Modelado de un sistema experto orientado al cuidado de cultivos de lechuga en un ambiente controlado tipo invernadero

Modeling facing a greenhouse lettuce crop care in an environment controlled type expert system

Manuel Rodríguez
Contacto: ing.comico@gmail.com
Laura Santana
Contacto: laurasantanao@gmail.com

Universidad Distrital Francisco José de Caldas

Tipo: Artículo de Investigación

Para citar este artículo: Rodriguez M. & Santana L. (2015). Modelado de un sistema experto orientado al cuidado de cultivos de lechuga en un ambiente controlado tipo invernadero. Revista TIA, pp 46-53

Fecha de recepción: 24 de noviembre de 2014
Fecha de aceptación: 18 de junio de 2015

Revista Digital TIA
Tecnología Investigación y Academia

Resumen
Se aborda una problemática que presenta el sector agropecuario nacional, donde se evidencia la baja calidad y altos costos de sus productos. El modelo de la investigación se centra en el cultivo de lechuga en invernaderos y tratará las problemáticas de producción y cuidado del mismo, utilizando un sistema experto para tales fines. Teniendo en cuenta las características del sistema experto y del cliente se aborda un modelo incremental basado en componentes y una metodología ágil XP. Las arquitecturas utilizadas dan respuesta a las problemáticas más importantes de la aplicación, estas arquitecturas son capas, distribuidas, de repositorio y por componentes.

Palabras clave: arquitectura distribuida, cliente servidor, metodología XP, modelo incremental, sistema experto.

Abstract
A problem that presents the national agricultural sector, where low cost and high quality of its products is evidenced addressed. The research model focuses on the cultivation of lettuce in greenhouses and address the problems of production and care of it, using an expert system for such purposes. Given the characteristics of the expert and the client system an incremental model based on an agile methodology components and XP is discussed. The architectures used respond to the most important issues on the application, these are layered architectures, distributed repository and components.

Key Words: distributed architecture , client-server, XP methodology, incremental model, expert system

I. Introducción

En Colombia el sector agropecuario enfrenta uno de los momentos más críticos, además, los tratados de libre comercio TLC han afectado de forma negativa a los pequeños productores y campesinos, ya que estos no cuentan con modelos de producción automatizados y en muchos casos se dedican a la producción de un solo cultivo, lo que limita la competitividad; de igual forma, no se logra un retorno de la inversión que sea competitivo y permita una actualización de las tecnología usadas.

Es aquí donde se pretende realizar un aporte a la sociedad aprovechando uno de los campos de la ingeniería de sistemas, la cibernética computacional, aplicado al cuidado de cultivos. Dentro de la esta se seleccionan los sistemas expertos, ya que brindan una solución a gran parte de la problemática y permiten un monitoreo de los cultivos los siete días de la semana, veinticuatro horas al día 7 x 24.

Los sistemas expertos aportaran a los cultivos de medianos y pequeños cultivadores una estructura de negocio, con la cual contarán con un experto en agronomía, disponible para realizar acciones de manera autónoma, a fin de mejorar la productividad y calidad de los cultivos. Uno de los puntos a su favor radica en que al controlar de manera autónoma las variables de entorno del invernadero, se genera un ambiente controlado que no propicia la aparición de plagas; lo anterior, acompañado de la correcta selección de las semillas, puede evitar la utilización de plaguicidas, dando como resultado final un producto orgánico que tiene un mayor valor en el mercado y es más codiciado por el usuario final.

Es así como en este documento se realiza el modelamiento de un sistema experto orientado al cuidado de cultivos tipo invernadero.

II. Modelo

Sobre la base de un análisis de la aplicación a desarrollar, se concibe la necesidad de trabajar e implementar el sistema por manejo de componentes, los cuales en su mayoría se verán reflejados como módulos encargados de las principales actividades del sistema, esto con el fin de afrontar el problema general desde una perspectiva más simple. Se plantean entonces los siguientes módulos:

1. *Módulo de monitoreo*

   Como bien lo sugiere su nombre, este módulo será el encargado de generar la interfaz con los diferentes sensores instalados dentro del invernadero, y a su vez gestionará la información y la manipulará para que esta sea comprendida y evaluada por los demás módulos.

2. *Módulo de evaluación*

   Recibirá como entrada las variables del sistema, también se alimentará de diferentes sistemas de información

gubernamentales, con el fin de evaluar las condiciones actuales del invernadero y asociar a estar variables un estado que puede ser bueno, malo o crítico.

3. *Módulo de decisión*

El módulo de decisión se puede considerar el más importante, ya que este implementa la lógica del negocio, se alimentará de una base de conocimientos que contará con las posibles acciones que se deben realizar para mantener el sistema en óptimas condiciones; este módulo solo entrara en acción cuando las variables reportadas por el módulo de evaluación presenten un estado crítico o malo. El modulo generara un *pull* de acciones a realizar y mediante un algoritmo de decisión elegirá la solución que genere el mayor impacto positivo dentro del sistema; en caso de que el algoritmo no genere una única solución esta será reportada a los agrónomos con el fin de que seleccionen una, también será reportado el Estado, dado el caso en que la solución encontrada esté fuera del alcance del sistema.

4. *Módulo de ejecución*

Una vez se ha tomado una decisión, este módulo se encargará de llevar a cabo dicha orden, indicando a los sistemas físicos las actividades a realizar, todo siguiendo los preceptos del módulo de decisión o del agrónomo.

5. *Módulo de reporte*

Todos los módulos del sistema se encuentran relacionados directamente con este módulo, ya que es el encargado de notificar o generar alertas, estados y acciones que se realizan en el invernadero. Este módulo también presenta la puerta de entrada para que el administrador del sistema y operadores generen una interacción con el aplicativo. Luego de evaluar los diferentes aspectos del software en desarrollo, se estipula la necesidad de manejar un modelo incremental, con el fin de generar un mejoramiento continuo y control de cambios de la aplicación. Es por ello que se plantea una primera iteración en la cual serán modeladas las funcionalidades básicas de cada uno de los módulos, dejando para futuras iteraciones la evolución de cada una de sus funciones.

Dentro del modelo de desarrollo incremental se puede generar un desarrollo por componentes, para ello se tienen en cuenta cinco actividades, como se observa en la figura 1, enfocadas a la creación de cada módulo.

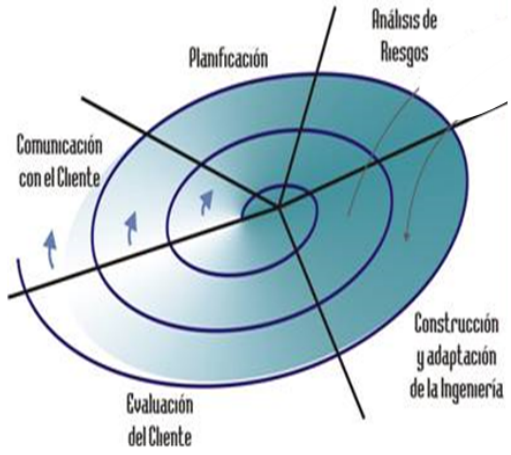

Figura 1. Desarrollo basado en componentes [1]

a) *Comunicación*

Es el momento en donde se identifican los requerimientos del sistema y se realiza el levantamiento de la información.

b) *Planificación*

En esta actividad se pretende fijar requerimientos y especificaciones y ceñirlos en un plan de desarrollo.

c) *Análisis del riesgo*

En esta actividad se pretende realizar un estudio de las amenazas y eventos no deseados y los daños y consecuencias que esas puedan producir. Como resultado de la fase se genera un prototipo, en el cual se tienen en cuenta los riesgos de cada uno de los ámbitos.

d) *Construcción y adaptación de la ingeniería*

Es la etapa en donde se identifican los componentes, se desarrollan y se utilizan de acuerdo a la necesidad que se presente.

e) *Evaluación*

Se realiza un análisis de los posibles cambios que requiera el módulo, si estos son viales se da paso a un nuevo ciclo y comienza de nuevo.

III. Metodología

Teniendo en cuenta la visión de componentes del sistema y al tener certeza de que se desarrollará de forma evolutiva basada en iteraciones, nace la necesidad de realizar una retroalimentación al finalizar cada iteración, como resultado de dicho análisis se evidencian nuevos requisitos o la necesidad de implementar cambios en algunos ya existentes. Estos ciclos deberán ser de cortos periodos de tiempo ya que si los cambios que son solicitados son críticos es necesario que no se incurra en una pérdida de recursos (tiempo, costos, etc.) en la implementación de los requerimientos equivocados (entiéndase equivocados como aquellos que no son los que el sistema exige se tengan en cuenta). Para dar respuesta a estas necesidades es apropiado implementar una metodología ágil, en donde es vital una constante revisión y pruebas dentro de cada ciclo, ya que estos dejan en evidencia puntos a mejorar y fortalezas de manera temprana; de igual forma, con ayuda de pruebas unitarias, de integración y validación se obtiene una visión del camino que se está tomando, con el objetivo de saber si es necesaria una reestructuración del mismo o solo se tienen en cuenta pequeñas mejoras.

Debido a las necesidades presentadas en lo anterior, se plantea el uso de la metodología XP; con dicha metodología se estipula un tiempo de tres semanas para cada iteración, en las cuales se debe entregar un producto funcional que haya

superado las pruebas establecidas. En cada una de las iteraciones se agregaran funcionalidades avanzadas que darán nuevas opciones y características a la aplicación [4]. Una de las pruebas más importantes son las de integración de los módulos de evaluación y decisión con el motor de inferencia y la base de conocimiento, que son la base del sistema, en donde la evidencia temprana de una falla o redefinición puede reducir los costos a gran escala; aún más que si se encuentran en un estado avanzado del desarrollo.

Por lo tanto, es claro que no se puede incurrir en errores que son detectados en etapas avanzadas del proceso, o por lo menos que los detectados no sean de gran implicación, ya que si se pasan por alto los resultados que se verán en el motor de inferencia, pueden hacer que el cultivo se pierda o se requiriera una gran inversión monetaria para lograr salvarlo. Ante estas proposiciones, entre otras, se evidencia que la metodología XP cumple con la mayoría de las necesidades planteadas.

> "Esta metodología consiste en un conjunto de prácticas, fundamentadas en valores que deben de mantener los participantes de proyecto, que a manera de trabajo en grupo, pretende lograr como producto final un software con mayor sencillez de desarrollo y un grado muy alto de calidad. [2]"

A lo largo de los años la aplicación de esta metodología ha dado resultados positivos en cuanto al cumplimiento de objetivos, las etapas que plantean han sido llevadas a muchos lugares del mundo, y en diversos ámbitos de la actividad humana donde se ha necesitado diseñar un sistema informático. La metodología XP cuenta con unas características especiales, como se observa en la figura 2, donde se encuentran las fases y actividades que sugiere. XP como metodología propone cuatro fases, las cuales se exponen a continuación:

a) *Fase de exploración:*

En esta fase se especificará el alcance general del proyecto. Se deben simplificar los requerimientos y trasladarlos a historias de usuarios sencillas. Es la fase de estimación de tiempos de desarrollo primarios,

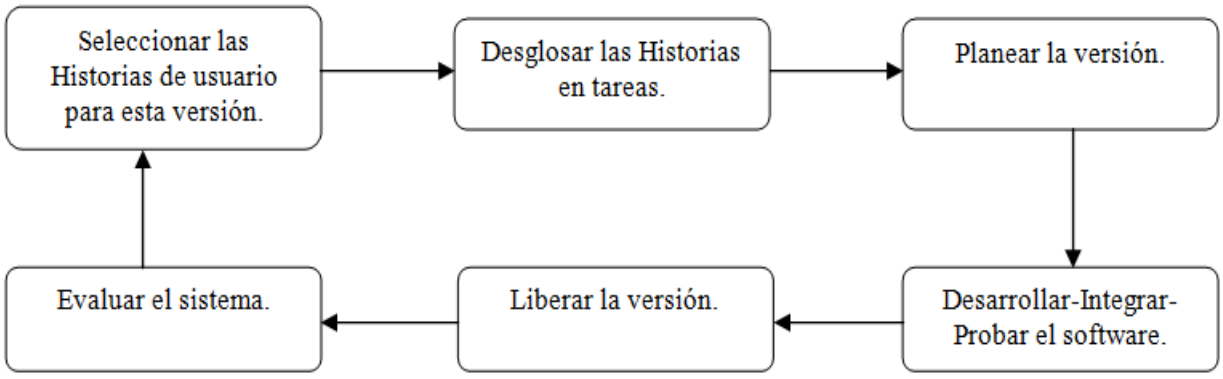

Figura 2. Ciclo de vida de un proyecto XP

ya que pueden cambiar en la retroalimentación de las interaciones. El objetivo de esta fase es visualizar el sistema de manera general y definir un tiempo aproximado de realización.

b) Fase de planificación

En esta fase los implicados internos formulan un orden de desarrollo de las historias de usuario incluyendo entregas. Para ello es necesario reunirse en varias oportunidades. El objetivo de esta fase es generar un plan de entregas.

c) Fase de iteraciones

Probablemente la fase más importante, ya que las actividades fundamentales se presentan acá, cada iteración presenta un producto funcional, en donde se implementaron las historias de usuario en el orden definido previamente.

Es necesario aclarar que, como las historias no son detalladas, se deberá realizar un análisis al comienzo de la fase en el que se diseñe cómo se llevarán a cabo dichas historias, se desarrollarán los elementos propuestos, se probarán y se entregará el avance.

d) Fase de puesta en producción

Si bien al final de cada iteración se entregan módulos funcionales y sin errores, puede ser deseable por parte del cliente no poner el sistema en producción, hasta tanto no se tenga la funcionalidad completa.

En esta fase no se realizan más desarrollos funcionales, pero pueden ser necesarias tareas de ajuste.

Las actividades que recomienda seguir esta metodología se plantean de manera sencilla pero a la vez son estrictas, y se inspiran en la total participación de los involucrados. Dichas etapas son:

a. Planificación

Se crean las historias de los usuarios, se pondera la importancia de las mismas, al igual que unos costos, se define entonces la agrupación de las mismas para la iteración venidera. Es decir, esta actividad se encuentra presente dentro de las tres primeras fases (exploración, planificación e iteraciones) y su resultado se observa en la tercera y cuarta fase (iteración y entrega del producto).

b. Diseño

Se propone manejar un diseño simple que genera una guía de implementación de las historias, este es en sí un artefacto que se puede modificar continuamente ajustado al cambio de requerimientos, se apoya especialmente en las tarjetas CRC (Colaborador-Responsabilidad-Clase) para identificar y organizar las clases orientadas al objeto.

c. Codificación

Antes de entrar a desarrollar esta actividad se recomienda realizar una prueba de unidad, esto con el fin de que el desarrollador tenga la claridad del enfoque que debe seguir. La programación en pareja es una característica de la metodología en donde, mientras un programador desarrolla el otro revisa, los roles se intercambian dentro de las parejas.

d. Pruebas

Esta actividad es muy importante dado que es una de las que deja ver el avance del proyecto o las falencias con las que cuenta, se definen cuatro tipos de pruebas para la metodología, las cuales son las de: unidad, integración,

validación y aceptación, estas últimas son las que apoyadas en las historias de usuario verifican la funcionalidad del entregable.

IV. Arquitectura

Dentro de la arquitectura se busca descubrir la organización y estructura global del sistema, este se considera uno de los pasos más importantes al interior de las metodologías ágiles; y en algunos casos es implementado desde las primeras etapas de desarrollo [3].

Para tomar una decisión acertada es necesario tener en cuenta la estrecha relación que existe entre los requerimientos no funcionales y la arquitectura de software, por ello tendremos en cuenta los siguientes aspectos:

a. *Rendimiento y mantenibilidad*

El rendimiento se considera un factor importante dentro del sistema, mas no un factor crítico, por ello desde el modelamiento del sistema se contempló crear un sistema modular que genera una constante comunicación entre sus componentes, sin que esto afecte de manera significativa el desempeño del sistema, pero que por otro lado favorece el mantenimiento y actualización que dé lugar a mejorar las características y funcionalidades de cada componente.

b. *Seguridad*

En todo sistema de información un factor importante y que no se puede tomar a la ligera es garantizar la integridad y seguridad, tanto de los equipos como de los datos, por ello los módulos de inferencia y base de conocimiento deberán estar ubicados en las capas internas con un alto nivel de validación, dando como resultado una arquitectura por capas para cada uno de los módulos.

c. *Protección*

La protección del sistema en es un factor crítico, por ello nuestra arquitectura debe garantizar un módulo que se ocupe de las operaciones relacionadas con la protección. Esto reducirá notablemente los costos y ofrecerá la posibilidad de tener módulos de protección relacionada, que en caso de falla desactiven con seguridad el sistema.

d. *Disponibilidad*

Es un factor crítico ya que el servicio que el sistema presta funcionará las veinticuatro horas de los sietes días de la semana (7 x 24). Para dar solución a este requerimiento será necesario abordar una arquitectura distribuida, en la que existan componentes redundantes, de manera que sea posible sustituir y actualizar componentes sin que se detenga el sistema.

V. Conclusiones

Dado el contenido del artículo se propone la posibilidad de realizar una implementación utilizando los modelos y las metodologías planteadas, se evidencia que se cumplen a satisfacción con los requerimientos mínimos para llevarla a cabo y con la metodología se observa que existe una gran posibilidad de extensión de las funcionalidades básicas del proyecto.

VI. Agradecimientos

Los autores agardecen a la ingeniera Alexandra Abuchar por gestionar el desarrollo del presente artículo.

VI. Referencias

[1] Metodología, S.f., Sinapsys, [en línea]. Consultado el 18 de octubre de 2014, disponible en: http://www.paginasprodigy.com/escomino/Sinapsys/sinapsysMetodologia.htm

[2] R. Pressman, *Ingeniería del Software un enfoque práctico*, Editorial Mc Graw Hill. Quinta Edición, Desarrollo basado en components, 2013.

[3] I. Somerville, *Ingeniería de Software*, Pearson, Novena Edición. Capítulo 6 Diseño arquitectónico, 2011.,

[4] R. Pressman, *Ingeniería del Software un enfoque práctico*, Editorial Mc Graw Hill. Sexta Edición, Capítulo 4 Desarrollo ágil, 2012,

[5] History: The Agile Manifesto, S.f., [en línea]. Consultado el 18 de septiembre de 2014, disponible en: http://www.agilemanifesto.org/history.html