Chapter 1 : Overview (Java SE 10 & JDK 10)

PROJECT DEVELOPMENT SPECIFICATION - INDUSTRY v Page 5 This PDS is intended to support the elements, procedures and documentation requirements presented.

Windows 10, version supports Bluetooth version 5. Camera Updates to Camera driver development include: This enables the platform to expose spherical frame sources for example, equirectangular frames , enabling apps to detect and handle video camera streams as well as to provide a capture experience. Display The following are updates to Display driver development in Windows 10, version GPU paravirtualization support - Enables display drivers to provide rendering capabilities to Hyper-V virtualized environments. Brightness - A new brightness interface to support multiple displays that can be set to calibrated nit-based brightness levels. Fixed function hardware is useful when GPU is already saturated in these scenarios and to enable parallel processing. This feature is optional and should only be implemented if fixed function hardware is available. This feature should not be implemented with 3D or Compute. D3D12 video decode now supports Decode Tier II, indicating driver supports Array of Textures that enable applications to amortize allocation cost and reduce peak memory usage during resolution change. Tiled resource tier and LDA atomics - A new cross node sharing tier to add support for atomic shader instructions working across linked adapter LDA nodes. GPU dithering support - Drivers can report the ability to perform dithering on the wire signal for a given timing mode. This allows the OS to explicitly request dithering in scenarios where a higher effective bit depth is needed than is physically available on the monitor link, for example for HDR10 over HDMI 2. Post-processing color enhancement override - Adds the ability for the OS to request that the driver temporarily disable any post-processing that enhances or alters display colors. This is to support scenarios where specific applications want to enforce colorimetrically accurate color behavior on the display, and safely coexist with OEM or IHV-proprietary display color enhancements. GPU performance data - Extensions to DdiQueryAdapterInfo will expose information such as temperature, fan speed, clock speeds for engines and memory, memory bandwidth, power draw, and voltages Miscellaneous - A new SupportContextlessPresent driver cap to help IHV onboard new driver. Driver will set this bit during adapter initialization if the adapter is hot-pluggable. Display Diagnostics - Kernel mode device driver interface DDI changes to allow the driver for a display controller to report diagnostic events to the OS. This provides a channel through which the driver can log events which would otherwise be invisible to the OS as the events are not a response to an OS request or something the OS needs to react to. Shared graphics power components - Allows non-graphics drivers to participate in the power management of a graphics device. A non-graphics driver will use a driver interface to manage one or more of these shared power components in coordination with the graphics driver. Shared texture improvements - Includes increasing the types of textures that can be shared across processes and D3D devices. This design enables the frame server OS component to support monochrome with minimal memory copying. Driver security Updates to Windows Driver Security Guidance and the Driver security checklist , which provides a driver security checklist for driver developers. Windows kernel This section describes the new and updated features for Windows kernel driver development in Windows 10, version Support was added to provide drivers with a sanctioned location that the operating system knows about where they can store file state. With this approach, the system can associate files in that location with a device or driver. There are distinct locations to store file states specific to the internals of a driver and specific to a device. For drivers that have file state, you can decide if the state written to disk is: NT services and kernel-mode and user-mode drivers can raise a custom trigger for a device by using the RtlRaiseCustomSystemEventTrigger function. A custom trigger, owned by the driver developer, notifies system event broker to start an associated background task with it, which is identified by a custom trigger identifier. You can now register for active session change notification and get a callback when the notification is fired. As part of this notification, some data is also shared with the caller. Networking This section outlines new features and improvements for Windows Networking driver development in Windows 10, version

Chapter 2 : vivo V5 - Full phone specifications

Initiate a specification development process in early to create Version 6. As qualification rates for Version 5 are likely to accelerate, a follow-on spec level needs to be developed.

Loose Ends Introduction Requirements and specifications are very important components in the development of any embedded system. While it is a common tendency for designers to be anxious about starting the design and implementation, discussing requirements with the customer is vital in the construction of safety-critical systems. For activities in this first stage has significant impact on the downstream results in the system life cycle. For example, errors developed during the requirements and specifications stage may lead to errors in the design stage. When this error is discovered, the engineers must revisit the requirements and specifications to fix the problem. This leads not only to more time wasted but also the possibility of other requirements and specifications errors. Many accidents are traced to requirements flaws, incomplete implementation of specifications, or wrong assumptions about the requirements. While these problems may be acceptable in non-safety-critical systems, safety-critical systems cannot tolerate errors due to requirements and specifications. Therefore, it is necessary that the requirements are specified correctly to generate clear and accurate specifications. There is a distinct difference between requirements and specifications. A requirement is a condition needed by a user to solve a problem or achieve an objective. A specification is a document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system, and often, the procedures for determining whether these provisions have been satisfied. For example, a requirement for a car could be that the maximum speed to be at least mph. The specification for this requirement would include technical information about specific design aspects. Another term that is commonly seen in books and papers is requirements specification which is a document that specifies the requirements for a system or component. It includes functional requirements, performance requirements, interface requirements, design requirements, and developement standards. So the requirements specification is simply the requirements written down on paper. Key Concepts Establishing Correct Requirements The first step toward developing accurate and complete specifications is to establish correct requirements. As easy as this sounds, establishing correct requirements is extremely difficult and is more of an art than a science. There are different steps one can take toward establishing correct requirements. Although some of the suggestions sound fairly obvious, actually puttting them into practice may not be as easy as it sounds. The first step is to negotiate a common understanding. There is no point in trying to establish exact specifications if the designers and customers cannot even agree on what the requirements are. Problem stems from ambiguities in stating requirements. Possible interpretations of this requirement includes building a bus, train, or airplane, among other possibilities. Although each of these transportation devices satisfy the requirement, they are certainly very different. Ambiguous requirements can be caused by missing requirements, ambiguous words, or introduced elements. The above requirement does not state how fast the people should be transported from Boston to Washington D. Taking an airplane would certainly be faster than riding a bus or train. These are also missing requirements. What exactly does "group" imply? A group can consist of 5 people, people, people, etc. The requirement states to "create a means" and not "design a transportation device". This is an example of introduced elements where an incorrect meaning slipped into the discussion. It is important to eliminate or at least reduce ambiguities as early as possible because the cost of them increases as we progress in the development life cycle. Often the problem one has in establishing correct requirements is how to get started. One of the most important things in getting started is to ask questions. Context-free questions are high-level questions that are posed early in a project to obtain information about global properties of the design problem and potential solutions. Examples of context-free questions include who is the client? These questions force both sides, designer and customer, to look at the higher issues. Also, since these questions are appropriate for any project, they can be prepared in advance. Another important point is to get the right people involved. There is no point in discussing requirements if the appropriate people are not involved in the discussion. Related to getting the right people involved is making meetings work. Having effective meetings is not as easy as it sounds. However, since they play a central role in establishing requirements it is essential to know how to make meetings work. There are important points to keep in mind when creating effective meetings, which include creating a culture of safety for all participants, keeping the meeting to an appropriate size, and other points. Ideas are essential in establishing correct requirements, so it is important that people can get together and generate ideas. Every project will also encounter conflicts. Conflicts can occur from personality clashes, people that cannot get along, intergroup prejudice such as those between technical people and marketing people, and level differences. It is important that a facilitator is present to help resolve conflicts. In establishing requirements, it is important to specifically establish the functions, attributes, constraints, preferences, and expectations of the product. Usually in the process of gaining information, functions are the first ones to be defined. Functions describe what the product is going to accomplish. It is also important to determine the attributes of a product. Attributes are characteristics desired by the client, and while 2 products can have similar functions, they can have competely different attributes. After all the attributes have been clarified and attached to functions, we must determine the constraints on each of the atrributes. Preferences, which is a desirable but optional condition placed on an attribute, can also be defined in addition to its constraints. This will largely determine the success of the product. Testing is the final step on the road to establishing correct requirements. There are several testing methods used, as listed below. This involves asking questions such as how fast? Technical review - A testing tool for indicating the progress of the requirements work. It can be formal or informal and generally only deals with technical issues. Technical reviews are necessary because it is not possible to produce error-free requirements and usually it is difficult for the producers to see their own mistakes. User satisfaction test - A test used on a regular basis to determine if a customer will be satisifed with a product. Black box test cases - Constructed primarily to test the completeness, accuracy, clarity, and conciseness of the requirements. Existing products - Useful in determining the desirable and undesirable characteristics of a new product. At some point it is necessary to end the requirements process as the fear of ending can lead to an endless cycle. This does not mean that it is impossible to revisit the requirements at a later point in the development life cycle if necessary. However, it is important to end the process when all the necessary requirements have been determined, otherwise you will never proceed to the design cycle. Establishing good requirements requires people with both technical and communication skills. Technical skills are required as the embedded system will be highly complex and may require knowledge from different engineering disciplines such as electrical engineering and mechanical engineering. Communication skills are necessary as there is a lot of exchange of information between the customer and the designer. Without either of these two skills, the requirements will be unclear or inaccurate. It is essential that requirements in safety critical embedded systems are clear, accurate, and complete. The problem with requirements is that they are often weak about what a system should not do. In a dependable system, it is just as important to specify what a system is not suppose to do as to specfiy what a system is suppose to do. These systems have an even greater urgency that the requirements are complete because they will only be dependable if we know exactly what a system will do in a certain state and the actions that it should not perform. Requirements with no ambiguities will also make the system more dependable. Extra requirements will usually be required in developing a dependable embedded system. The universe can be considered a system, and so can an atom. A system is very loosely defined and can be considered as any of the following definitions. Systems engineering is not a technical specialty but is a process used in the evolution of systems from the point when a need is identified through production and construction to deployment of the system for consumer use. The development of embedded systems also requires the knowledge of different engineering disciplines and can follow the techniques used for systems engineering. Therefore, it is appropriate that the steps used in establishing system requirements also be applicable to requirements for embedded systems. The conceptual system design is the first stage in the systems design life cycle and an example of the systems definition requirements process is shown in Figure 1. Each individual box will be explain below. Example of system requirements definition process [Blanchard90] In establishing system requirements, the first step is to define a need. This need is based on a want or desire. Usually, an individual or organization identifies a need for an item or function and then a new or modified system is developed to fulfill the requirement. After a need is defined, feasibility studies should be conducted to evaluate various technical approaches that can be taken. The system operational requirements should also be defined. This includes the definition of system operating characteristics, maintenance support concept for the system, and identification of specific design criteria. In particular, the system operational requirements should include the following elements. Performance and physical parameters - Definition of the operating characteristics or functions of the system. Use requirements - Anticipation of the use of the system. Operational deployment or distribution - Identification of transportation and mobility requirements. Includes quantity of equipment, personnel, etc.

Chapter 3 : V-Model - Wikipedia

This feature is not available right now. Please try again later.

The testing stream generally consists of: Installation qualification IQ Operational qualification OQ Performance qualification PQ The development stream can consist depending on the system type and the development scope of customization, configuration or coding. Applications[edit] Off-Core alternatives illustrating upward and downward iterations and Time and Maturity dimension. Mooz [3] [7] The V-model is used to regulate the software development process within the German federal administration. Nowadays it is still the standard for German federal administration and defense projects, as well as software developers within the region. The concept of the V-model was developed simultaneously, but independently, in Germany and in the United States in the late s: It was taken over by the Federal Ministry of the Interior for the civilian public authorities domain in summer It was created to show the test and integration approach which was driven by new challenges to surface latent defects in the software. The need for this new level of latent defect detection was driven by the goal to start automating the thinking and planning processes of the air traffic controller as envisioned by the automated enroute air traffic control AERA program. The reason the V is so powerful comes from the Hughes culture of coupling all text and analysis to multi dimensional images. Its primary use is in project management [3] [4] and throughout the project lifecycle. One fundamental characteristic of the US V-model is that time and maturity move from left to right and one cannot move back in time. All iteration is along a vertical line to higher or lower levels in the system hierarchy, as shown in the figure. The expansion of the model to a dual-Vee concept is treated in reference. In project management it is a method comparable to PRINCE2 and describes methods for project management as well as methods for system development. The V-Model, while rigid in process, can be very flexible in application, especially as it pertains to the scope outside of the realm of the System Development Lifecycle normal parameters. Advantages[edit] These are the advantages V-model offers in front of other systems development models: The users of the V-model participate in the development and maintenance of the V-model. A change control board publicly maintains the V-Model. The change control board meets anywhere from every day to weekly and processes all change requests received during system development and test. The organization and execution of operation, maintenance, repair and disposal of the system are not covered by the V-model. However, planning and preparation of a concept for these tasks are regulated in the V-model. The V-model addresses software development within a project rather than a whole organization.

Chapter 4 : Bluetooth Technology Website

Specification sheet Ultimaker S5 The Ultimaker S5 not only delivers best-in-class technical specifications for a desktop 3D printer, but gives you the performance and peace of mind that comes with using the complete 3D printing.

Problem Things I want to cover up in this project are: Writing new code for API to meet standards of the Swagger system. Making the API with best practices, so that it can be easily maintained and can serve for heavy traffic. Make new API endpoints available for the statistical data, provide at https: After making this API, we can easily consume the statistical data of public lab for analysis, which further generates many possibilities. Implementation Swagger Specification There are two approaches: A top-down approach where you would use the Swagger Editor to create your Swagger definition and then use the integrated Swagger Codegen tools to generate server implementation. Either you create the definition manually using the same Swagger Editor mentioned above , or if you are using one of the supported frameworks JAX-RS, node. We will be using top-down approach for our new endpoints. These endpoints are mainly of statistical endpoints. These endpoints will be new so it is better to follow swagger specification right from the beginning. For this, there will be the following steps: Logical Design of Statistical Data providing endpoints. In this section first I will brainstorm on how the endpoints should be created i. Writing them on specification file. Generating Code for the corresponding endpoints with the parameters as specified in particular endpoint. Testing the expected behavior until a satisfactory level has achieved. Documentation and blogs of the features added or updated. So we have already had some basic implementation of swagger in our app. New State After Project completion 1. Moving to the latest version version 3 for specification. How I will implement a new module is explained in next section Statistical Integration Currently, we are showing our statistical data on https: Here it is rendered inside an HTML view. It will involve the following steps: Categorising the statistical endpoints and nesting them logically. On the basis of the first step new endpoints will be added to existing statistical routes and if required the current statistical routes will be improved accordingly. Specification file will be updated accordingly. The code will be generated from the specification file and will be updated as per our requirements. Testing will be done on the various routes added and updated. If the response is not up to the good level, go to step 1. Deep dive into making it 1. There will be two files in making this feature work. The very first stats. Now I will use the features of the grape framework to make the endpoints described earlier of statistical usage. This will help in making logic clean. This class will be mounted to the root class where are adding swagger documentation. From there the gem grape-swagger-rails will take the class stats. I want to add a functionality in which you get the CSV file having data as per your requirements. This can be included as: Making an endpoint on which all of the available data columns can be seen. This authentication will be done on a token basis. If needed we can use authorization to limit it to a specific set of people. These name will go in parameters. At the backend, we catch this on a method in our controller Grape method in this case. From there we will use a custom service which will make the CSV file from it, which is having only those columns. This file will send back as a response to the client. Token Support for Stateless Nature I will use the following gems to implement this feature. Then we will run some generators: Now we will configure the knock file, as follow: The controller section of the API will be going in a separate file, which will inherit from Grape:: And other API files will inherit from it. For example, the root file will look like: This would give us authentication methods include Knock:: These are the sequence of screenshots if you want to see the actual one please hit this link. First-time contribution I am a first-time contributor to Public Labs. And yes I have gone through the welcome page and have set up the project accordingly. I have already started contributing to the following projects: I am providing this information just to clarify that I may not have contributed a lot before in Public Lab, but on my experience and skill set, I can assure you that I can do this project. I have heard that Public labs welcome new contributors and it is a very enriching experience for them. This is the thing which drives me, as a software contributor in this good. I always wonder why there are so many algorithms to do the same work, this thing puts me in the situation where I started finding the difference in between all these algorithms. Mainly of graphs and link-list. Then in the college, I got my love, Ruby. I have worked in 3 startups as a Web developer intern. I am handling both production and testing servers of the company which is on Elastic Beanstalk of AWS. Subjects includes Fundamental of computing, C, Physics, how the network works and actually what is the problem we are facing now in terms of technology trends and data. One of my driving force to learn new technologies has come from there. Those sleepless nights with code have really pushed me to make the best software possible. I am currently a coach of 3 teams in Rails Girls Summer of Code. Speaker I was the lightning talk speaker at RubyConf India I am one of the main speakers in RubyConf Kenya For your satisfaction, you can check the feedbacks of my clients on my Personal website or on my LinkedIn. I have set-up the development environment on my local machine. I have also worked on following issues 1. On those basis I can say I have familiar with the development phase of Public Lab. Teamwork and self-motivation I have worked in a team several times: In my first internship, it was a team of two. Me and senior developer, in that I learned the importance of regular update from both the sides Junior and senior developer. In my second internship, it was a team of 4 developers and I learned what other developers Data science interns and clients actually need from web developers. I understand the importance of clean and to the point git commit messages, and actually how branches work in git and GitHub. I also learned more about version control when I started contributing in OpenSource. So keeping the above experience in mind I think, I can work in a team without disturbing the mentor for common bugs and simple things which I can google. And for the same, I work hard in order to understand the things and deliver them in the correct manner. I am a big fan of a book "Software developer lifecycle manual - by John Sonmez", this is one of my great self-motivation boosters. I have worked remotely many times. So I have the experience to keep up the things even when they do not work as we expect. Passion After some alarming dangers to our environment people around the world started doing something to improve the environmental conditions around them. I want these people to connect and support each other so that a major change could take place. Public Lab is what is what we all are looking for. Public Lab can help me in to join this league of DIY people. As I am a software developer I am ready to give them full software support as per their needs. I am also impressed by the different fields of people in the community. I strongly believe that with people having this much of experience in different fields, we can definitely build something awesome, and I want to be a part of that. Audience I think the project will improve the performance of the API, which will lead to the better serving of the clients. With the statistical data endpoints, we can make apps to better monitor the data and to improve data analysis on some low power devises. Also, we can make this data available for 3rd party integration and people from around the world can access it for their own usage. I will keep my code as per the industry standards i. Commitment I believe that this work needs attention and hard-work equivalent to a full-time job and I am ready to deliver that. My main motive is to complete the work before time and also get some time for feedback and production bug fixing.

Chapter 5 : Paper for Topic: Requirements & Specifications

In software development, the V-model represents a development process that may be considered an extension of the waterfall model, and is an example of the more general V-model. Instead of moving down in a linear way, the process steps are bent upwards after the coding phase, to form the typical V shape.

It is also known as Verification and Validation model. The V-Model is an extension of the waterfall model and is based on the association of a testing phase for each corresponding development stage. This means that for every single phase in the development cycle, there is a directly associated testing phase. This is a highly-disciplined model and the next phase starts only after completion of the previous phase. V-Model - Design Under the V-Model, the corresponding testing phase of the development phase is planned in parallel. The Coding Phase joins the two sides of the V-Model. This phase involves detailed communication with the customer to understand his expectations and exact requirement. This is a very important activity and needs to be managed well, as most of the customers are not sure about what exactly they need. The acceptance test design planning is done at this stage as business requirements can be used as an input for acceptance testing. System Design Once you have the clear and detailed product requirements, it is time to design the complete system. The system design will have the understanding and detailing the complete hardware and communication setup for the product under development. The system test plan is developed based on the system design. Doing this at an earlier stage leaves more time for the actual test execution later. Architectural Design Architectural specifications are understood and designed in this phase. Usually more than one technical approach is proposed and based on the technical and financial feasibility the final decision is taken. The system design is broken down further into modules taking up different functionality. The data transfer and communication between the internal modules and with the outside world other systems is clearly understood and defined in this stage. With this information, integration tests can be designed and documented during this stage. It is important that the design is compatible with the other modules in the system architecture and the other external systems. The unit tests are an essential part of any development process and helps eliminate the maximum faults and errors at a very early stage. These unit tests can be designed at this stage based on the internal module designs. Coding Phase The actual coding of the system modules designed in the design phase is taken up in the Coding phase. The best suitable programming language is decided based on the system and architectural requirements. The coding is performed based on the coding guidelines and standards. The code goes through numerous code reviews and is optimized for best performance before the final build is checked into the repository. Unit Testing Unit tests designed in the module design phase are executed on the code during this validation phase. Unit testing is the testing at code level and helps eliminate bugs at an early stage, though all defects cannot be uncovered by unit testing. Integration Testing Integration testing is associated with the architectural design phase. Integration tests are performed to test the coexistence and communication of the internal modules within the system. System Testing System testing is directly associated with the system design phase. System tests check the entire system functionality and the communication of the system under development with external systems. Most of the software and hardware compatibility issues can be uncovered during this system test execution. Acceptance Testing Acceptance testing is associated with the business requirement analysis phase and involves testing the product in user environment. Acceptance tests uncover the compatibility issues with the other systems available in the user environment. It also discovers the non-functional issues such as load and performance defects in the actual user environment. Requirements have to be very clear before the project starts, because it is usually expensive to go back and make changes. This model is used in the medical development field, as it is strictly a disciplined domain. The following pointers are some of the most suitable scenarios to use the V-Model application. Requirements are well defined, clearly documented and fixed. Product definition is stable. Technology is not dynamic and is well understood by the project team. There are no ambiguous or undefined requirements. The project is short. The simplicity of this model also makes it easier to manage. Works well for smaller projects where requirements are very well understood. Simple and easy to understand and use. Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process. Not a good model for complex and object-oriented projects. Poor model for long and ongoing projects. Not suitable for the projects where requirements are at a moderate to high risk of changing. Once an application is in the testing stage, it is difficult to go back and change a functionality. No working software is produced until late during the life cycle.

Chapter 6 : What's new in driver development - Windows drivers | Microsoft Docs

EPICS Specification Development Module. 10ì›”25ì•¼ 6ì‹œ PenNë‰´ìŠ¤_ì‚¬ë²•ë†•ë‹¨ íŠ¹ë³„ìž¬íŒ•ë¶€? ì•¸ë¯¼ìž¬íŒ• / 3ë¶„ê¸° ì„±ìž¥ë¥ ë•„ % æ–‡ì •ë¶€ ì‹¤ë ¥ ë“œëŸ¬ë‚¬ë‹¤ / æ–‡ì •ê¶Œ ì•¼ìž•ë¦¬.

Requirements analysis[edit] In the requirements analysis phase, the first step in the verification process, the requirements of the system are collected by analyzing the needs of the user s. This phase is concerned with establishing what the ideal system has to perform. However it does not determine how the software will be designed or built. Usually, the users are interviewed and a document called the user requirements document is generated. It is used by business analysts to communicate their understanding of the system to the users. The users carefully review this document as this document would serve as the guideline for the system designers in the system design phase. The user acceptance tests are designed in this phase. See also Functional requirements. There are different methods for gathering requirements of both soft and hard methodologies including; interviews, questionnaires, document analysis, observation, throw-away prototypes, use case and static and dynamic views with users. System design[edit] Systems design is the phase where system engineers analyze and understand the business of the proposed system by studying the user requirements document. They figure out possibilities and techniques by which the user requirements can be implemented. If any of the requirements are not feasible, the user is informed of the issue. A resolution is found and the user requirement document is edited accordingly. The software specification document which serves as a blueprint for the development phase is generated. This document contains the general system organization, menu structures, data structures etc. It may also hold example business scenarios, sample windows, reports for the better understanding. Other technical documentation like entity diagrams, data dictionary will also be produced in this phase. The documents for system testing are prepared. Architecture design[edit] The phase of the design of computer architecture and software architecture can also be referred to as high-level design. The baseline in selecting the architecture is that it should realize all which typically consists of the list of modules, brief functionality of each module, their interface relationships, dependencies , database tables , architecture diagrams, technology details etc. The integration testing design is carried out in the particular phase. The designed system is broken up into smaller units or modules and each of them is explained so that the programmer can start coding directly. The low level design document or program specifications will contain a detailed functional logic of the module , in pseudocode: The unit test design is developed in this stage. Validation phases[edit] In the V-model, each stage of verification phase has a corresponding stage in the validation phase. These UTPs are executed to eliminate bugs at code level or unit level. A unit is the smallest entity which can independently exist, e. These tests verify that units created and tested independently can coexist and communicate among themselves. System Test ensures that expectations from application developed are met. The whole application is tested for its functionality, interdependency and communication. System Testing verifies that functional and non-functional requirements have been met. Load and performance testing, stress testing, regression testing, etc. Test Plans are composed by business users. UAT is performed in a user environment that resembles the production environment, using realistic data. Criticism[edit] The V-Model has been criticized by Agile advocates and others as an inadequate model of software development for numerous reasons. It is too simple to accurately reflect the software development process, and can lead managers into a false sense of security. The V-Model reflects a project management view of software development and fits the needs of project managers, accountants and lawyers rather than software developers or users. Although it is easily understood by novices, that early understanding is useful only if the novice goes on to acquire a deeper understanding of the development process and how the V-Model must be adapted and extended in practice. If practitioners persist with their naive view of the V-Model they will have great difficulty applying it successfully. It is inflexible and encourages a rigid and linear view of software development and has no inherent ability to respond to change. It provides only a slight variant on the waterfall model and is therefore subject to the same criticisms as that model. It provides greater emphasis on testing, and particularly the importance of early test planning. However, a common practical criticism of the V-Model is that it leads to testing being squeezed into tight windows at the end of development when earlier stages have overrun but the implementation date remains fixed. It is consistent with, and therefore implicitly encourages, inefficient and ineffective approaches to testing. It implicitly promotes writing test scripts in advance rather than exploratory testing; it encourages testers to look for what they expect to find, rather than discover what is truly there. It also encourages a rigid link between the equivalent levels of either leg e. It lacks coherence and precision. There is widespread confusion about what exactly the V-Model is. If one boils it down to those elements that most people would agree upon it becomes a trite and unhelpful representation of software development. Disagreement about the merits of the V-Model often reflects a lack of shared understanding of its definition. Current state[edit] Supporters of the V-Model argue that it has evolved over time and supports flexibility and agility throughout the development process. Lately, it is being adopted by the medical device industry.

Chapter 7 : ðŸŽˆ Public Lab: GSoC proposal: v2 API development | Grape | Swagger Specification

Once written, an OpenAPI specification and Swagger tools can drive your API development further in various ways: Design-first users use Swagger Codegen to generate a server stub for your API. The only thing left is to implement the server logic -- and your API is ready to go live!

Chapter 8 : Borland C++ Development Suite Specs - CNET

Differentiation Differentiation -development of cellular specialization Differentiation is a process; multi-step/multi-phase -preceded by the commitment of cell to certain fate(s).

Chapter 9 : V-Model (software development) - Wikipedia

vivo V5 Android smartphone. Announced Nov Features â€³ IPS LCD display, MT chipset, 13 MP primary camera, 20 MP front camera, mAh battery, 32 GB storage, 4 GB RAM, Corning.