- елемент ϕ_{yxmij}, ϕ_{yymij}, ϕ_{yzmij} належить відповідно $(n+R)s$-й, $(n+R)(s+R)$-й, $(n+R)(s+2R)$-й клітині матриці ϕ;
- елемент ϕ_{zxmij}, ϕ_{zymij}, ϕ_{zzmij} належить відповідно $(n+2R)s$-й, $(n+2R)(s+R)$-й, $(n+2R)(s+2R)$-й клітині матриці ϕ.

Повну матрицю Якобі ϕ одержимо, виконавши цю процедуру для всіх КЕ елементів.

У випадку використання лагранжевих тетраедрів 2-го, 3-го і 4-го порядків потрібно застосувати відповідні кубатурні формули чисельного інтегрування[1] і провести вивід основних залежностей за викладеною вище методикою.

Література

1. Карашецький В.П. Кубатурні формули чисельного інтегрування за об'ємом тетраедра на основі інтерполяційних повних поліномів // Науковий вісник НЛТУ України : зб. наук.-техн. праць. – Львів : РВВ НЛТУ України. – 2007. – Вип. 17.6. – С. 258-264.

Карашецкий В.П. **Расчет трехмерных вихревых магнитных полей методом конечных элементов**

Выведены основные формулы метода конечных элементов для расчета трехмерных статических вихревых магнитных полей в областях, заполненных нелинейными безгистерезисными анизотропными средами.

Ключевые слова: вихревое магнитное поле, магнитная характеристика, лагранжевий тетраэдр, метод конечных элементов, кубатурна формула.

Karashetskyy V.P. **Calculation of three-dimensional eddy magnetic fields of the method finite element**

Basic formulas of the finite element method for calculating of three-dimensional static eddy magnetic fields filled with nonlinear without hysteresis anisotropic environments were obtained.

Keywords: eddy magnetic field, magnetic characteristic, Lagrangian tetrahedron, finite element method, cubature formula.

УДК 519.718.3

Бакалавр Д.І. Кутянський; доц. М.В. Дідковська, канд. техн. наук – НТУ України "Київський політехнічний інститут"

ПОБУДОВА СТІЙКОЇ ДО ВІЗАНТІЙСЬКИХ ЗБОЇВ СИСТЕМИ

Проаналізовано різні види збоїв, що можуть виникнути в роботі програмних систем. Продемонстровано загальний підхід до побудови відмовостійкої системи та розглянуто одну з можливих задач проектування відмовостійких систем та її розв'язок.

Ключові слова: відмовостійкість, відмовостійкі системи, візантійські збої.

Вступ. З поширенням використання комп'ютерних систем у різних сферах діяльності зростають і вимоги до надійності роботи програмно-апаратних комплексів. У роботі багатьох систем необхідною умовою є відсутність збоїв, або, принаймні, утримання імовірності їх появи нижче певного рівня. При цьому часто бажаного результату не вдається досягти лише за допомогою використання надійних компонентів, а потрібно модифікувати архітектуру системи так, щоб вона була толерантною до різних видів збоїв. Наприклад, під час контролі температури певного критичного об'єкта, датчик

температури, яким би надійним він не був, рано чи пізно може вийти з ладу, або одноразово повідомити неправильну температуру, на основі чого може бути прийнято неправильне рішення щодо дій, які мають бути виконані. Для уникнення такої ситуації можна використовувати одночасно кілька датчиків для вимірювання однієї і тієї ж температури і приймати рішення на основі набору результатів з усіх датчиків – інтуїтивно це має підвищити надійність системи, оскільки ймовірність одночасного збою кількох датчиків нижча, ніж одного. Аналогічні проблеми трапляються не лише в суто технічних системах, але й в фінансових, банківських і т. ін Таким чином, задача полягає в побудові архітектури системи, здатної бути толерантною до різних видів збоїв, незалежно від прикладної області, в якій працює система.

Мета роботи. Метою цієї роботи є ознайомлення і порівняння різних типів збоїв, що можуть виникати в програмних системах та шляхів побудови систем, толерантних до цих збоїв. Демонструється одна з задач побудови відмовостійких систем та її можливий розв'язок. Також наведено можливі напрями подальшого дослідження відмовостійких систем.

Визначення та властивості відмовостійких систем

Визначення та основні підходи до реалізації. Розподілені програмні системи часто обговорюють у поняттях сервісів і їх клієнтів. Кожен сервіс надається одним або кількома серверами. Сервіс надає можливість клієнтам виконувати деякі операції, посилаючи запити сервісу. Хоча й найпростіший спосіб реалізації сервісу – це його виконання на єдиному централізованому сервері, такий підхід до проектування сервісу буде настільки ж відмовостійким, наскільки і сервер. Якщо такий рівень відмовостійкості неприйнятний, то сервіс для своєї роботи має використовувати кілька серверів, безвідмовність роботи яких не має залежати один від одного. Зазвичай для цього використовуються репліки (точні копії) єдиного сервера, які виконуються на різних комп'ютерах у межах розподіленої системи, для їх координації, комунікації між собою і з клієнтом використовуються спеціально розроблені протоколи. Фізична й електрична ізоляція серверів між собою знижують можливість одночасних відмов виконання операцій цими серверами.

Загальним підходом до реалізації відмовостійкого сервісу за допомогою реплікації серверів і координації взаємодії клієнта та реплік сервера є скінченний автомат [1]. Також за допомогою цього підходу легше зрозуміти основи проектування протоколів управління реплікацією. Скінченний автомат складається зі змінних стану, які кодують стан автомату і команд, які цей стан можуть змінювати. Виконання команд детерміністичне, тобто команда за однакових вхідних даних завжди видає один і той же результат. Команди виконуються атомарно та можуть змінювати змінні стану або подавати інформацію на вихід або і те, і інше. Клієнт під час виконання запиту вказує команду та додаткові аргументи, необхідні для її виконання.

Види збоїв. Будемо називати компонент системи збійним, якщо його поведінка не відповідає його специфікації. Розглядатимемо два принципово різних класи можливих збоїв. Перший з них – візантійський збій. Це збій, за якого один із компонентів системи у відповідь на запити проявляє довільну і, часто, зловмисну поведінку, можливо, він це робить за домовленістю з інши-

ми збійними компонентами [2]. Другий тип збою – це збій-зупинка: компонент у відповідь на появу помилки змінює свій стан таким чином, що інші компоненти можуть відстежити появу цієї помилки, і після цього зупиняє свою роботу [3]. Саме другий тип збою було б логічно називати відмовою, але цей термін закріпився за обома типами. Очевидно, що візантійський збій несе набагато гірші наслідки, ніж збій-зупинка, тому що без використання додаткових перевірок клієнт не отримує можливості перевірити коректність виконання операції компонентом. Система називатиметься відмовостійкою, якщо вона загалом продовжує коректну роботу при тому, що один або декілька її компонентів піддались збою. Також проектування систем, які базуються на якихось специфічних припущеннях щодо збою компонентів, ризиковане, оскільки ці припущення можуть не задовольнятись, тому розумно було б реалізовувати критичні системи так, щоб вони були стійкими до візантійських збоїв. Для більшості систем достатньо бути стійкими до збоїв-зупинок.

Системи, стійкі до збоїв. Систему, що складається з деякого набору компонентів, будемо називати відмовостійкою порядку t (або t-відмовостійкою), якщо вона загалом задовольняє специфікацію за умови, що не більше ніж t її компонентів відмовлять протягом деякого інтервалу часу. Відмовостійкість систем традиційно характеризується середнім часом між збоями, ймовірністю збою на заданому інтервалі часу та іншими статистичними метриками [4]. Хоча описані вище статистичні характеристики є важливими для кінцевого користувача, опис системи в термінах порядку її відмовостійкості протягом деякого інтервалу часу також має свої переваги. Зокрема якщо якась система називається t-відмовостійкою, то стають очевидними припущення щодо умов її коректного функціонування, а середній час між збоями або інші статистичні метрики такої інформації не дають. Важливою відмінністю порядку відмовостійкості від решти наведених характеристик є те, що вона не залежить від надійності компонентів, з яких складається система і, таким чином, є метрикою відмовостійкості, яка забезпечена архітектурою системи, а не використанням надійних компонентів. Тобто статистичні характеристики відмовостійкості системи можна підвищити завдяки використанню більш надійних компонентів, при цьому порядок відмовостійкості системи не зміниться. Статистичні показники відмовостійкості системи порядку t можуть бути обчислені на основі статистичних характеристик компонентів системи. При заданих показниках надійності компонентів і бажаних статистичних показниках відмовостійкості системи можна розрахувати її порядок.

Для отримання відмовостійкого автомата порядку t необхідно реплікувати один автомат і виконувати його на різних серверах у межах розподіленої системи. За умови, що кожна репліка виконуватиметься без збоїв і кожен із серверів почне свою роботу з однакового стану та виконуватиме ті ж запити користувачів у тому ж порядку, всі сервери виконуватимуть одні й ті ж дії і видаватимуть один і той же результат. Якщо якийсь з серверів або група серверів піддасться збою, то все одно клієнт має можливість отримати правильний результат операції, базуючись на роботі решти серверів.

Якщо сервери піддаються візантійським збоям, то для забезпечення відмовостійкої системи порядку t необхідна $2t+1$ репліка, тоді правильним

результатом виконання операції вважатиметься результат, отриманий більшістю з реплік. За наявності $2t+1$ репліки більшість з результатів буде правильними навіть за наявності t реплік зі збоями. Процес отримання єдиної відповіді логічно назвати голосуванням. Якщо ж сервери піддаються лише збоям-зупинкам, то для досягнення порядку відмовостійкості t потрібна $t+1$ репліка і результатом системи можна вважати результат будь-якої з реплік. Це твердження справедливе, тому що репліки видають лише правильні результати і навіть за наявності t збоїв залишиться одна робоча репліка.

Надалі сконцентруємо увагу на візантійських збоях і розглянемо задачу щодо них.

Проблема візантійських генералів

Одна з базових задач забезпечення відмовостійкості відома під назвою "Проблема візантійських генералів". Абстрактно її можна сформулювати в термінах групи візантійських генералів, яким за допомогою лише обміну повідомленнями потрібно узгодити спільний план дій. Проблема прийняття спільного рішення полягає в тому, що серед генералів можуть бути зрадники, які своїми діями намагатимуться не допустити прийняття правильного рішення рештою генералів. Тоді задача полягає у формуванні такого алгоритму, який давав би змогу відданим генералам домовитись між собою.

Віддані генерали повинні мати алгоритм, який гарантуватиме, що, по-перше, всі віддані генерали дотримуватимуться єдиного плану дій. Це означає, що всі без винятку віддані генерали виконуватимуть дії суворо відповідно до алгоритму, при цьому зрадники можуть поводитися як завгодно. Алгоритм має бути таким, що задовольняє цій умові незалежно від дій зрадників. По-друге, невелика кількість зрадників не може забезпечити того, що віддані генерали узгодять поганий план.

Розглянемо процес прийняття рішення відданими генералами. Кожен з генералів спостерігає за ворогом і передає результат своїх спостережень решті генералів. Нехай $v(i)$ – це повідомлення, яке передає i-й генерал. Після цього кожен генерал якось комбінує повідомлення $v(1)$, ..., $v(n)$ у план дій, де n – це загальна кількість генералів. Для того, щоб дотримуватись єдиного плану дій, віддані генерали мають однаково з повідомлень $v(1)$, ..., $v(n)$ виводити план дій. Наприклад, якщо треба прийняти рішення про те, нападати чи ні, то повідомленням $v(i)$ може слугувати особиста думка кожного генерала щодо нападу, а план буде прийнято, базуючись на більшості рішень. Таким чином, якщо кількість зрадників мала, то їм вдасться вплинути на прийняте рішення, лише якщо віддані генерали розділились у своїх думках приблизно порівну, але тоді жодне з остаточно прийнятих рішень не можна назвати поганим. Проте описаний процес базується на припущенні, що всі генерали отримуватимуть однаковий набір повідомлень $v(1)$, ..., $v(n)$, що може бути неправдою, оскільки зрадники можуть посилати різні повідомлення різним генералам. Для того, щоб умова щодо однаковості набору $v(1)$, ..., $v(n)$ виконувалась, потрібно, щоб кожен відданий генерал користувався одним і тим же набором $v(1)$, ..., $v(n)$ (умова 1), або, що те саме, кожні два віддані генерали мають користуватись одним і тим же значенням $v(i)$. З цієї умови випливає, що генерали не зобов'язані використовувати значення $v(i)$, отримане безпосе-

редньо від *i*-го генерала, оскільки він може виявитись зрадником. Але, вводячи таку умову, ми уможливлюємо також ситуацію, коли аналогічна недовіра може проявитись щодо повідомлення відданого генерала. Цього не можна допустити, тому що мала кількість зрадників може вплинути на результати голосування. Тому якщо *i*-й генерал відданий, тоді повідомлення, яке він відсилає, має бути використано всіма відданими генералами як значення $v(i)$ (умова 2). Оскільки обидві умови стосуються повідомлення, що надсилається одним генералом $v(i)$, то вихідну задачу можна переформулювати, розглядаючи поведінку командира, що розсилає наказ підлеглим лейтенантам. При цьому вихідні умови перетворюються у:

1. Усі віддані лейтенанти підкоряються однаковому наказу.
2. Якщо командир відданий, то всі лейтенанти підкоряються наказу, виданому генералом.

При цьому умова 1 випливає з умови 2, якщо командир відданий, що в загальному випадку не обов'язково.

Рішення задачі з використанням усних повідомлень. Покажемо тепер, що $3m+1$ генералів можуть узгодити план за наявності серед них m зрадників за допомогою обміну усними повідомленнями. Спершу треба пояснити, на яких припущеннях базується обмін повідомленнями між генералами. По-перше, кожне відіслане повідомлення досягає адресата без змін. По-друге, отримувач повідомлення знає, хто його відіслав. По-третє, відсутність повідомлення може бути виявлено. Перші два припущення не дають змогу зрадникам впливати на комунікацію інших генералів між собою, а третє не дає змогу їм відтягувати прийняття рішення відданими генералами, не посилаючи повідомлень. Алгоритм прийняття рішення передбачає, що кожен з генералів має змогу надсилати повідомлення напряму адресатам. Якщо генерал – зрадник, то він може взагалі не відсилати повідомлень жодному лейтенанту, в такому разі вважатимемо, що вони всі мають відступити.

Визначимо алгоритм обміну усними повідомленнями $УП(m)$, для всіх невід'ємних m, за допомогою якого генерал надсилає повідомлення n-1 лейтенанту. Покажемо, що алгоритм $УП(m)$, де m – це кількість зрадників, вирішує проблему генералів, якщо їхня кількість $n>3m$. Для доведення нам знадобиться також функція *majority* $(v_1, \ldots v_n)$, яка дорівнюватиме v за умови, що більшість з v_i, які слугують її аргументами, дорівнюють v (якщо бути більш точним, то потрібна не одна функція, а сімейство функцій, для кожного з можливих n). Алгоритм будуємо за індукцією. База індукції $УП(0)$:

1. Генерал надсилає повідомлення кожному з лейтенантів.
2. Кожен лейтенант використовує значення, отримане в повідомленні, або значення "відступити", якщо він не отримав повідомлення.

Припущення індукції: нехай існує алгоритм обміну повідомленнями $УП(m_1), m_1 > 0$. Крок індукції, для $УП(m)$, $m=m_1+1$:

1. Генерал надсилає повідомлення кожному з лейтенантів.
2. Для кожного i нехай v_i – це повідомлення, отримане i-м лейтенантом. Лейтенант i поводить себе як командир в алгоритмі $УП(m-1)$, розсилаючи повідомлення v_i решті $n-2$ лейтенантам.

3. Для кожного i та j таких, що $i \neq j$, нехай v_j буде повідомленням, яке лейтенант i отримує від лейтенанта j на другому кроці цього алгоритму або значення "відступити", якщо відповідне повідомлення лейтенант не отримав. При цьому лейтенант i буде використовувати значення $majority(v_1,...,v_{n-1})$.

Лема. Для довільних m і k алгоритм *УП* (m) задовольняє умову 2, якщо кількість генералів більша ніж $2k+m$ і серед них не більше k зрадників.

Доведення. Проведемо доведення індукцією по m. Умова 2 розглядає лише випадок, коли командир не зрадник. Оскільки кожне відіслане повідомлення отримує адресат без змін, то алгоритм *УП*(0) працює, якщо командир відданий, тому лема правдива для $m=0$, а це і є база індукції. Припустимо, що умова леми виконується для $m-1$, де $m>0$ і доведемо її для m. На першому кроці відданий командуючий відсилає повідомлення v усім $n-1$ лейтенантам. На другому кроці кожен з відданих лейтенантів виконують дію згідно алгоритму *УП*(m) з $n-1$ генералом. Оскільки за гіпотезою $n>2k+m$, то $(n-1)>2k+(m-1)$, тобто можна застосувати припущення індукції, і отримати, що всі віддані лейтенанти отримають $v_j=v$ від кожного відданого Лейтенанта j. Оскільки зрадників за умовою не більше, ніж k, а також $(n-1)>2k+(m-1)\geq 2k$, то більшість з $n-1$ генералів віддані. А оскільки кожен відданий генерал користується значенням $v_i=v$ для більшості з $n-1$ різних i, то їм доступна функція $majority(v_1,...,v_n\text{-}1)=v$ на кроці 3, що доводить умову 2.

Теорема. Для довільного m алгоритм *УП*(m) задовольняє умовам 1 і 2, якщо кількість генералів більша за $3m$, а зрадників серед них не більше, ніж m.

Доведення. Проведемо індукцією за m. Якщо зрадників немає, то очевидно алгоритм задовольняє обом умовам, тому припустимо, що алгоритм працює для $m-1$ і доведемо, що він працює для m.

Спочатку розглянемо випадок, коли командир відданий. Беручи $k=m$ в лемі отримуємо, що *УП*(m) задовольняє другій умові, а оскільки командир відданий, то автоматично і першій. Залишається розглянути ситуацію, коли командир зрадник.

Оскільки всього зрадників не більше, ніж m, і при цьому один з них командир, то серед лейтенантів є не більше, ніж $m-1$ зрадник. Генералів всього більше, ніж m, тобто лейтенантів більше, ніж $3m-1$, а $3m-1>3(m-1)$, тому ми можемо застосувати припущення індукції для *УП*($m-1$). Таким чином, для кожного j будь-які два віддані лейтенанти отримують однакове значення v_j на третьому кроці алгоритму (це випливає з умови 2, якщо один з них і є лейтенант j і з умови 1 в іншому випадку). А оскільки кожен з відданих лейтенантів отримує один і той же вектор $v_1,...,v_{n-1}$, то і значення функції $majority(v_1,...,v_{n-1})$ на третьому кроці алгоритму, що доводить умову 1.

Аналіз результатів задачі. Як бачимо, для досягнення відмовостійкості за умов описаної проблеми візантійських генералів необхідна, по-перше, значна надлишковість кількості генералів та, по-друге, велика кількість повідомлень, якими вони повинні обмінятись для досягнення згоди. Існують також інші вхідні обмеження, які можуть траплятись у прикладних задачах, для них і алгоритм, і потреба в надлишковості та кількості повідомлень можуть відрізнятись. Наприклад, якщо припустити, що генерали володіють механізмом перевірки особи відправника повідомлення та цілісності дос-

тавленої інформації, то можна значно скоротити кількість раундів обміну повідомленнями.

Висновки. Відмовостійкі системи – це системи, що дають змогу їх клієнтам отримати бажаний результат, незважаючи на те, що в середині системи можуть відбуватись збої. У роботі наведено класифікацію і порівняння можливих збоїв та проблем прийняття рішення, пов'язаних з різними типами збоїв. Показано, що для систем, які мають демонструвати більшу міру відмовостійкості, потрібно зазвичай використовувати значну надлишковість компонентів, і не можна робити жодних специфічних припущень щодо характеру збоїв, які можуть відбутись, без модифікації алгоритму.

Подальші дослідження можуть охоплювати: скорочення надлишковості компонентів та кількості повідомлень за умови додаткових обмежень на роботу відмовостійкої системи, дослідження можливості розв'язання прикладних задач за допомогою запропонованого алгоритму за прийнятний час, дослідження продуктивності систем за наявності збоїв.

Література

1. Schneider, F.B. 1990. Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial. ACM Computing Surveys 22, 4 (December). – P. 299-319.

2. Lamport, L., Shostak, R., and Pease, M. 1982. The Byzantine generals' problem. ACM TOPLAS 4, 3 (July). – P. 382-401.

3. Schneider, F.B. 1984. Byzantine generals in action: Implementing fail-stop processors. ACM TOCS 2, 2 (May). – P. 145-154.

4. Siewiorek, D.P., and Swarz, R.S. 1982. The Theory and Practice of Reliable System Design. Digital Press, Bedford, Mass. – P. 55-64.

Кутянский Д.И., Дидковская М.В. **Построение стойкой к византийским сбоям системы**

Проведен анализ разных видов сбоев, которые могут возникнуть при работе программных систем. Продемонстрирован общий подход к построению отказоустойчивой системы и рассмотрена одна из возможных задач проектирования отказоустойчивых систем и ее решение.

Ключевые слова: отказоустойчивость, отказоустойчивые системы, византийские сбои.

Kutianskyi D.I., Didkovska M.V. **Design of Byzantine fault-tolerant system**

This work analyses different types of faults which can occur in programmatic systems. General approach of design of fault-tolerant system is described. In addition, one of design problems with its solution is analyzed.

Keywords: fault-tolerance, fault-tolerant systems, Byzantine faults.

УДК 331.5.024.54:004

Доц. Г.В. Прошак, канд. екон. наук – Львівський НУ ім. Івана Франка

ІНФОРМАЦІЙНА СИСТЕМА УПРАВЛІННЯ ЛЮДСЬКИМ КАПІТАЛОМ І ЇЇ ВИКОРИСТАННЯ В ДЕРЖАВНІЙ СЛУЖБІ ЗАЙНЯТОСТІ

Розглянуто поняття "людський капітал", його види. Проаналізовано роботу державної служби зайнятості щодо залучення до економічної діяльності людського капіталу незайнятого населення. Досліджено особливості інформаційної системи уп-