DOWNLOAD PDF SYSTEM DESIGN AND IMPLEMENTATION

Chapter 1 : System Implementation - GIS Wiki | The GIS Encyclopedia

What do design and implementation of a computer-related system (e.g. a computer system, an OS, a programming language, a DBMS, a software program) mean? Think "design of a system" as a process, is its input the requirements of the users of the system, and is its output the user interface?

In summary hospital billing is amount of money to be paid or charges given to a patient for treatment care, cure of illness and other services render by the doctor. Hospitals are responsible for the supply of drugs to patients at specific drug rate, but their billing has been manually processed. The aim of this project work is to design and implement hospital billing system. Before now there have been numerous problems faced by the hospital in operating the manual system of billing which is not very efficient, a lot of paper work has been done, receipt, irrespective of the number of patients take long period of time to prepare, due to the voluminous numbers of drugs used. Records of patients serve as information for preparing management decision are very difficult to locate manually, because of these problems errors arising from such system become very noticeable. The hospital billing software is to hold information on patients bills in different departments in the hospital. The development of the system is to ensure that the number of patients being attended to on daily basis is known or track down and also at same time accounting for bills of the patients. It helps the hospital to know with ease how much they realize. This information can be used by a hospital to determine and calculate the average number of patient that is treated in the hospital in a week, month or year. The total cash the hospital can make and expenditure in a particular period of time can be determined using the figures. Files and records are not properly kept due to the use of paper to keep records. Inability to search for a particular record when the need arises. Slow processing of information. Errors are easily made and hardly detected. The manual system does not ensure durability of record storing. Information is not well maintained. To design a system that will accurately compute total bill of patients. To create a system that will be used to obtain reports of billing records of patients. To implement a system that can be used to update billing information of patients. It will ensure that patients are attended on time and a how a patient can get registered first, before the proper bill process. A program that will be interactive whereby they must be an operator staff to supply patients billing data into the computer system. Chapter one is concerned with the introduction of the research study and it presents the preliminaries, theoretical background, statement of the problem, aim and objectives of the study, significance of the study, scope of the study, organization of the research and definition of terms. Chapter two focuses on the literature review, the contributions of other scholars on the subject matter is discussed. Chapter three is concerned with the system analysis and design. It presents the research methodology used in the development of the system, it analyzes the present system to identify the problems and provides information on the advantages and disadvantages of the proposed system. The system design is also presented in this chapter. Chapter four presents the system implementation and documentation, the choice of programming language, analysis of modules, choice of programming language and system requirements for implementation. Chapter five focuses on the summary, constraints of the study, conclusion and recommendations are provided in this chapter based on the study carried out. Is health care institution or a place where people who are sick, injured are given medical treatment and care. Is a process of showing an invoice how much one owes or has to pay for services rendered. Is a person who receive or in need of treatment from a doctor in a hospital. A change in body or mind that shows that one is not healthy. This is a qualified medical doctor who practices medical service. Is a patient who is not hospital for 24 hours but visited hospital clinic for treatment. Is a patient admitted to the hospital and stay over night usually several days, weeks or months.

Chapter 2 : Systems implementation

Systems implementation is a set of procedures performed to complete the design contained in the approved systems design document and to test, install, and begin to use the new or revised Information System.

Activities include, but are not limited to: If it is a large system involving many different departments, maintenance and support may be needed for a longer time. If is a smaller system, maintenance and support may only be needed for a short time. Systems Development Methods[edit] This section discusses the most popular methods for developing computer-based information systems. A popular, traditional method is called structured analysis, but a newer strategy called object-oriented analysis and design also is used widely. Each method offers many variations. Some organizations develop their own approaches or adopt methods offered by software vendors or consultants. Most IT experts agree that no single, best system development strategy exists. Instead, a systems analyst should understand the alternative methods and their strengths and weaknesses. Structured Analysis Structured analysis is a traditional systems development technique that is time-tested and easy to understand. Because it describes the processes that transform data into useful information, structured analysis is called a process-centered technique. In addition to modeling the processes, structured analysis includes data organization and structure, relational database design, and user interface issues. Structured analysis uses a series of phases, called the systems development life cycle SDLC to plan, analyze, design, implement, and support an information system. Structured analysis relies on a set of process models that graphically describe a system. Process modeling identifies the data flowing into a process, the business rules that transform the data, and the resulting output data flow. Basically, the structured analysis technique requires that the developer defines three things: In order to see how all these functions work together, the data flow diagram DFD is needed to show the inputs, processes storage, and outputs. Object-oriented analysis defines the different types of objects that are doing the work and interacting with one another in the system and by showing user interactions, called use cases, are required to complete tasks. Systems analysts use O-O methods to model real-world business processes and operations. The result is a set of software objects that represent actual people, things, transactions, and events. Using an O-O programming language, a programmer then transforms the objects into reusable code and components. O-O analysis uses object models to represent data, behavior, and by what means objects affect other objects, By describing the objects data and methods processes needed to support a business operation, a system developer can design reusable components that allow faster system implementation and decreased development cost. The object-oriented approach has many benefits, they provide naturalness and reuse. The approach is natural because people tend to think about things in terms of tangible objects and because many systems within an organization uses the same objects i. Other Development Strategies In addition to structured analysis and O-O methods, there are other systems development techniques created by individual companies. Using MSF, you design a series of models, including a risk management model, a team model, model has a specific purpose and outputs that contribute to the overall design of the system. Although the Microsoft process differs from the SDLC phase-oriented approach, MSF developers do the same kind of planning,ask the same kinds of fct-finding questions,deal with the same kinds of design and implementation issues, and resolve the same kinds of problems. MSF uses O-Oanalysis and design concepts, but also examines a broader business and organizational context that surrounds the development of an information system [9]. Ad Hoc[edit] Ad hoc, is something that one can use to do a specific task but the process that was used cannot be used for another process. The whole project cannot run at that level. One can use a template to create a project but with Ad Hoc, it is not possible. As whole the term "Ad hoc" means for this purpose only. Often considered the classic approach to the systems development life cycle, the waterfall model mostly predictive describes a development method that is linear and sequential. Waterfall development has distinct goals for each phase of development. Once a phase of development is completed, the development proceeds drops over the waterfall into the next phase and there is no turning back. The advantage of waterfall development is that it allows for departmentalization and managerial control. A schedule can be set with deadlines for each stage of

development and a product can proceed through the development process like a car in a carwash, and theoretically, be delivered on time. Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order, without any overlapping or iterative steps. The disadvantage of waterfall development is that it does not allow for much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage. This pure waterfall model makes it very difficult because there is no room for error and that is virtually impossible when dealing with humans. In the modification waterfall model, phases of projects will overlap influencing and depending on each other. For instance, if the analysis phase is completed and the project moves into the design phase but something was left out in the requirements in the analysis phase making it hard to implement in the design phase then additional project management tasks need to be added causing an overlap. Efficiency is another reason why overlapping might occur. Some activities depend on the results of prior work. In the project planning phase, there might be some additional project management tasks that need to be added, in the analysis phase, additional analysis activities may be added, and in the design phase, additional design activities may be added. Basically, the modified waterfall model is a more efficient model to use. Today, many information systems and projects are based on the modified waterfall model. In terms of an information system, prototypes are employed to help system designers build an information system that is intuitive and easy to manipulate for end users. Prototyping is an iterative process that is part of the analysis phase of the systems development life cycle. Sometimes, end users are trying to improve on the business processes or simplify a procedure. Prototyping comes in many forms - from low tech sketches or paper screens Pictive from which users and developers can paste controls and objects, to high tech operational systems using CASE computer-aided software engineering or fourth generation languages and everywhere in between. Advantages of prototyping include; Reduction of developments time and cost User involvement.

Chapter 3 : Systems Analysis and Design/Introduction - Wikibooks, open books for an open world

System goals include ease of design, implementation, maintenance, flexibility, and efficiency. Implementation Â¶ At first, operating systems were written in assembly, but now a days C/C++ is the language commonly used.

Geoprocessing timelines Batch process timelines Best practice: High availability, redundancy, security, and special performance considerations drive requirements for increased hardware and software costs. Recommendations should be backed up with facts to support proper cost and benefit analysis. Funding constraints Recommended solutions must fit within reasonable organizational funding constraints or they will not be accepted. The final design must be affordable. An organization will not implement a solution that is beyond its financial resources. With system design, cost is a function of performance and reliability. If cost is an issue, the system design must facilitate a compromise between user application performance, system reliability, cost and schedule. The design consultant must identify a hardware solution that provides optimum performance and reliability within identified budget constraints. Current technology enables distribution of GIS solutions to clients throughout an enterprise environment, but there are limitations that apply to any distributed computer system design. It is important to clearly understand real GIS user needs and discuss alternative options for meeting those needs with system support staff to identify the most cost-effective solution. It may be necessary to review several alternative software technology patterns along with a variety of system deployment options to identify and establish the best implementation strategy. Maintain a current plan Figure Planning is critical for: Providing a framework for enterprise GIS implementation. Ensuring upper management support for required GIS investments. Managing the evolution of enterprise GIS operations. Technology is changing more rapidly every year. During the s, GIS planning was a detailed, rigorous process required to identify and justify major changes in business processes necessary to achieve the benefits provided by GIS technology. GIS implementation would take several years to reach the final planned stateâ€”and technology would be relatively stable throughout that period. Today, technology is changing much faster, and it is difficult to plan for more than one year at a time. Technology keeps improving, and adjustments must be made each year to keep pace with the change. Planning methodology is becoming more agile, adapting to the rapid change in technology. Enterprise GIS planning is an ongoing process, and should be updated on an annual basis to keep pace with technology. Most GIS deployments evolve over many years of incremental technology improvements, and: Implementation plans normally address a two- or three-year schedule to ensure that the budget is in place for the anticipated deployment needs. Project planning should be adjusted annually to take advantage of technology improvements and adjust for technology change. Changes in user workflow requirements will adjust loads on the selected platform solution. Changes in the platform architecture will identify expected performance improvements and system capacity. The CPT is a simple analysis tool that is easy to change and understand. System architecture design Figure System architecture design is an integral part of the GIS business needs assessment. GIS enterprise operations are both compute processing and network traffic intensive, which means: GIS operations can place heavy demands on server processing and network traffic loads. Network capacity can be one of the determining factors driving proper software technology selection. In some cases, hardware constraints can drive software technology selection. Infrastructure requirements should always be understood and considered before making a final software technology selection. System Design Process Figure Once you have identified your project workflows, you are ready to complete your system design. The CPT is developed for use based on a standard system architecture design process as shown in Figure Each cycle of the system architecture design process includes the following steps: Technical architecture strategyâ€”High-level overview showing user site locations, network bandwidth connections, and central data center locations. User location information is collected during the user needs analysis. User requirements analysisâ€”CPT Requirements analysis section is configured to represent the site locations, user workflows, peak loads, and network bandwidth for the enterprise design solution. Network suitability analysisâ€”CPT Design completes the network suitability analysis and identifies any communication bottlenecks. Network bandwidth upgrades are identified to

complete the network suitability analysis. Platform architecture selectionâ€”CPT Design Platform tier is configured to represent the design solution. Identify platform tier nicknames, select platforms, and identify platform rollover settings. Software configurationâ€”CPT Design Software Configuration module is used to assign workflow software to supporting platform tier software install and make workflow data source selection. Enterprise design solutionâ€”Once configured, the CPT Design tab completes the system architecture design analysis and provides the platform solution. System Architecture Deployment Strategy Figure Planning is the first step in supporting a successful system deployment. A system design team should review current GIS and hardware system technology, review user requirements, and establish a system architecture design based on user workflow needs. A deployment schedule, as shown in Figure Phased implementation strategies can significantly reduce implementation risk. Computer technology continues to evolve at a remarkable pace. Integration standards are constantly changing with technology and, at times, may not be ready to support immediate system deployment needs. New ideas are introduced into the market place every day, and a relatively small number of these ideas develop into dependable long-term product solutions. The following best practices are recommended to support a successful enterprise GIS implementation. System deployment phases Represent all critical hardware components planned for the final system solution. Use proven low-risk technical solutions to support full implementation. Include test efforts to reduce uncertainty and implementation risk. Qualify hardware solutions for initial production phase. Initial Production Phase Do not begin until final acceptance of pilot phase. Deploy initial production environment. Use technical solutions qualified during the pilot phase. Demonstrate early success and payoff of the GIS solution. Validate organizational readiness and support capabilities. Validate initial training programs and user operations. Qualify advanced solutions for final implementation. Final Implementation Phase Do not begin until final acceptance of initial production phase. Plan a phased roll-out with reasonable slack for resolving problems. Use technical solutions qualified during previous phases. Prioritize roll-out timelines to support early success. Implementation strategies are accelerating with faster technology evolution Production upgrades are scheduled to enable required technology advancement. Product upgrade deployments are integrated into production roll-out schedule when ready. Enterprise GIS environments are upgraded incrementally to meet operational requirements. Functional and performance testing is completed before production roll-out. Configuration control for each upgrade is critical for implementation success. Virtual Desktop and Server Technology Figure Virtual server technology is reducing the cost of managing a rapidly changing IT environment. Many ESRI development and testing operations are currently supported in virtual desktop or virtual server environments. Vendors are improving management and performance monitoring of virtual server environments, and it is becoming more practical to manage and deploy production environments in virtual server deployments. A majority of GIS operations are being deployed today with virtual server environments. Platform virtualization technology provides IT managers with a way to abstract the installed platform software environment from the physical platform hardware. There are two fundamental levels of virtualization, one being virtual desktop environments hosted within a physical platform operating system that interfaces with the physical platform hardware and the other being a virtual server environment hosted on a hyper-visor layer that interfaces with the assigned physical platform hardware. In both cases, the virtual desktop or server contains its own dedicated operating system and software install separate from other virtual systems deployed on the same hardware. The benefits include faster provisioning times, physical server platform consolidation, fast recovery from system failures, simplified production delivery and recovery, and optimum configuration control. All of these benefits directly contribute to lower overall systems management costs and a more stable operating environment. The disadvantages include additional software cost and some performance overhead. There may also be functional limitations limited access to hardware graphic cards and performance monitoring software which in many cases can be managed with the proper deployment selections. The real need for more rapid adaptive deployment schedules to keep pace with changing technology which also must be coupled with more stable production deployments reduced production downtime and more rapid failure recover drive the need for virtual platform environments - virtualization is one solution that addresses some real IT management needs. The potential disadvantages can be managed by proper deployment strategies. The performance overhead for

virtual desktop environments is much higher than for server environments, and for this reason virtual desktops are normally limited to software development environments where performance and scalability is not a critical factor. Server consolidation benefits can be leveraged in a Staging environment, were multiple release candidates can undergo test and validation in preparation for production deployment. Several production release candidates can be testing in parallel on the same physical server platform. Production deployment can benefit from deploying an existing virtual server install Staging configuration that has completed final test and acceptance to a higher capacity production physical server by simply moving the Staging server release to the production platform. If there is a production failure identified after deployment; it is a simple process to move the production environment back to the previous release. Deploying virtual server staging environments to a physical server production environment is also a viable option - ensuring optimum performance and scalability for the production environment.

Chapter 4 : Systems design - Wikipedia

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified blog.quintoapp.coms design could be seen as the application of systems theory to product development.

The goal of the implementation phase is to implement a system correctly, efficiently, and quickly on a particular set or range of computers, using particular tools and programming languages. This phase is a set of activities with: Design, environmental, and performance requirements. Reconciliation, transformation, conversion, monitoring, testing. Designers see objects as software abstractions. Implementors see them as software realities. However, as with the transition from analysis to design, structural continuity of concepts and constructs means that design and even analysis notions should flow smoothly and traceably. The chief inputs from design to implementation may be categorized in a manner similar to those of previous phases. Again, while the headings are the same, the details differ. A computational design of the system. The machines, languages, tools, services, and systems available to build the system. The expected response times of the system. Quality, scheduling, compatibility with other systems, etc. Implementation activities are primarily environmental. They deal with the realities of particular machines, systems, languages compilers, tools, developers, and clients necessary to translate a design into working code. Implementation-level design is a reconciliation activity, where in-principle executable models, implementation languages and tools, performance requirements, and delivery schedules must finally be combined, while maintaining correctness, reliability, extensibility, maintainability and related criteria. While OO methods allow and even encourage design iteration, such activities must be tempered during the implementation phase. In analogy with our remarks in Chapter 25 , if everything can change, then nothing can be implemented reliably. Implementation phase changes should ideally be restricted to occasional additions rather than destructive modifications. Implementation activities may be broken across several dimensions, including the construction of intracluster software, intercluster software, infrastructure, tools, and documentation, as well as testing, performance monitoring, configuration management and release management. Most of these were touched on briefly in Chapter Many excellent texts, articles, manuals, etc. In keeping with the goals and limitations of this book, we restrict further discussion of the implementation phase to a few comments about testing and assessment that follow from considerations raised in Parts I and II. A design must be testable. An implementation must be tested. Tests include the following: Tests that force most or all computation paths to be visited, and especially those that place components near the edges of their operating conditions form classic test strategies. Tests should be applied across the range of systems on which the software may execute. Tests may employ suspected nonportable constructions at the compiler, language, tool, operating system, or machine level. Tests of interobject and interprocess coordination should be built at several granularity levels. For example, tests of two or three interacting objects, dozens of objects, and thousands of them are all needed. Use cases laid out in the analysis phase should actually be run as tests. Tests may be designed to operate for hours, days, or months to determine the presence of deadlock, lockup, or nontermination. Hardware and software errors may be infused into systems before or during testing in order to test response to faults. While we do not concentrate much in this book on user interface design, any system, even one without an interactive interface, must meet basic human factors requirements. Tests and observations with potential users form parts of any test strategy. Use by outsiders rather than developers often makes up for lack of imagination about possible error paths by testers. Tests should never be thrown out unless the tests are wrong. Any changes in classes, etc. Most regression tests begin their lives as bug reports. When tests fail, the reasons must be diagnosed. People are notoriously poor at identifying the problems actually causing failures. Effective system-level debugging requires instrumentation and tools that may need to be hand-crafted for the application at hand. Classes and tasks may be armed with tracers, graphical event animators, and other tools to help localize errors. Performance Assessment Analysis-level performance requirements may lead to design-phase activities to insert time-outs and related alertness measures in cases where performance may be a problem. However, often, designers cannot be certain whether some of these measures help or hurt. Thus, while designers provide

plans for building software that ought to pass the kinds of performance requirements described in Chapter 11 , their effects can usually only be evaluated using live implementations. Poorer alternatives include analytic models, simulations, and stripped-down prototypes. These can sometimes check for gross, ball-park conformance, but are rarely accurate enough to assess detailed performance requirements. Performance tests may be constructed using analogs of any of the correctness tests listed in the previous section. In practice, many of these are the very same tests. However, rather than assessing correctness, these check whether steps were performed within acceptable timing constraints. The most critical tests are those in which the workings of the system itself are based on timing assumptions about its own operations. In these cases performance tests and correctness tests completely overlap. For example, any processing based on the timed transition declarations described in Chapters 11 and 19 will fail unless the associated code performs within stated requirements. As with correctness tests, the reasons for performance test failures must be diagnosed. Again, people are notoriously poor at identifying the components actually causing performance problems. Serious tuning requires the use of performance monitors, event replayers, experimentation during live execution, and other feedback-driven techniques to locate message traffic and diagnose where the bulk of processing time is spent and its nature. Performance tuning strategies described in Chapter 25 may be undertaken to repair problems. If all other routes fail, then the implementors have discovered an infeasible requirement. After much frustration, many conferences, and too much delay, the requirements must be changed.

Chapter 5 : Operating Systems Design and Implementation by Andrew S. Tanenbaum

Thus, maintenance changes the existing system, enhancement adds features to the existing system, and development replaces the existing system. It is an important part of system development that includes the activities which corrects errors in system design and implementation, updates the documents, and tests the data.

A system block diagram initial proposal is presented in Following the methodology described in the last update, we proceed with its implementation. The whole electronic will be mounted in a case with a protection of at least IP The power supply will be through a battery which is mounted inside the case. It shall be possible to charge the battery with a connector on the case. Navigation Module This module provides all the information about the location, time, weather conditions, and pictures taken from the hot air balloon. Will be used a GPS Module to get the latitude, longitude, time. Will be used a set of sensor to perform the measure of Humidity, Atmospheric Pressure, Temperature. The candidates suggested are the Oak Humidity and Oak Atmospheric sensors. Will be used a standard USB webcam, with an easy holder mount. The pictures will be transmit over the online connection and display them in the web application. Offline Data Storage Module This module is used to storage all produced files, tracks or photos are on a micro SD card on the computer module Iris Board , allowing easy access to the data offline. As add on will be implement a mobile app where the user can find several details related with the flight, such as, location, weather conditions, etc. Development module This module is the main interface between the whole system and its development. The DVI connector on the Iris will be accessible for development or service. This may only be possible when the case is open but it should not be necessary to disassemble everything. Its description is presented next. Context Description This project in collaboration with the company Toradex AG, aim to develop a data acquisition system of environmental data and positioning, during a hot air balloon flight. Offering to the user different possible applications where the data obtain can be used, for example, back â€”up of an existing flight instrument, weather conditions monitoring, among others that could require this kind of data. An initial set of functional requirements was provided at the beginning, here are the most relevant: Record position â€” 3D tracking Latitude, altitude, longitude. Record physical variables e. Provide offline storage of the data recorded. As well a list of hardware constraints: Main computer unit is the ColibriT20 module on an Iris Carrier board. The diagram below is a shows a representation of the system context Context diagram 2. The next table list the systems stakeholders with a short description of their interest.

Chapter 6 : Operating-System Design and Implementation â€" Operating Systems Study Guide

System Design and Implementation Every system implementation or upgrade should deliver a clear ROI and help your organization better achieve its overarching business goals. By working with TECH LOCK Â® experts, you can ensure your IT department becomes a competitive advantage for your organization instead of just a cost center.

Old and new systems are used simultaneously. Provides fallback when new system fails. Offers greatest security and ultimately testing of new system. New system may not get fair trail. Direct Cutover Conversion New system is implemented and old system is replaced completely. Forces users to make new system work Immediate benefit from new methods and control. No fall back if problems arise with new system Requires most careful planning Supports phased approach that gradually implement system across all users Allows training and installation without unnecessary use of resources. Avoid large contingencies from risk management. A long term phasein causes a problem of whether conversion goes well or not. Phase-In Method Working version of system implemented in one part of organization based on feedback, it is installed throughout the organization all alone or stage by stage. Provides experience and line test before implementation When preferred new system involves new technology or drastic changes in performance. Gives impression that old system is erroneous and it is not reliable. File Conversion It is a process of converting one file format into another. For example, file in WordPerfect format can be converted into Microsoft Word. For example, Microsoft Word can open and save files in many other word processing formats. Post-Implementation Evaluation Review PIER PIER is a tool or standard approach for evaluating the outcome of the project and determine whether the project is producing the expected benefits to the processes, products or services. It enables the user to verify that the project or system has achieved its desired outcome within specified time period and planned cost. PIER ensures that the project has met its goals by evaluating the development and management processes of the project. To identify the opportunities to add additional value to the project. To determine strengths and weaknesses of the project for future reference and appropriate action. To make recommendations on the future of the project by refining cost estimating techniques.

DOWNLOAD PDF SYSTEM DESIGN AND IMPLEMENTATION

Chapter 7 : What do design and implementation mean? - Software Engineering Stack Exchange

Design and implementation â€¢Software design and implementation is the stage in the software engineering process at which an executable software system is developed.

System elements are made, bought, or reused. If implementation involves a production process, a manufacturing system which uses the established technical and management processes may be required. The element is constructed employing appropriate technologies and industry practices. This process bridges the system definition processes and the integration process. Figure 1 portrays how the outputs of system definition relate to system implementation, which produces the implemented system elements required to produce aggregates and the SoI. A system element will be verified against the detailed description of properties and validated against its requirements. Figure 2 provides the context for the implementation process from the perspective of the U. Department of Defense DoD. It is important to understand that these views are process-oriented. While this is a useful model, examining implementation only in terms of process can be limiting. Depending on the technologies and systems chosen when a decision is made to produce a system element, the implementation process outcomes may generate constraints to be applied on the architecture of the higher-level system; those constraints are normally identified as derived system requirements and added to the set of system requirements applicable to this higher-level system. The architectural design has to take those constraints into account. If the decision is made to purchase or reuse an existing system element, it has to be identified as a constraint or system requirement applicable to the architecture of the higher-level system. Conversely, the implementation process may involve some adaptation or adjustments to the system requirement in order to be integrated into a higher-level system or aggregate. Implementation also involves packaging, handling, and storage, depending on the concerned technologies and where or when the system requirement needs to be integrated into a higher-level aggregate. The system element requirements and the associated verification and validation criteria are inputs to this process; these inputs come from the architectural design process detailed outputs. Execution of the implementation process is governed by both industrial and government standards and the terms of all applicable agreements. This may include conditions for packaging and storage, as well as preparation for use activities, such as operator training. The developing or integrating organization will likely have enterprise-level safety practices and guidelines that must also be considered. Activities of the Process The following major activities and tasks are performed during this process: Define the implementation strategy - Implementation process activities begin with detailed design and include developing an implementation strategy that defines fabrication and coding procedures, tools and equipment to be used, implementation tolerances, and the means and criteria for auditing configuration of resulting elements to the detailed design documentation. In the case of repeated system element implementations such as for mass manufacturing or replacement elements , the implementation strategy is defined and refined to achieve consistent and repeatable element production; it is retained in the project decision database for future use. The implementation strategy contains the arrangements for packing, storing, and supplying the implemented element. Realize the system element - Realize or adapt and produce the concerned system element using the implementation strategy items as defined above. Realization or adaptation is conducted with regard to standards that govern applicable safety, security, privacy, and environmental guidelines or legislation and the practices of the relevant implementation technology. This requires the fabrication of hardware elements, development of software elements, definition of training capabilities, drafting of training documentation, and the training of initial operators and maintainers. Provide evidence of compliance - Record evidence that the system element meets its requirements and the associated verification and validation criteria as well as the legislation policy. This requires the conduction of peer reviews and unit testing, as well as inspection of operation and maintenance manuals. Acquire measured properties that characterize the implemented element weight, capacities, effectiveness, level of performance, reliability, availability, etc. Package, store, and supply the implemented element - This should be defined in the implementation strategy. Artifacts and Ontology Elements This process may create several artifacts such as an

implemented system.

Chapter 8 : System Analysis and Design System Implementation and Maintenance

Systems Design, Implementation, Maintenance, and Review 4/2/ Systems Design. Systems Design A phase in the development of an IS system that answers the question "How will the information system do what it must do to obtain a solution to a problem?".

Chapter 9 : System Design & Implementation | Honeywell Intelligrated

Implementation is the process that actually yields the lowest-level system elements in the system hierarchy (system breakdown structure). System elements are made, bought, or reused. Production involves the hardware fabrication processes of forming, removing, joining, and finishing, the software realization processes of coding and testing, or.