УДК 681.3.06

ВИКОРИСТАННЯ ВЕБ-ТЕХНОЛОГІЙ В СИСТЕМАХ КЕРУВАННЯ

Рязанцев О.І., Щербаков Є.В., Щербакова М.Є.

THE USE OF WEB TECHNOLOGIES IN CONTROL SYSTEMS

Ryazantsev O.I., Shcherbakov E.V., Shcherbakova M.E.

Розглянуті сучасні мови розмітки, стилізації і програмування веб – додатків та проаналізована ефективність програм та візуальних інтерфейсів, розроблених з їх використанням. Показано, що можливостей цих мов та засобів їх реалізації достатньо для розробки основних підсистем робочих станцій систем керування технологічними процесами. Досліджені особливості планування та запуску задач, які функціонують на робочих станціях комп'ютерних систем керування. Для задач верхнього рівня керування розроблені методика і алгоритм диспетчеризації задач, який враховує для кожної задачі відразу декілька показників, наприклад, її важливість, час вирішення задачі, інтенсивність роботи задачі з базами даних і т. п. Запропонований алгоритм диспетчеризації задач використовується в диспетчері реального часу сучасної версії пакету „Кварц-2013".

Ключові слова: *HTML, CSS, JavaScript, веб – виконавці, комп'ютерна система керування, реальний вимір часу, диспетчеризація.*

Вступ. В даний час мова гіпертекстової розмітки HTML, таблиці стилів CSS та мова програмування JavaScript стали настільки розповсюдженими, що багато операційних систем, зокрема, Windows 8, Firefox OS, Gnome та Chrome OS прийняли їх в якості основних засобів розробки додатків. Окрім того, мобільні пристрої на базі ОС iPhone та Android підтримують веб-представлення, що дозволяє включати функціональність HTML5 та JavaScript в їх рідні додатки. JavaScript також просувається в світ технічних засобів. Такі проекти, як Arduino, Tessel, Espruino і NodeBots в найближчому майбутньому передвіщають час, в якому JavaScript може стати спільною мовою для систем керування та робототехніки.

Метою роботи є розробка методу планування задач реального виміру часу з урахуванням багатьох їх характеристик та реалізація алгоритмів цього методу сучасними засобами веб – розробки.

Основні компоненти та ефективність засобів веб – розробки. Створення програми на JavaScript [1] таке ж просте, як редагування текстового файлу і відкриття його в браузері. Тут немає складних середовищ розробки, які потрібно завантажувати і встановлювати, і не потрібно вивчати складні IDE. Крім того, мова JavaScript проста в освоєнні. Базовий синтаксис відразу знайомий будь-якому програмісту, який вже мав справу хоча б з однією з мов сімейства C. Жодна інша мова не може похвалитися таким низьким бар'єром входження, як JavaScript. В сучасних браузерах більшість коду JavaScript - це відкомпільований, добре оптимізований і виконуваний як машинний код, і тому швидкість виконання близька до програм, написаних на традиційних мовах програмування.

В даний час засоби веб - розробки на базі HTML 5, CSS 3 та JavaScript за своїми функціональними можливостями і швидкості виконання розроблених веб - додатків досягли рівня, достатнього для їх використання в системах керування, які працюють в режимі реального виміру часу. Дійсно, засоби швидкого відображення графічних і мультимедійних даних на базі Canvas API, веб - сокети для зв'язку робочих станцій і контролерів у реальному часі на базі протоколів Інтернет, технологія Web Storage для швидкого збереження даних у зовнішній пам'яті комп'ютера, механізми веб - виконавців HTML 5 для обробки інформації в паралельних потоках і, нарешті, істотний прогрес в обчислювальній ефективності алгоритмів, написаних на JavaScript, дають можливість застосовувати веб - технології практично для всіх підсистем верхнього рівня систем керування технологічними процесами і об'єктами.

З перелічених вище веб - технологій важливу роль відіграють веб - виконавці, оскільки саме вони дають можливість розробляти конкурентноздатні засоби диспетчеризації обчислювальних процесів у вузлах систем керування, які працюють в режимі реального виміру часу.

Веб - виконавці представляють собою ефективно працюючі потоки виконання [2]. Веб - виконавці живуть в автономному середовищі виконання, однак без доступу до об'єктів браузера Window або Document, і можуть спілкуватися з головним потоком тільки через механізм асинхронної передачі повідомлень. Це означає, що

паралельна модифікація об'єктної моделі документа (DOM) все ще неможлива, але разом з тим веб - виконавці дають можливість використовувати синхронні API і писати функції, які виконуються досить довго та при цьому не гальмують цикл обробки подій і не підвішують браузер. Створення нового виконавця не є великоваговою операцією, такою, як відкриття нового вікна браузера. Але виконавці не найпростіші потоки з усіх, і тому не має сенсу створення нових виконавців для виконання тривіальних операцій. Для складних веб - додатків може виявитися корисним створювати десятки виконавців, але малоймовірно, що додатки з сотнями або тисячами виконавців будуть зустрічатися на практиці.

Як у будь-якому потоковому API, є дві частини визначення веб - виконавця. Першою частиною є об'єкт Worker: він визначає, як виконавець представляється ззовні, з боку потоку, який його створює. Другою частиною є WorkerGlobalScope - глобальний об'єкт для нового виконавця, який визначає, як потік виконавця виглядає з внутрішньої сторони, з коду виконавця.

Потік виконавця виконує свій код (і всі імпортовані сценарії) синхронно зверху вниз, а потім входить в асинхронну фазу, в якій він відповідає на події та таймери. Якщо виконавець зареєстрував обробник події onmessage, він не буде завершений ні за яких обставин до тих пір, поки є можливість надходження нових повідомлень. Але якщо виконавець не слухає повідомлень, він працюватиме лише до тих пір, поки існує невирішені завдання (такі як завантаження і обробка таймерів) і не будуть опрацьовані всі зворотні виклики, пов'язані з завданнями. Після того, як завершиться обробка всіх зареєстрованих зворотних викликів, у виконавця зникає можливість почати нове завдання і в цей час можна безпечно завершити роботу потоку.

Нижче на прикладі доволі складної підсистеми диспетчеризації задач робочих станцій систем керування буде показано, як можна використовувати описані вище веб – технології для програмування систем реального виміру часу.

Алгоритм диспетчеризації задач реального виміру часу за структурним критерієм. Особливістю управління обчислювальним процесом в режимі реального виміру часу є те, що для всіх задач, які підлягають виконанню, наперед відомі всі основні характеристики, а також системні ресурси, які використовуються під час їх роботи. Це дає можливість підвищити ефективну продуктивність системи, якщо для диспетчеризації задач будуть використовуватися алгоритми, що забезпечують у порівнянні з найпростішими алгоритмами диспетчеризації більш повне завантаження процесора і мінімізують втрати продуктивності через колізії, які виникають при одночасній роботі із загальними системними ресурсами.

Диспетчеризація задач, тобто вибір задач на виконання процесорами робочої станції, представляє собою серйозну проблему [3]. Доводиться враховувати, що одночасна робота декількох процесорів із загальними ресурсами системи, такими, як лінії зв'язку, таблиці даних, може призвести до значного зниження продуктивності.

Найпростішими і часто використовуваними дисциплінами обслуговування є добре відомі навіть з повсякденного побуту безпріоритетна дисципліна «першим прийшов - першим обслуговується» і дисципліна обслуговування в порядку старшинства пріоритетів. Стосовно дисципліни вибору задач на вирішення це означає, що вибирається задача, яка раніше за всіх прийшла в чергу або має вищий пріоритет. Така дисципліна вимагає лінійного впорядкування всіх задач, які підлягають виконанню, що для складних систем реального виміру часу, в яких для встановлення старшинства між задачами потрібно враховувати кілька не порівнюваних між собою показників, може виявитися недоцільним. У той же час можливість паралельного виконання в ОС Windows декількох задач дозволяє перейти від їх лінійного впорядковування по одному, скалярному критерію до структурного, тобто до диспетчеризації задач відразу за групою критеріїв [4]. При цьому черговість встановлюється не між окремими задачами, а між групами рівноцінних задач. Такий підхід не дає ніяких переваг, якщо для впорядковування черги задач є тільки один числовий показник, але відкриває нові можливості, якщо таких показників декілька.

Показниками, що характеризують задачу, можуть бути: період повторення задачі, об'єм оперативної пам'яті, зайнятої задачею, час її вирішення, інтенсивність роботи з таблицями баз даних, час зайнятості каналів зв'язку для отримання значень технологічних параметрів від OPC - серверів і т. д. Може бути встановлено порядок вирішення задач по кожному з показників, але інформація, необхідна для порівняння цих показників між собою, як правило, відсутня. Одна задача може вимагати багато часу для роботи з таблицями баз даних, мало часу зайнятості каналів зв'язку і середнього часу вирішення, інша - малого часу для роботи з таблицями баз даних, великого часу зайнятості каналів зв'язку і багато часу для вирішення. Як порівняти такі задачі?

Розглянемо методику порівняння задач, що характеризуються декількома числовими властивостями. Для кожної задачі помістимо значення всіх її властивостей у спеціальну структуру, звану блоком управління задачею (tcb). При цьому перелік всіх задач, що обробляються алгоритмом диспетчеризації, буде представлено масивом структур розмірності N (tcn[N]), де N – число всіх задач. Будемо вважати, что задача tcn[i] має перевагу перед задачею tcn[k], якщо задача tcn[i]

переважає задачу tcn[k] хоча б за однією властивістю (tcn[i].j > tcn[k].j), а по всім іншим не гірша за неї. Наприклад, нехай задача tcn[1] характеризується наступними властивостями: число звернень до таблиць бази даних історії - 12, час вирішення - 7 секунд, задача tcn[2] - 12 і 5 секунд відповідно. І нехай «кращою» вважається та задача, в якої число звернень до таблиць баз даних та час вирішеня менше. Тоді задача tcn[2] буде мати перевагу перед задачею tcn[1], оскільки за другою властивістю вона краща, а значення першої властивості в обох задач однакові.

Будемо вважати, що задачі tcn[i] і tcn[k] не порівнювані між собою, якщо задача tcn[i] переважає задачу tcn[k] за значеннями одних властивостей, а задача tcn[k] переважає задачу tcn[i] за значеннями інших. Наприклад, tcn[1] ={12, 7} і tcn[2] ={16, 5} не порівнювані між собою. Незалежно від того, вважається «кращим» більше значення властивості або менше, порівнювати ці задачі без додаткової інформації ми не можемо.

Відсутність вимоги лінійного впорядкування задач в списку дозволяє об'єднати деякі не порівнювані і еквівалентні задачі в одну групу і цій групі присвоїти номер (ранг), що визначає порядок виконання: чим менше номер, тим вище ранг групи задач.

В структуру, що представляє блок управління задачею, введемо набір однорозрядних полів, значеннями яких будуть булеві змінні tcn[i].p[j] (0 <= j <= N-1). Поля p, що представляють ці булеві змінні в кожній структурі, в сукупності утворюють квадратну матрицю, елементами якої є змінні tcn[i].p[j]. З визначення tcn[i].p[j] слідує, що одиниці в і-му рядку матриці визначають задачі tcn[j], по відношенню до яких tcn[i] віддається перевага. Тому якщо в j-му стовпці стоять всі нулі, то, значить, немає задачі, якій віддається перевага по відношенню до задачі tcn[j]. Розглянуті властивості задач і метод побудови матриці дозволяють здійснювати їх упорядкування. Розглянемо приклад.

Нехай у списку є дев'ять задач. Кожна задача характеризується трьома властивостями: часом вирішення, кількістю звернень до таблиць бази даних історії і часом використання каналів зв'язку для отримання оперативних даних. Значення властивостей задач наведено в табл. 1.

Таблиця 1

Властивості задач

Номер задачі	1	2	3	4	5	6	7	8	9
Властивість 1 (час зайнятості каналів зв'язку, мсек)	3	1	2	6	7	2	4	2	9
Властивість 2 (час вирішення задачі, мсек)	15	10	17	7	28	16	19	17	11
Властивість 3 (число звернень до таблиць бази даних історії)	7	21	27	4	2	22	8	23	21

Побудуємо булеву матрицю з елементами tcn[i].p[j] (табл. 2). Для цього значення властивостей задачі tcn[i] будемо віднімати зі значень відповідних властивостей задачі tcn[j].

Таблиця 2

Булева матриця для визначення рангів задач

	1	2	3	4	5	6	7	8	9
1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	1	0	0	0	1	0	1	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0	0
8	0	1	0	0	0	1	0	0	0
9	0	1	0	1	0	0	0	0	0

В табл. 2 цифри першого стовпця і першого рядка позначають номери задач. Для того щоб визначити групу задач першого рангу, достатньо знайти стовпці, в яких стоять тільки нулі. Такими стовпцями будуть 3-й, 5-й, 7-й і 9-й. Задачі з цими номерами отримають ранг 1. Ці задачі й мають бути вибрані для виконання. Яка з цих чотирьох задач має йти раніше за інших, без додаткової інформації визначити не можна, тому такі задачі повинні виконуватися в режимі квантування.

Визначимо групу 2-го рангу. Викреслимо з матриці стовпці 3, 5, 7, 9 та відповідні їм рядки. В отриманій матриці знову знайдемо стовпці, що містять лише нулі. Задачам, що відповідають цим стовпцям, присвоїмо ранг 2. Так само визначається група задач рангу 3 і т.д.

Методика диспетчеризації за структурним критерієм з урахуванням несумісності задач по використовуваних ресурсах може бути реалізована за допомогою наступного простого алгоритму.

З метою запобігання одночасного запуску на виконання несумісних між собою задач, для всіх задач, що виконуються і готові до виконання, підтримується симетрична булева матриця, кожен елемент якої визначається наступним чином: tcn[i].r[j] = 1, якщо і-та задача сумісна по використовуваних ресурсах с j-ю задачею; 0 – в іншому випадку.

Процедура вибору задач на виконання полягає в наступному. Переглядаються рядки матриці сумісності задач, помічені номерами виконуваних задач. Відмічаються всі стовпці матриці, в яких у виділених рядках знаходяться одиничні значення. Номери цих стовпців визначають задачі, сумісні з виконуваними. Серед них і вибираються задачі з найвищим рангом для виконання. При завершенні виконання якої-небудь задачі з матриці сумісності задач видаляються відповідні рядок і стовпець, і для вільних процесорів вибираються задачі відповідно до відкоригованої матриці.

Застосування веб-технологій для реалізації алгоритму диспетчеризації задач за структурним критерієм. Алгоритм диспетчеризації за структурним критерієм, представлений вище, був

використаний при розробці диспетчера задач реального часу системи SCADA пакету «Кварц-2013» [5], основні компоненти якого розроблені з використанням програмних веб - технологій.

Диспетчер задач (рис.) забезпечує функціонування в багатозадачному режимі до 32-х задач відповідно до важливості кожної задачі, часу початкового запуску і/або періоду повторення задачі та інших характеристик, які встановлюються індивідуально для кожної задачі.

Диспетчер був розроблений як веб - додаток, який створює для кожної користувацької задачі окремий потік виконання. Для цього використовувалася технологія Web Workers [4], яка дозволяє при виконанні скрипта на стороні браузера створювати спеціальний об'єкт Worker, який виконує задані функції у фоновому потоці, а не потоці інтерфейсу користувача. Паралельне виконання задач поліпшило пропускну здатність системи та дозволило краще використовувати можливості багатоядерних процесорів.

В своїй роботі задача може використовувати практично всі можливості ОС Windows і середовища програмування, використаного для розробки програми. Використовуючи OPC - сервери, в програмі можуть оброблятися практично всі технологічні дані пакету «Кварц-2013», включаючи первинні дані, які надходять від контролерів. В задачах, які функціонують під управлінням диспетчера, може використовуватися подієве управління виконанням програм, в тому числі відпрацювання тимчасових подій з точністю до часток секунди.

Кожна з задач, яка функціонує на робочій станції під управлінням диспетчера задач реального часу, описується своїм блоком управління, який зберігає значення основних параметрів (характеристик) задачі. На рис. 1 блоки управління задачами представлені вертикальними панелями з номерами задач в верхній частині панелі. Одночасно у вікні диспетчера видно блоки управління тільки п'яти задач із суміжними номерами. Горизонтальна лінійка скролінгу в нижній частині вікна диспетчера забезпечує перегляд і роботу з блоками всіх 32-х задач. На панелі блоку управління кожної задачі є наступні поля і елементи управління задачею: номер задачі, файл запуску задачі, тип задачі, час початкового запуску, період повторення задачі, дата и час чергового запуску, поточний стан задачі та кнопка управління її станом. Незаповнена (пуста) панель блоку управління задачею містить тільки номер задачі і одну кнопку "Введення", яку слід натиснути, якщо потрібно задати характеристики нової задачі, яка буде виконуватися під керуванням диспетчера з пріоритетом, рівним номеру панелі.

Одним з найбільш визначальних критеріїв для кожної задачі, що функціонує на робочій станції під управлінням диспетчера задач реального часу, є її пріоритет, який задає її важливість серед інших паралельно виконуваних задач. Пріоритет задачі є цілим числом у діапазоні від 1 до 32, чим менше число, тим вищий пріоритет задачі. Пріоритет задачі задається номером у верхній частині панелі, на якій задаються та відображаються характеристики її блоку управління. Таким чином, у вікні диспетчера задач блоки управління задачами розміщуються зліва направо відповідно до убування важливості (пріоритету) задач.

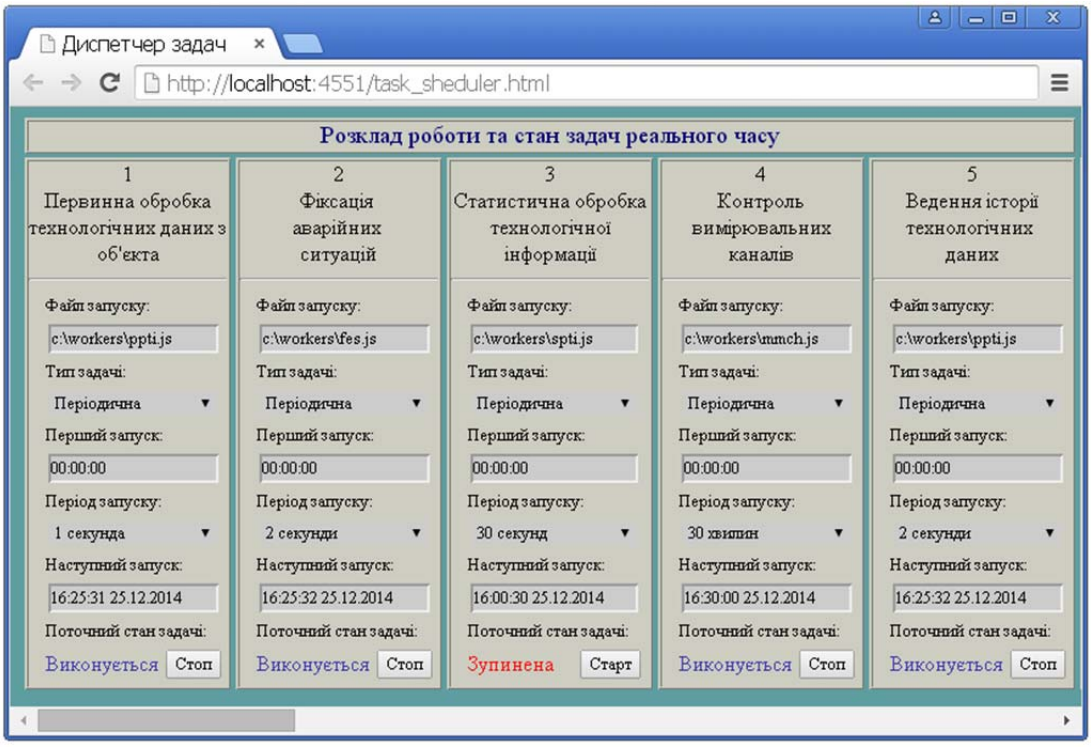

Рис. Диспетчер задач

Диспетчер дозволяє запускати на виконання як скрипти на мові JavaScript, так і виконувані файли з розширенням .exe. Створюваному для кожної задачі об'єкту Worker передається для виконання вказаний користувачем файл зі скриптом із розширенням .js, або, якщо користувач вибрав для виконання exe-файл, викликається спеціальна процедура на JavaScript, яка його запускає. Обраний користувачем програмний модуль буде запускатися на виконання на періодичній або разовій основі. Тип задачі визначає спосіб запуску програми функціонування задачі і задається методом вибору відповідного елемента зі списку, який з'являється після клацання мишею по стрілці в однойменному вікні блоку управління задачею.

Висновки. Використання веб–технологій, таких як мова гіпертекстової розмітки HTML, таблиці стилів CSS, мова програмування JavaScript, засоби відображення графічних даних на базі Canvas API, механізми веб - виконавців HTML 5 при розробці підсистеми диспетчеризації задач реального часу є ще одним підтвердженням їх ефективності для розробки підсистем верхнього рівня систем керування технологічними процесами і об'єктами. Використання цих технологій та методики диспетчеризації задач за структурним критерієм дало можливість скоротити втрати часу робочої станції на виконання задач верхнього рівня управління АСК ТП приблизно на 8 – 10 %. Це було досягнуто, в першу чергу, за рахунок зменшення непродуктивних простоїв задач в чергах очікування звільнення послідовно використовуваних ресурсів, таких як загальні таблиці баз даних технологічного процесу на диску, загальні канали зв'язку з контролерами і т. д. Використання технології Web Workers в диспетчері задач, реалізованому як веб - додаток, дозволило виконувати задачі з однаковим пріоритетом одночасно в різних потоках відповідно до розробленої дисципліни диспетчеризації.

Література

→ Adam Freeman. The Definitive Guide to HTML5 / Adam Freeman. - New York : Springer Science+Business Media, 2012. – 1053 p.

→ Web Workers. W3C Working Draft 13 March 2012 [Електронний ресурс]. - Режим доступу : http://www.w3.org/TR/2012/WD-workers-20120313/

→ Winchester C. Developing OPC HMI for a Web Browser [Електронний ресурс] / Winchester C., Pocock N. - Режим доступу : ftp://ftp.softwaretoolbox.com/ demodnld/webexes/OPC_HMI_to_web.pps

→ Рязанцев А. И. Метод планирования задач для систем экологического мониторинга / А. И. Рязанцев, Е. В. Щербаков, М. Е. Щербакова // Вісник Східноукр. нац. ун-ту імені В. Даля. - 2012. - № 15 (186). - С. 146-151.

→ Щербаков Е. В. Автоматизированное проектирование ППО КСУ на базе пакета программ "КВАРЦ" : монография / Щербаков Е. В., Щербакова М. Е., Охрамович В. К. ; [под ред. А. Г. Руденко] - Луганск : Изд-во Восточноукр. нац. ун-та имени В. Даля, 2003. - 200 с.

References

1. Adam Freeman. The Definitive Guide to HTML5 / Adam Freeman. - New York : Springer Science+Business Media, 2012. – 1053 p.
2. Web Workers. W3C Working Draft 13 March 2012. - http://www.w3.org/TR/2012/WD-workers-20120313/
3. Winchester C. Developing OPC HMI for a Web Browser / Winchester C., Pocock N. - ftp://ftp.softwaretoolbox.com/demodnld/webexes/OPC_HMI_to_web.pps
4. Ryazantsev A. I. Task schedule method for ecological monitoring systems / A. I. Ryazantsev, E. V. Shcherbakov, M. E. Shcherbakova // Announcer of the Volodymyr Dahl East Ukrainian National University - 2012. - №15 (186). - С. 146-151
5. Shcherbakov E. V. Automated design of PS CCS on the base of program package "Kvarts" : monography / Shcherbakov E. V., Shcherbakova M. EY., Okhramovich V. K. ; [edited by O. G. Rudenko]. - Luhansk : [Publishing office of East Ukrainian Volodymyr Dahl National University], 2003. - 200 p.

Рязанцев А. И., Щербаков Е. В., Щербакова М. Е. Использование веб - технологий в системах управления

Рассмотрены современные языки разметки, стилизации и программирования веб – приложений, а также проанализирована эффективность программ и визуальных интерфейсов, разработанных с их использованием. Показано, что возможностей этих языков и средств их реализации достаточно для разработки основных подсистем рабочих станций систем управления технологическими процессами. Исследованы особенности планирования и запуска задач, которые функционируют на рабочих станциях компьютерных систем управления. Для задач верхнего уровня управления разработаны методика и алгоритм диспетчеризации задач, который учитывает для каждой задачи сразу несколько показателей, например, ее важность, время решения задачи, интенсивность работы задачи с базами данных и т. п. Предложенный алгоритм диспетчеризации задач используется в диспетчере реального времени современной версии пакета „Кварц-2013”.

Ключевые слова: *HTML, CSS, JavaScript, веб - исполнители, системы управления, реальный масштаб времени, диспетчеризация.*

Ryazantsev O. I., Shcherbakov E. V., Shcherbakova M. E. The use of Web technologies in control systems.

Considered the modern languages for styling, markup and programming Web applications, as well as the efficiency of programmes and visual interfaces developed using them. Shows that the capacity of these languages and their implementation is sufficient for the development of the main subsystems of the workstations of control systems of technological processes. Researched features of planning and running tasks that run on the workstations of computer control systems. For top-level management objectives developed the method and algorithm for scheduling tasks based on each task by several indicators, for example, its importance, time challenge, intensity of work tasks with database, etc. The proposed scheduling algorithm used in real-time task manager of modern version of the package "Quartz 2013".

Keywords: *HTML, CSS, JavaScript, Web Worker, control systems, real-time, scheduling.*

Рязанцев Олександр Іванович – д.т.н., професор, завідувач кафедри комп'ютерної інженерії, Технологічний інститут Східноукраїнського національного університету імені Володимира Даля (м. Сєвєродонецьк), ryzancev@mail.ru

Щербаков Євген Васильович – к.т.н., доцент, доцент кафедри комп'ютерної інженерії, Технологічний інститут Східноукраїнського національного університету імені Володимира Даля (м. Сєвєродонецьк), gkvarc@gmail.com

Щербакова Марина Євгенівна – к.т.н., доцент, доцент кафедри комп'ютерної інженерії, Технологічний інститут Східноукраїнського національного університету імені Володимира Даля (м. Сєвєродонецьк), ms432@mail.ru

Рецензент: **Суворін О. В.** – д.т.н., доцент

Стаття подана 27.11.2014