch things out even further by having the testers work an iteration behind the programmers. Fortunately, most teams realize that programmers and testers need to work together in the same iteration, but not extend that to being a whole team working together. This leads to the process shown in this figure. This figure shows a first iteration devoted to analysis. A second iteration possibly slightly overlapping with the first is devoted to user experience design. And then a third iteration is devoted to coding and testing. This is not agile. What we see in this figure is an iterative waterfall. In traditional, full waterfall development, a team does all of the analysis for the entire project first. Then they do all the design for the entire project. Then they do all the coding for the entire project. Then they do all the testing for the entire project. In the iterative waterfall of the figure above, the team is doing the same thing but they are treating each story as a miniature project. They do all the analysis for one story, then all the design for one story, then all the coding and testing for one story. This is an iterative waterfall process, not an agile process. Ideally, in an agile process, all types of work would finish at exactly the same time. The team would finish analyzing the problem at exactly the same time they finished designing the solution to the problem, which would also be the same time they finished coding and testing that solution. It can be achieved some times. But it can remain the goal a team can work towards. A team should always work to overlap work as much as possible. And upfront thinking analysis, design and other types of work should be done as late as possible and in as little detail as possible while still allowing the work to be completed within the iteration. If you are treating your user stories as miniature specification documents, stop. Start instead thinking about each as a promise to have a conversation. Feel free to add notes to some stories about things you want to make sure you bring up during that conversation. But adding these notes should be an optional step, not a mandatory step in a sequential process. Leaving them optional avoids turning the process into an iterative waterfall process and keeps your process agile. Download Scrum Master Guide.

Chapter 7 : What is the difference between waterfall and iterative development model? - calendrierdelascie

8 Agile vs Iterative vs Waterfall - {Process} Waterfall Iterative (hybrid) Agile Quality Quality focus changes from Analysis > Design > Code > Test Quality focus shifts between Analysis/Design phase to Coding/Testing phase Quality focus on all aspects of SDLC at any given time.

There are several ways to develop software, two of the most prominent methods being waterfall and Agile. And as anytime there are two ways to go about something, a debate rages about which is best. Does it matter really? What is the waterfall methodology? Much like construction and manufacturing workflows, waterfall methodology is a sequential design process. This means that as each of the eight stages conception, initiation, analysis, design, construction, testing, implementation, and maintenance are completed, the developers move on to the next step. Advantages of the Waterfall Methodology 1. The waterfall methodology stresses meticulous record keeping. Having such records allows for the ability to improve upon the existing program in the future. With the waterfall methodology, the client knows what to expect. Disadvantages of the Waterfall Methodology 1. Waterfall methodology relies heavily on initial requirements. However, if these requirements are faulty in any manner, the project is doomed. If a requirement error is found, or a change needs to be made, the project has to start from the beginning with all new code. The whole product is only tested at the end. If bugs are written early, but discovered late, their existence may have affected how other code was written. Additionally, the temptation to delay thorough testing is often very high, as these delays allow short-term wins of staying on-schedule. If the client realizes that they need more than they initially thought, and demand change, the project will come in late and impact budget. When should you use waterfall methodology? When there is a clear picture of what the final product should be. When definition, not speed, is key to success. Instead of a sequential design process, the Agile methodology follows an incremental approach. Developers start off with a simplistic project design, and then begin to work on small modules. The work on these modules is done in weekly or monthly sprints, and at the end of each sprint, project priorities are evaluated and tests are run. These sprints allow for bugs to be discovered, and customer feedback to be incorporated into the design before the next sprint is run. The process, with its lack of initial design and steps, is often criticized for its collaborative nature that focuses on principles rather than process. Advantages of the Agile Methodology 1. The Agile methodology allows for changes to be made after the initial planning. Re-writes to the the program, as the client decides to make changes, are expected. At the end of each sprint, project priorities are evaluated. This allows clients to add their feedback so that they ultimately get the product they desire. The testing at the end of each sprint ensures that the bugs are caught and taken care of in the development cycle. Because the products are tested so thoroughly with Agile, the product could be launched at the end of any cycle. Disadvantages of Agile Methodology 1. With a less successful project manager, the project can become a series of code sprints. If this happens, the project is likely to come in late and over budget. When should you use Agile methodology? When rapid production is more important than the quality of the product. When clients will be able to change the scope of the project. When you have skilled developers who are adaptable and able to think independently. When the product is intended for an industry with rapidly changing standards. Both the Agile and waterfall methodologies have their strengths and weaknesses. The key to deciding which is right for you comes down to the context of the project. Is it going to be changing rapidly? If so, choose Agile. Do you know exactly what you need? Then maybe waterfall is the better option. Consider taking aspects of both methodologies and combining them in order to make the best possible software development process for your project. Do you prefer the Agile or waterfall methodology? Have you ever tried combining the two? How did that work out? Let us know in the comments section, or join the conversation on Facebook , Twitter , or LinkedIn. Thank to ihasb33r for the use of their respective photographs.

DOWNLOAD PDF DIFFERENCE BETWEEN WATERFALL MODEL AND ITERATIVE MODEL

Chapter 8 : Waterfall vs. Incremental vs. Spiral vs. Rad Model: Key Difference

To manage the level of complexity during the software development cycle, various SDLC model is implemented by the software companies. Their aim is to deliver a quality software product, however each model is unique in terms of their software development approach. To effectively learn the SDLC models.

Published by Glen Lipka on June 16, Recently, I was in some conversations about development practices. I believe the methodology used in a technology company has a huge impact on the quality of the product. I drew up a little diagram to show the differences under ideal conditions. To be as fair as possible, I had the product launch on the exact same date. This is not a look at specific coding practices, but rather how the product is defined and built. The main difference is that in the waterfall approach the specification is heavyweight and goes through as many details as possible. By doing this, the development effort can be reduced to only 10 units. This is because all of the decisions are made up front. QA and Acceptance takes a little longer because there is a dark period where the engineers are working and you have to validate that they actually built to spec. In the iterative model, there is no dark period. It forces the product manager to work with engineers the whole time. It forces the engineers to make decisions in the middle of development, which means they will have to redo some of the work. This is why the development is 14 units rather than The key question is: Which product will be better? My strong opinion on this is that the latter model will yield a vastly superior product. The reason is that it is impossible to make good decisions up front. You have to see it all in action. You have to touch the product and use it to make good decisions. Plus, no one is perfect. We all make bad assumptions, bad decisions and use faulty logic. Being able to change your mind helps you avoid painting yourself in a corner with bad decisions early on. Additionally, the interaction between engineers and product managers often yields a much more creative outcome. Engineers are creative people if given a chance. It allows you to refactor code and learn from mistakes. At the end of the process, everyone has been part of the process so acceptance is de-facto finished and only QA remains. I strongly suggest you examine your practices and see if you are doing the waterfall approach. Do you really think that heavy specification is the right use of your product managers time?

DOWNLOAD PDF DIFFERENCE BETWEEN WATERFALL MODEL AND ITERATIVE MODEL

Chapter 9 : SDLC Models Explained: Agile, Waterfall, V-Shaped, Iterative, Spiral | Existek Blog

Difference Between Waterfall Model and Spiral Model While in the spiral model, the customer is made aware of all the happenings in the software development, in the waterfall model the customer is not involved.

Overview[edit] Iterative development was created as a response to inefficiencies and problems found in the waterfall model. Learning comes from both the development and use of the system, where possible key steps in the process start with a simple implementation of a subset of the software requirements and iteratively enhance the evolving versions until the full system is implemented. At each iteration , design modifications are made and new functional capabilities are added. The procedure itself consists of the initialization step, the iteration step, and the Project Control List. The initialization step creates a base version of the system. The goal for this initial implementation is to create a product to which the user can react. It should offer a sampling of the key aspects of the problem and provide a solution that is simple enough to understand and implement easily. To guide the iteration process, a project control list is created that contains a record of all tasks that need to be performed. It includes items such as new features to be implemented and areas of redesign of the existing solution. The control list is constantly being revised as a result of the analysis phase. The iteration involves the redesign and implementation of iteration is to be simple, straightforward, and modular, supporting redesign at that stage or as a task added to the project control list. In a light-weight iterative project the code may represent the major source of documentation of the system; however, in a critical iterative project a formal Software Design Document may be used. The analysis of an iteration is based upon user feedback, and the program analysis facilities available. The project control list is modified in light of the analysis results. Phases[edit] Incremental development slices the system functionality into increments portions. In each increment, a slice of functionality is delivered through cross-discipline work, from the requirements to the deployment. Inception identifies project scope, requirements functional and non-functional and risks at a high level but in enough detail that work can be estimated. Elaboration delivers a working architecture that mitigates the top risks and fulfills the non-functional requirements. Construction incrementally fills-in the architecture with production-ready code produced from analysis, design, implementation, and testing of the functional requirements. Transition delivers the system into the production operating environment. Each of the phases may be divided into 1 or more iterations, which are usually time-boxed rather than feature-boxed. Architects and analysts work one iteration ahead of developers and testers to keep their work-product backlog full. The team applied IID in a series of 17 iterations over 31 months, averaging around eight weeks per iteration. The DoD Instruction There are two approaches, evolutionary and single step [waterfall], to full capability. An evolutionary approach is preferred. It can also be done in phases. Recent revisions to DoDI