АЛГОРИТМИЗАЦИЯ ЭТАПОВ СОЗДАНИЯ ОТКАЗОУСТОЙЧИВЫХ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

Абидов А. А., к.т.н.

доцент кафедры «Информационные технологии в цифровой экономике» Ташкентский государственный экономический университет

Аннотация: *Развитие цифровой экономики неразрывно связано с нарастающим ростом влияния информатизации во все области системы массового обслуживания, компьютерных сетей, органов управления и многих социальных сфер и отраслей экономики. В современном мире интенсивность развития компьютернқх систем зависит от надежной работы программного обеспечения информационных систем различного уровня. В настоящей статье автором предложен подход для создания программных средств функционирование в условиях возмущающего воздействия среды для периодической диагностики и анализ нормального состояния контролируемого продукта. Кроме этого, в статье показано трансформация информационных технологий в сторону создания безотказных систем обеспечения информационной безопасности.*

Ключевые слова: *Цифровая экономика, надежность программного обеспечения, диагностика, безотказность, возмущающее воздействие среды, восстановление нормального состояния.*

Abstract*: The development of the digital economy is inextricably linked with the growing influence of informatization in all areas of the queuing system, computer networks, governments and many social spheres and sectors of the economy. In the modern world, the intensity of the development of computer systems depends on the reliable operation of the software of information systems of various levels. In this article, the author proposes an approach for creating software tools for functioning under disturbing environmental influences for periodic diagnostics and analysis of the normal state of a controlled product. In*

addition, the article shows the transformation of information technology towards the creation of trouble-free information security systems.

***Key words**: Digital economy, software reliability, diagnostics, non-failure operation, environmental disturbance, normal state restoration.*

Введение

Разработка успешного IT-продукта – сложный многоступенчатый процесс с рядом обязательных этапов, часть из которых может идти параллельно. Стоит отметить, что аутсорс-компании по-разному определяют этапы разработки продукта и здесь важно, чтобы процесс был полностью прозрачен для разработке программ заказчика. Существуют различные подходы определения этапов разработки программных средств. В данной статье предложенное направление учитывает классическое понятие ко всем этапам создания информационных систем. При этом рассматривается и вопросы обеспечения информационной безопасности.

Анализ литературы по теме

На протяжении всей истории создания программных продуктов предпринимаются целый ряд попыток формулирования единого набора правил и методик создания программного обеспечения.

Создателями самых известных процессов являются крупные компании — производители ПО. Например: MSF (Microsoft Solution Framework [1]) от компании Microsoft или RUP (Rational Unified Process) от компании Rational Software Corp. Но та же Microsoft использует для разработки собственных продуктов смесь MSF и Agile methodology, признавая, что полностью описанный, унифицированный фреймворк MSF несколько далек от практики. Многие успешные компании-разработчики программного обеспечения используют в качестве процесса производства компиляцию общепризнанных подходов и методик.

Полный цикл разработки программного продукта состоит из следующих основных стадий: анализ требований; разработка архитектуры; разработка продукта; тестирование; установка клиенту; поддержка.

Считается, что планированием должны заниматься менеджеры или руководители команд. На самом деле, в нем участвует каждый работник внутри проектной команды [2]. На этапе написания спецификаций весь проект разбивается на подзадачи. Часто промежуток времени между окончанием работы первого программиста и началом работы тестера достаточно большой. Это не позволяет разработчику исправлять свои ошибки по «горячим следам», пока не забыты все особенности кода [3]. Юнит-тесты — это программные модули, которые тестируют внешние методы и свойства данного класса.

Хранилище исходного кода позволяет хранить все исходные коды проекта в одном месте, вести лог всех изменений с указанием номера версии, даты, имени программиста и описания изменений. Программ, которые реализуют функции хранилища, много. Возможно, самая популярная из бесплатных программ — SVN[4].

Более подробно о работе с SVN можно прочитать в материал http://tortoisesvn.tigris.org/ «Управление версиями в Subversion»[5].

Хранилище задач позволяет вести учет всех изменений системы, распределять те или иные задачи между релизами, служит единым местом постановки задач всем членам проектной команды. Примерами подобных систем являются FogBugz[6] и Jiro[7].

Функции по гарантии качества выполняют люди, кровно заинтересованные в успехе продукта, для которых не бывает мелочей. Например, в компании Apple такую функцию выполнял ее директор Стив Джобс. И, как правило, именно от этих людей зависит то, как воспримут продукт конечные пользователи.

Джоэль Спольски[8], основатель и владелец компании Fog Creek Software, один из создателей Excel в компании Microsoft, придумал тест «The Joel Test: 12 Steps to Better Code»[9].

Учеными доказано, что большие данные играют огромнейшую роль в нашей жизни. Без них невозможно существование многих привычных вещей, которые нас окружают. С помощью больших данных можно автоматизировать многие процессы, практически без участия человека.

Невозможно представить современный мир со всеми его услугами и сервисами, которые мы сегодня потребляем без использования больших данных и современных программных продуктов.

Методология исследования

Методологической базой исследования являются системный и аналитический подход, позволяющие представить научные исследования социально-экономический явлений и процессов в их развитии, взаимосвязи и взаимообусловленности.

Методология исследования определена принципами научного познания, научными достижениями, отраженными в публикациях классиков и ученых современного периода в области разработки и эффективной эксплуатации информационных систем. Методической основой явились разработки современных ученых, занимающихся оптимизацией разработки и применения программного обеспечения в отраслевых информационных системах, законодательные и нормативные акты Республики Узбекистан по развитию отрасли информационно-коммуникационных технологий.

В ходе написания данной статьи использованы методы логического, сравнительного, экономико-математического анализа.

Анализ и результаты

Информационные системы и технологии в современном мире – главный инструмент развития нового технологического уклада.

Также обеспечения качества жизни людей, национальной безопасности.

Как было отмечено выше, важную роль в эффективном функционировании современных информационных систем играет программное обеспечение.

Программное обеспечение можно охарактеризовать, как некий набор программ, правил, а также соответствующей документации системы, предназначенных для эффективной и оптимальной обработки информации.

В современный период в Республике Узбекистан все отраслевые данные накапливаются в статистических органах. И организация работы этой сферы требуют отчеты всех предприятий республике периодически, и этот период программное обеспечение функционировает по правилу 24/7, т.е. в онлайн режиме.

В этой связи можно провести формализацию этапов реализации задачи создания и эксплуатации программных продуктов (рис. 1).

Пусть задачи связанные с электронным документооборотом в некоторой сфере составляют следующее множество – {M}, которое разделим формально на задачи M_1, M_2, ...

$$\{M\}=(M_1, M_2, ...,M_n) \quad (1)$$

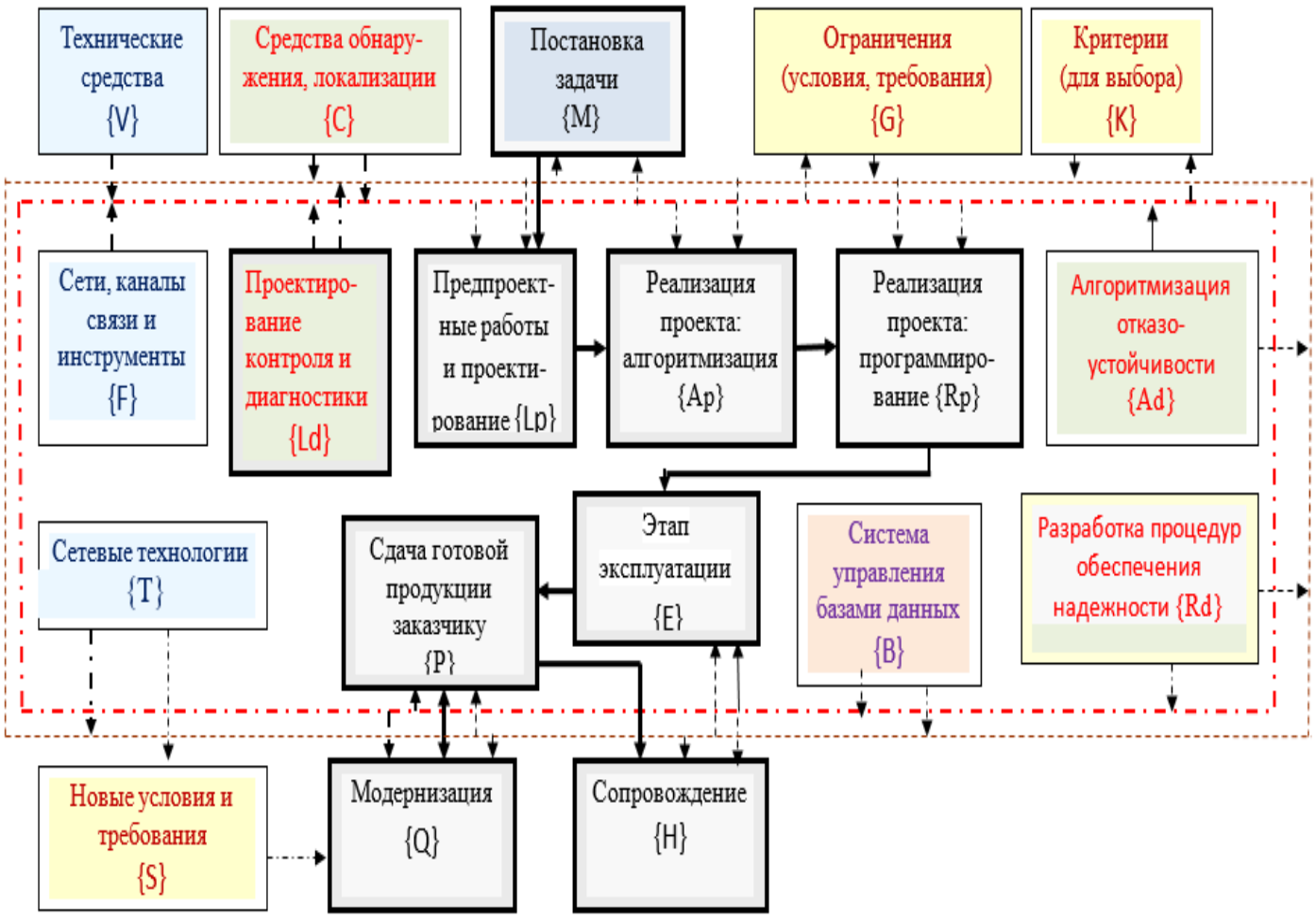
Предположим, что для каждой i- задачи имеется несколько ограничений, но не более m. Тогда, можно составить следующие формулы(2) и (3).

$$\{G\}=(G_1, G_2, ...,G_n) \quad (2)$$

Где, Gi – множество ограничений для i- задачи.

Точно также выделим множество критериев математическим символом Ki для n задач формулой (3) и средств обнаружения и локализации ошибок множеством Ci для n задач формулой (4):

$$\{K\}=(K_1, K_2, ...,K_n) \quad (3)$$



Прямые линии – переход к этапам разработки и функционирования системы
Пунктирные линии – учет ограничений, критерий, новых условий и требований

Рис. 1. Функциональная схема этапов создания отказоустойчивих систем реального времени

$$\{C\}=(C_1, C_2, ..., C_n) \quad (4)$$

Далее, введем следующие обозначения:

- комплекс технических средств - Vi,

- каналы связи и инструменты - Fi,

- технология сетей - Ti,

- системы управления базами данных – Bi.

Элементы этих комплексов, множеств можно представить следующими формулами (5), (6), (7), (8)

$$\{V\}=(V_1, V_2, ..., V_n) \quad (5)$$

$$\{F\}=(F_1, F_2, ..., F_n) \quad (6)$$

$$\{T\}=(T_1, T_2, ..., T_n) \quad (7)$$

$$\{B\}=(B_1, B_2, ..., B_n) \quad (8)$$

Для решения конкретной алгоритмической задачи будут затребованы следующие этапы:

- предпроектные и организационные работы($L^p i$),
- проектирование контроля и диагностики($L^d i$),
- алгоритмизация этапов создания ПО (A^p_i),
- алгоритмизация отказоустойчивости (A^d_i),
- разработка проекта и производства ($R^p i$),
- разработка процедур обеспечения надежности $\{R^d_i\}$
- эксплуатация проекта (Ei),
- сдача законченной работы заказчику(Pi),
- сопровождение проекта на уровне программной разработки(Hi),
- изменение программы для модернизации(Qi).

Эти комплексы работ формально будут представлены следуюшим образом:

$$\{L\}=(L_1, L_2, ..., L_n) \quad (9)$$

$$\{A\}=(A_1, A_2, ..., A_n) \quad (10)$$

$$\{R\}=(R_1, R_2, ..., R_n) \quad (11)$$

$$\{E\}=(E_1, E_2, ..., E_n) \quad (12)$$

$$\{P\}=(P_1, P_2, ..., P_n) \quad (13)$$

$$\{H\}=(H_1, H_2, ..., H_n) \quad (14)$$

$$\{Q\}=(Q_1, Q_2, ..., Q_n) \quad (15)$$

От предпроектных и организационных работ до конца выполнения всех этапов проекта будет определен функциональная последовательность,

согласованный с компетентными органами, и программное обеспечение будет основано на выбранном алгоритме и фазе реализации проекта.

В качестве ограничений можно перечислить следующие факты:

- ответ на каждый запрос не могут превышать одну минуту, или могут применяться общие требования, включая информацию обо всех юридических лиц республики, условиях использования и других.

Предпроектные и организационные работы связаны с задачей связаны с задачей предпроектные работы и проектирование ($L^p i$) и проектирование контроля и диагностики ($L^d i$).

Данную взаимосвязь представим в виде формулы (16)

$$\{L\}=(L^p, L^d) \quad (16)$$

Здесь R^p, R^d определяются следующими (17) и (18) формулами:

$$\{L^p\}=(L^p_1, L^p_2,..., L^p_n) \quad (17)$$

$$\{Ld\}=(L^d_1, L^d_2,..., L^d_n) \quad (18)$$

Задачи разработки алгоритмизация этапов создания ПО (A^p_i) связанқ с задачей алгоритмизация отказоустойчивости (A^d_i).

Данную взаимосвязь представим в виде формулы (19)

$$\{A\}=(A^p, A^d) \quad (19)$$

Здесь A^p, A^d определяются следующими (20) и (21) формулами:

$$\{A^p\}=(A^p_1, A^p_2,..., A^p_n) \quad (20)$$

$$\{A^d\}=(A^d_1, A^d_2,..., A^d_n) \quad (21)$$

Задачи разработка проекта и производства ($R^p i$) связана с задачей разработка процедур обеспечения надежности $\{R^d_i\}$.

Данную взаимосвязь представим в виде формулы (22)

$$\{R\}=(R^p, R^d) \quad (22)$$

Здесь R^p, R^d определяются следующими (23) и (24) формулами:

$$\{R^p\}=(R^p_1, R^p_2,..., R^p_n) \quad (23)$$

$$\{R^d\}=(R^d_1, R^d_2,..., R^d_n) \quad (24)$$

Как известно, на стадии эксплуатации, могут появиться новые требования и условия (S). Их можно представить в виде формулы (25).

$$\{S\}=(S_1, S_2, ..., S_n) \quad (25)$$

Тогда, создав множества {Ω} и{Ψ}, все процессы-этапы создания программного продукта условно обозначим формулой (25), и множество {Ω} определить формула (26):

$$\{\Omega\}=(M, L, A, R, P, E, Q, H) \quad (26)$$

Ограничения, условия, требования, критерии условно обозначим множеством (Ψ), соответственно формулой (27) :

$$\{\Psi\}=(C, G, R, S) \quad (27)$$

Кроме того, нам понадобится множество, охватывающее технические средства, указывающее взаимосвязь с компьютерной сетью, каналами связи, языками программирования и базой данных и т.д. Данные взаимосвязи можно выразить следующим множеством:

$$\{Э\}=(V, F, T, B) \quad (28)$$

И наконец, результатом или математическим моделираванием будет определена последовательностью операций над тремя множествами

$$\{Y\} = \{\Omega\}U\{\Psi\}U\{Э\} \quad (29)$$

В произвольно вқбранной отрасли или сфере экономики этапы создания отказоустойчивых систем программного продукта формально являются частью этих фаз-этапов разработки и их компонентов.

Таким образом, из выше определенной модели следует, что большую роль в эффективном функционировании современных информационных систем играет программное обеспечение. Его оптимизация в условиях интеграции информационных систем является важной задачей, способствующей эффективному функционированию отраслей и сфер национальной экономики

Заключение

В условиях формирования единой интегрированной информационной системы национальной экономики, программное обеспечение должно служить эффективным инструментом для надежной организации поступления информации в интегрированную систему, приведения ее к виду, удобному для принятия оптимальных управленческих решений. Данную задачу на сегодняшний день выполняют хранилища данных, работающие по специальным алгоритмам. Хранилища данных выполняют функции предварительной подготовки и хранения данных для системы принятия решений на основе информации из системы управления предприятием (или базы данных предприятия или отрасли), а также информации из сторонних источников, которые в достаточном количестве стали доступны на современном рынке информационных продуктов и услуг и с каждым днем получают все большее и большее развитие.

Использованая литература

[1] Michael S. V. Turner. Microsoft® Solutions Framework Essentials. Microsoft, 2006.

[2] Kent Beck, Martin Fowler. Planning Extreme Programming. Addison-Wesley Professional, 2000.

[3]Kent Beck. Test Driven Development: By Example.Addison-Wesley Professional, 2002.

[4] http://tortoisesvn.tigris.org/

[5] http://svnbook.red-bean.com/

[6] http://www.fogcreek.com/FogBugz/

[7] http://www.atlassian.com/software/jira/

[8] http://www.joelonsoftware.com/AboutMe.html

[9] http://www.joelonsoftware.com/articles/fog0000000043.html