УДК 621.039

В.С. ХАРЧЕНКО

Національний аерокосмічний університет ім. М.Є. Жуковського «ХАІ», Україна

ГАРАНТОЗДАТНІ СИСТЕМИ ТА БАГАТОВЕРСІЙНІ ОБЧИСЛЕННЯ: АСПЕКТИ ЕВОЛЮЦІЇ

Запропоновано модифіковану таксономічну схему гарантоздатності з урахуванням факторів еволюції комп'ютерних систем (зміни функціональних вимог, вимог до гарантоздатності та характеристик середовища), а також операційного циклу і рівнів забезпечення відмовостійкості. Узагальнено таксономію багатоверсійних обчислень в гарантоздатних системах і проаналізовано методи її оцінки. Визначено поняття самоеволіціонуючої системи. Проаналізовано деякі еволюційні аспекти технологій, що використовуються при побудові гарантоздатних комп'ютерних систем.

Ключові слова: *гарантоздатність, багатоверсійні обчислення, таксономія, операційний цикл відмовостійкості, еволюція, самоеволюціонуючі системи*

Вступ. Проблема гарантоздатності в контексті еволюції

Гарантоздатність та еволюція. У 2004 році у першому числі журналу «Dependable and Secure Computing» було опубліковано роботу A. Avizienis, J.-C. Laprie, B. Randell та C. Landwehr, присвячену систематизації базових понять та таксономії гарантоздатних (надійних та безпечних) обчислень [1]. Ця стаття стала однією з найбільш цитованих у наукових публікаціях спеціалістів відповідної предметної області. Гарантоздатність є складною властивістю системи надавати задані послуги (виконувати функції), яким можна виправдано довіряти; гарантоздатною є система, якій притаманна така властивість. За ці роки концепція гарантоздатності знайшла певного розвитку як ключова у галузі критичних (бізнес-критичних) комп'ютерних систем (КС).

Природно, що оцінка та забезпечення гарантоздатності є за своєю суттю комплексною проблемою, оскільки:

- по-перше, гарантоздатність – це складна властивість, яка включає безвідмовність, готовність, здатність до обслуговування, достовірність, функціональну та інформаційну (конфіденціальність і цілісність) безпеку, а також, за певних умов, живучість [1,2]; забезпечення необхідного рівня показників за кожною з них впливає на інші;

- по-друге, КС є комплексом апаратних, програмних і мережевих компонент, працездатність яких може бути порушена внаслідок фізичних, проектних дефектів і дефектів взаємодії (з зовнішнім середовищем); зрозуміло, що необхідно ураховувати вплив кожної з цих компонент з урахуванням різних дефектів і різних принципів підвищення стійкості до них;

- по-третє, поведінку і забезпечення необхідних характеристик гарантоздатності слід розглядати у більш широкому контексті – контексті еволюції КС та її середовища; йдеться про складні системи та інфраструктури («системи систем»), умови використання яких можуть змінюватися впродовж їх життєвого циклу.

Слід відзначити, що аспекти еволюції та стійкості до її факторів досить інтенсивно аналізуються й досліджуються в останні роки. Це знайшло своє відбиття у таких поняттях як «evolvability» та «evolvable system» (тобто «здатність до еволюції» та «еволюціонуюча система» або «система, здатна до еволюції»), а також «resilience» і «resilient system» (дослівно «здатність до супротиву» і «системи, здатні до супротиву») [3].

Проблема еволюції та еволюціонуючих систем не є новою і розглядалася раніше у різних контекстах багатьма дослідниками [4]. Відповідно до [5] еволюцією є зміна наслідуваних ознак популяції організмів на протязі кількох поколінь. На відміну від біологічних організмів чіткого визначення поняття еволюції для технічних об'єктів немає, хоча терміни «еволюція систем», «еволюція технологій» та інші вживаються досить часто.

Це стосується, перш за все, комп'ютерних технологій та систем, оскільки їх зміни проходять дуже динамічно і ознаки «популяцій» змінюються впродовж не десятиліть, а років або навіть місяців. Більш того, такі більш динамічні зміни можуть мати місце для однієї працюючої системи. Найпростішим прикладом такої еволюції є «м'яке» оновлення web-сервісів за допомогою спеціальної платформи, що надає користувачу можливість безперервно отримувати послуги, якість яких (обсяг послуг, надійність,

оперативність) поступово зростає [6].

На нашу думку, цікаво було б більш детально проаналізувати аспект еволюції у контексті гарантоздатності та принципів її забезпечення, зокрема, відмовостійкості та багатоверсійності. До того ж, багатоверсійність і похідні від неї поняття, у свою чергу, потребують подальшого розвитку таксономічної схеми, описаної у [7], де зроблено узагальнення ключових понять, та аналізу еволюційних змін технологій, що безпосередньо впливають на гарантоздатність комп'ютерної системи та інфраструктури (системи систем) [8].

Мета та структура роботи. Відповідно до цього, *метою* даної роботи є розвиток таксономічних схем гарантоздатності та багатоверсійності з урахуванням факторів еволюції. Структурно вона складається з чотирьох розділів: у *першому* з них пропонується таксономічна схема гарантоздатності, що узагальнює виклики, обумовлені несправностями та змінами вимог і умов використання; *другий* – присвячено аналізу операційного циклу та рівнів забезпечення відмовостійкості та стійкості до факторів еволюції; у *третьому* – деталізовано елементи та розвинуто таксономію багатоверсійних обчислень, а також надано огляд проблем оцінювання та вибору багатоверсійних технологій у контексті факторів еволюції; останній розділ пов'язаний з аналізом деяких закономірностей еволюційних змін технологій забезпечення гарантоздатності. У висновках сформульовано напрями подальших досліджень гарантоздатних та здатних до еволюції комп'ютерних систем.

1. Таксономія гарантоздатності та здатності до еволюції

1.1. Паталогічні ланцюжки й фактори еволюції

Дефекти, помилки, відмови. Основними елементами таксономічної схеми гарантоздатності є загрози F (дефекти, помилки та відмови, уразливості та втручання), механізми захисту від них T (відмовостійкість і відмовобезпека) , а також первинні P_1 та вторинні P_2 властивості (рис.1). Фізичні дефекти або несправності апаратних засобів (physical fault, f_p) призводять до порушень або помилок обчислювального процесу (error, e_p), які визначають відповідну подію для системи – збій або відмову (failure, F_p) і перехід у непрацездатний (частково непрацездатний) стан, тобто маємо *патологічний ланцюжок*: $f_p \rightarrow e_p \rightarrow F_p$.

Аналогічні ланцюжки існують для проектних дефектів (design fault, f_d) і дефектів взаємодії (interaction fault, $f_a = \{f_{ap}, f_{ai}\}$) внаслідок фізичних (f_{ap}) і інформаційних (f_{ai}) впливів, тобто: $f_d \rightarrow e_d \rightarrow F_d$ та $f_a \rightarrow e_a \rightarrow F_a$ ($f_{ap} \rightarrow e_{ap} \rightarrow F_{ap}$ або $f_{ai} \rightarrow e_{ai} \rightarrow F_{ai}$).

Слід зазначити, що внаслідок інформаційних впливів неприпустимо змінюється інформаційно-технічний стан системи, коли або порушується цілісність інформації, або зовнішнє середовище (інша система) несанкціановано отримує доступ до неї. Такий доступ можливий через наявність уразливостей компонент, які є специфічним видом дефектів (фактично це не дефект, а передумова порушення, яке матиме місце як помилка e_{ai} тільки за умови дії зовнішнього втручання).

На базі означених f-e-F-ланцюжків може бути побудована спеціальна нотація для аналізу подій та моделювання поведінки КС. Вона є модифікацією Occurence Nets (Causal Nets або Occurence Graphs) і була запропонована B. Randell для фізичних і проектних несправностей та отримала назву Structured Occurence Nets.

Зміни вимог та середовища. Крім несправностей та обумовлених ними порушень технічного або інформаційно-технічного стану, можливі інші виклики, пов'язані з факторами еволюції. До них слід віднести:

- *зміни вимог до системи* (функціональних вимог $\{r_{fq}\}$, $q = 1,\ldots, n_f$ або вимог до складових гарантоздатності $\{r_{cw}\}$, $w = 1,\ldots, n_d$), які повинна реалізувати КС;
- *зміни характеристик зовнішнього середовища* $\{r_{ej}\}$, $j = 1,\ldots, n_e$, які вона повинна ураховувати. Множину R, що об'єднує множини $\{r_{fq}\}$, $\{r_{cw}\}$ і $\{r_{ej}\}$, будемо називати факторами еволюції.

За аналогією з f-e-F-ланцюжками можна побудувати відповідну послідовність для факторів еволюції або *еволюційні ланцюжки*. Внаслідок появи нової (зміни) вимоги $r_h \in R$, $R = \{r_{fq}\} \cup \{r_{cw}\} \cup \{r_{ej}\}$, в системі здійснюється неспецифікована за попередніми станами зміна інформації (назвемо її еволюційною помилкою, u) і система перейде у неспецифікований стан (цей стан не може розглядатися як еволюційна відмова, D).

Відповідно, маємо r-u-D-ланцюжки, які описують поведінку системи за умов дії факторів еволюції. Таким чином, у загальному випадку слід розглядати три пари множин fr = {f,r}, eu = {e,u}, FD = {F,D} та послідовність $fr \rightarrow eu \rightarrow FD$.

1.2. Узагальнена таксономічна схема

Урахування R-факторів. З урахуванням розширення множини викликів, елементам $r_h \in R$ повинні бути поставлені у відповідність додаткові елементи таксономічної схеми. Стани, толерувальні механізми, первинні та вторинні властивості, пов'язані з дією R-факторів, позначені на рис.1 як S_R, T_R, P_{1R}, P_{2R} відповідно і показані у нижній частині штриховими лініями.

Механізму протидії та системі, де він реалізується (як можна зробити висновок з аналізу відомих публікацій, обговорень на спеціальних робочих зустрічах та конференціях [3]), в англомовній літературі відповідають терміни «resilience» і «resilient system». Їх переклад не є однозначним, оскільки можливі варіанти: «стійкість» і «стійка система», або «еластичність» та «еластична система», або «здатність до супротиву (опору)» і «система, здатна до супротиву (опору)». Перша пара є вельми загальною, друга – дещо штучною, третя – громіздкою і обмеженою за змістом.

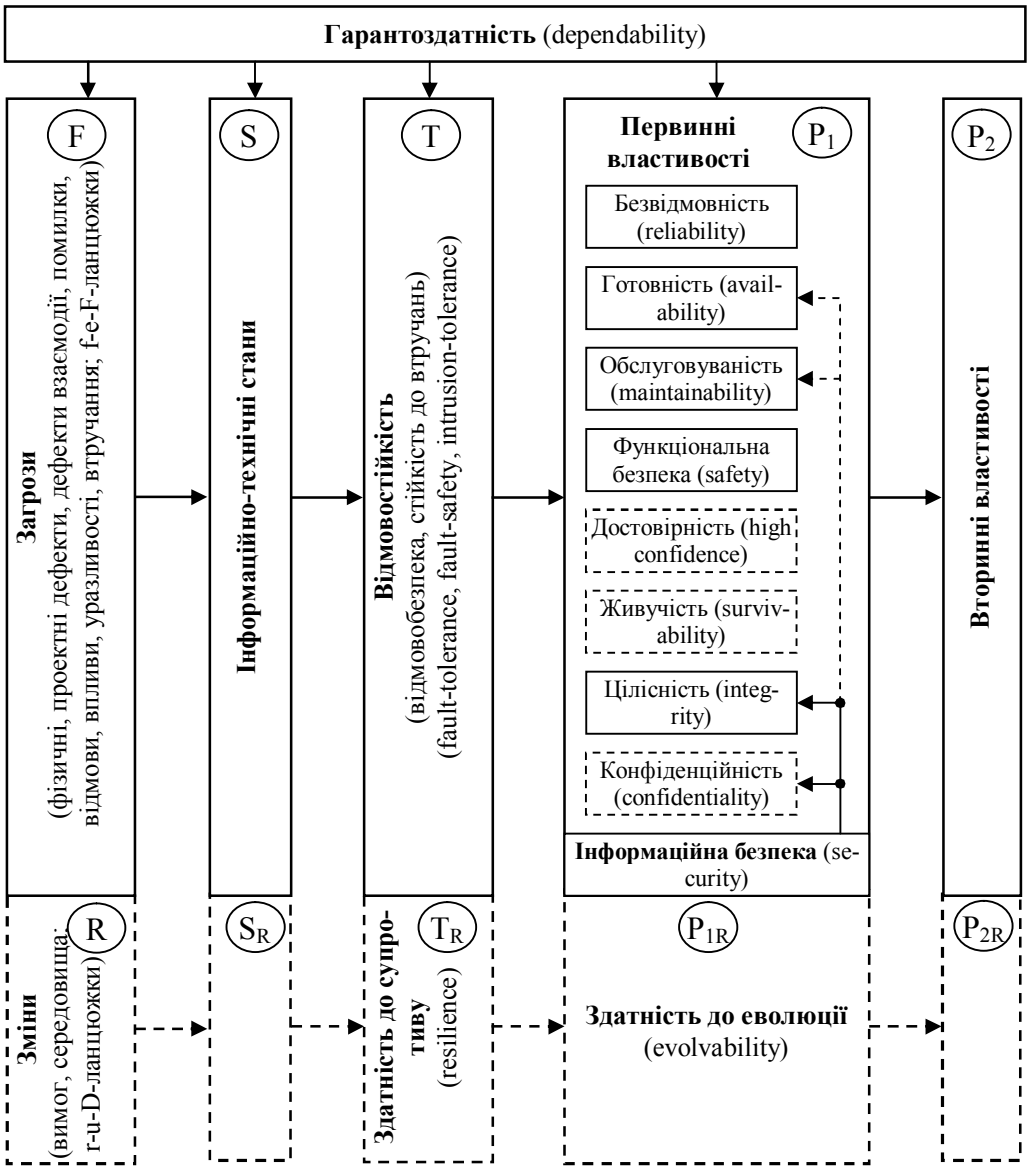

Рис.1. Таксономічна схема гарантоздатності з урахуванням факторів еволюції

Будемо умовно називати такий механізм *R-механізмом* або *R-стійкістю*, а систему, у якій він реалізується – R-системою. Завдяки такому механізму маємо первинну властивість – «*здатність до еволюції*» або «еволюційність» («evolvability») та систему, якій притаманна така властивість, тобто «*систему, здатну до еволюції*» або «еволюціонуючу систему» («evolvable system»).

Другі терміни у парах не досить точно відбивають сутність понять, оскільки необхідно підкреслити, що йдеться не про звичайну здатність системи до модернізації, яка є однією з традиційних складових якості, а про її спроможність динамічно адаптуватися до означених змін на протязі тривалого часу без перерв або з перервами, величина яких не може бути більшою за наперед визначене припустиме значення. в процесі еволюції.

Таким чином, таксономічна схема гарантоздатності, з урахуванням еволюційного аспекту, має *додатковий ланцюжок*: R-фактори – R-стійкість

(здатність до супротиву, парирування наслідків змін, в додаток до відмовостійкості) – первинна властивість (здатність до еволюції, в додаток до первинних властивостей гарантоздатності). Вторинними властивостями здатності до еволюції можуть бути пристосованість до прогнозування, виявлення та усунення наслідків еволюційних змін, які аналогічні вторинним властивостям здатності до обслуговування.

Інформаційно-технічний стан та його порушення. Додатковим елементом таксономічної схеми, на нашу думку, повинен стати *інформаційно-технічний стан* (ІТС) S, який об'єднує інформаційну і технічну складові порушень працездатності [9]. Дефекти та несправності f_p, f_d і f_{ap} призводять до порушення працездатного технічного стану S_t, а f_{ai} (і можливо, f_d) – до порушення працездатного інформаційної складової S_i ІТС. Зміни функціональних вимог r_f і вимог до гарантоздатності r_c, а також зміни характеристики середовища r_e також можуть призвести до *порушення інформаційно-технічного стану*, якщо система не адаптована до таких змін. Їх оперативне урахування потребує корегування множини справних, працездатних, частково працездатних, непрацездатних (безпечних і небезпечних) інформаційної та технічної складової ІТС, оскільки при цьому може змінюватися множина функціональних станів системи S_z.

Відповідність викликів (загроз, обумовлених дефектами, і змін вимог та середовища) та складових ІТС, які можуть бути порушені внаслідок їх появи, показано у табл.1.

Таблиця 1

Відповідність викликів та порушень ІТС

Виклики		Порушення ІТС		Нові функціональні стани, S_z
Загрози	Зміни	S_t	S_i	
f_p		+		
f_d		+	+	
f_{ap}		+		
f_{ai}			+	
	r_f	+	+	+
	r_c	+	+	+
	r_e	+	+	+

1.3. Системи зі змінними параметрами

КС, у яких здійснюється відновлення при відмовах компонент внаслідок дефектів f_p, f_d і f_a, фактично є *системами зі змінними параметрами*, оскільки [10,11]:

- усунення дефектів програмних компонент f_d (без внесення нових) приводить до зменшення інтенсивності прояву дефектів, що залишилися, тобто до зменшення інтенсивності відмов; якщо при усуненні вносяться нові дефекти, то динаміка зміни буде більш складною;

- після усунення проектного дефекту змінюється інтенсивність відновлення, оскільки може збільшуватися час на пошук та усунення дефекту;

- при виявленні та усуненні уразливостей («накладання патчів») зменшується інтенсивність відмов внаслідок дефектів взаємодії f_{ai};

- заміна апаратних компонент при відмовах внаслідок f_p приводить фактично до зміни параметру потоку відмов, оскільки замінюється лише той компонент, що відмовив.

Для гарантоздатних систем, у яких використовуються різні види надмірності, кількість та складність факторів, що призводить до зміни параметрів, зростають. Отже, такі системи є системами зі змінними параметрами. Вони можуть розглядатися як частковий випадок (певна модель) еволюціонуючих систем. Зміна параметрів може проходити також внаслідок зміни вимог до гарантоздатності r_c, коли дія фактору еволюції буде більш цілеспрямованою.

2. Відмовостійкість та R-стійкість

2.1. Операційні цикли та ієрархія стійкості

Операційні цикли для різних несправностей. Відмовостійкість (fault-tolerance) характеризується *операційним циклом* $\Omega_F = \{\omega_{Fk}\}$, описаним у [9], що складається з операцій прогнозування ω_{F1}, попередження ω_{F2}, виявлення ω_{F3}, локалізації ω_{F4}, ізоляції ω_{F5}, парирування ω_{F6} відмов, а також реконфігурації структури ω_{F7} і відновлення інформації ω_{F8}. Послідовність та часові межі операцій можуть змінюватися з урахуванням специфіки дефектів і самої КС.

Аналогічні цикли мають похідні відмовостійкості – відмовобезпека (fault-safety), стійкість до зовнішніх втручань (intrusion-tolerance) та інш. Зрозуміло, що операційний цикл Ω_F (операції ω_{Fk}) може декомпозуватися для різних дефектів (f-e-F-ланцюжків) за типами та тяжкістю наслідків.

Наприклад, для уразливостей програмних компонент (підвиду дефектів f_{ai}) маємо таке:

- ω_{Fai1} – прогнозування можливих характеристик атак на цю уразливість (імовірність, спосіб, часові параметри);

- ω_{Fai2} – попередження втручання шляхом активізації відповідних засобів або накладання патчів для скасування цієї уразливості;

- ω_{Fai3} – детектування атаки через контроль вхідних даних;

- ω_{Fai4} (ω_{Fai5}) – локалізація (ізоляція) компоненти, на яку здійснено атаку (завдяки уразливості якої вона здійснюється);

- ω_{Fai7} – реконфігурація структури, яка забезпечує парирування ω_{Fai6} (наприклад, шляхом мажоритування) можливої відмови внаслідок атаки на уразливість; вона може здійснюватися шляхом динамічної перебудови з використанням диверсних компонент [12];

- ω_{Fai8} – відновлення надання послуг (продовження виконання наступних операцій сервісу) та вибір стаціонарної конфігурації.

Ієрархія рівнів забезпечення відмовостійкості. Слід підкреслити, що операційні цикли відмовостійкості можуть реалізовуватися на різних рівнях ієрархії систем і для різних систем. Це стосується як систем на кристалі (system-on-chip), так і потужних ІТ-інфраструктур. Подальший розвиток програмованої логіки пов'язується з концепцією природно надійних (гарантоздатних) кристалів, яка успадковує ідеї природної надійності комп'ютерів завдяки новим технологічним можливостям.

Наприклад, для ПЛІС можливими рівнями реалізації процедур забезпечення відмовостійкості є такі:

1) *нанорівень*, на якому використовуються молекулярні схеми надмірних структур, що дозволяють парирувати частку дефектів на початковій стадії;

2) *рівень логічних комірок*, на якому може бути застосовані надмірні логічні базиси (базиси, толерантні до «відмов» логічних змінних);

3) *рівень бібліотечних елементів*, з яких формується проект системи; така бібліотека включає елементи з вбудованим контролем або зі структурною надмірністю; використання таких елементів чи можливостей автоматичного мажоритування регістрових схем при проектуванні надається виробниками ПЛІС вже зараз;

4) *внутрішньо кристальний рівень*, на якому реалізуються різні варіанти відмовостійких структур; їх вибір та завантаження здійснюються відповідно характеристик кристалу, кількості, типу та розміщення комірок та зв'язків, що відмовили [13];

5) *зовнішньо кристальний рівень*, де реалізуються традиційні засоби відмовостійкості для надвеликих інтегральних схем з урахуванням специфіки ПЛІС і різних видів надмірності: структурної, версійної, часової та інш. [14].

Для ІТ-інфраструктур відмовостійкість забезпечується з рівня окремих компонент до рівня геокластерів. Зрозуміло, що ресурси відмовостійкості на різних рівнях слід розглядати як загальний «вертикально інтегрований» ресурс. Це надає можливість оптимізувати витрати на забезпечення гарантоздатності для системи в цілому.

2.2. Операційний цикл R-стійкості

За аналогією з відмовостійкістю може бути сформований *операційний цикл R-стійкості*, $\Omega_R = \{\omega_{Ry}\}$, який складається з операцій:

- *прогнозування змін* вимог до функціональності ω_{Rf1}, гарантоздатності ω_{Rc1} та характеристик середовища ω_{Re1};

- *попередження змін* ω_{Rf2}, ω_{Rc2}, ω_{Re2}; ця підмножина операцій може бути пустою, оскільки зовнішні зміни, скоріш за все, є об'єктивними і не залежать від самої системи; виключення становить операція ω_{Re2}, яка може полягати у формуванні спеціальних впливів на середовище, які, у разі необхідності, попередять зміни або оптимізують його характеристики;

- *визначення* ω_{Rf3}, ω_{Rc3}, ω_{Re3} та ідентифікації ω_{Rf4}, ω_{Rc4}, ω_{Re4} *змін*; операції ω_{Rf3}, ω_{Rc3} та ω_{Rf4}, ω_{Rc4} можуть реалізовуватися шляхом введення додаткового каналу специфікованої інформації, операції ω_{Re3}, ω_{Re4} – шляхом спеціальних засобів контролю та ідентифікації параметрів зовнішнього середовища;

- *компенсації* (*парирування*) *змін*, яка полягає у відповідній перебудові системи задля їх урахування шляхом реконфігурації структури (операції ω_{Rf5}, ω_{Rc5}, ω_{Re5}, які аналогічні операції ω_{F7}) і переходу до продовження функціонування у нових умовах (операції ω_{Rf6}, ω_{Rc6}, ω_{Re6}, які аналогічні операції відновлення інформації ω_{F8}).

Зрозуміло, операції множини Ω_R мають не тільки певну аналогію з операціями множини Ω_F, але й можуть частково поєднуватися з ними шляхом використання спільних засобів. Це стосується, перш за все, операцій ω_{Re}, якщо йдеться про пристосування системи до змін характеристик зовнішнього середовища.

2.3. Гарантоздатність і самоеволюціонучі системи

Автономні та самоеволюціонучі системи. Останніми роками інтенсифікувалися дослідження, пов'язані зі створенням автономних і самокерованих систем (Autonomous and Autonomic Systems). Разом з цим набули поширення публікації, ключовими у яких є терміни «самоуправління» (self-management), «самоадаптація» (self-adaptation), «саморепродукція або самовідтворення» (self-reproduction), «самооптимізація» (self-optimization) та інші [15]. Їх загальна частина «само» для автономних систем, з одного боку, є природною, оскільки такі системи за визначенням повинні вирішувати всі задачі самостійно, без зовнішньої підтримки.

З іншого боку, повинен бути визначений якіс-

ний та кількісний рівені автономності таких механізмів, витрати та додаткові ризики, пов'язані з їх реалізацію.

Що стосується гарантоздатності, то можна говорити про відомі механізми самоконтролю (self-checking), самодіагностування (self-diagnostic), самовідновлення або саморемонт (self-recovery або self-repairing) та ін. Вони об'єднуються у багатьох роботах терміном «самолікування» (self-healing). При застосуванні таких термінів для складової «само» слід визначити:

- *приріст відповідного показника* (наприклад, достовірності контролю та діагностування, що характеризується ймовірностями помилок першого-третього роду по різних дефектах);

- *складність і безвідмовність додаткових засобів* (наприклад, вбудованих засобів контролю та діагностування; зрозуміло, що у цьому випадку система ускладнюється та погіршується її безвідмовність);

- *вплив на часові характеристики* (додаткові затримки внаслідок вбудованих засобів) та інш.

В контексті R-стійкості слід зазначити, що для автономних систем забезпечення еволюційних механізмів ускладнюється, оскільки може базуватися тільки на використанні внутрішніх ресурсів, тобто така система повинна мати деякий запас здатності до еволюції (здатності до урахування зміни вимог і характеристик зовнішнього середовища), яка ініціюється самою системою.

Тоді доречним може бути використання терміну «*самоеволюціонуюча система*» (self-evolvable system).

Самоеволюціонучі системи із зовнішніми ресурсами. У загальному випадку самоеволюціонуча система для динамічного реагування на дію R-факторів використовує не тільки власні ресурси, але й ресурси середовища (інших систем). При цьому можуть задіюватися *зовнішні компоненти та зв'язки*, які є загальним ресурсом для різних систем.

Прикладом таких систем є сервіс-орієнтовані web-системи WS, які побудовані за схемою (рис.2) «користувач (C) – проміжна платформа або middleware (MW) – цільові сервіси (TS)» [16]. Цільові сервіси у такій системі утворюють:

- вертикальні композиції-кортежі для нарощування функціональності, тобто вибудовування ланцюжків із цільових сервісів, які формують більш складний композитний необхідний вертикально інтегрований сервіс $SV = \langle Ts_{1h},\ldots,Ts_{vh},\ldots,Ts_{Vh}\rangle$, $h = 1,\ldots,H$ (замовлення квітків-бронювання готелю-екскурсійна програма …);

- горизонтально інтегровані підсистеми $SH = \{Ts_{vh},\ldots,Ts_{vh},\ldots,Ts_{vH}\}$, $v = 1,\ldots,V$, з цільових (диверсних) сервісів зі співпадаючою функціональністю або спеціально клонованих сервісів, що забезпечують необхідний початковий рівень гарантоздатності.

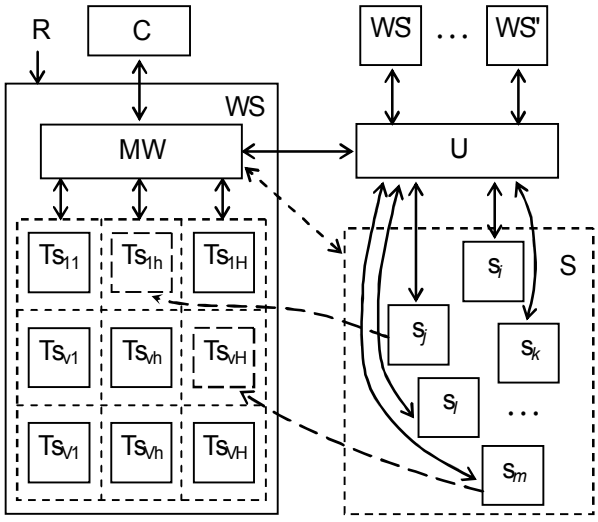

Рис. 2. Структура самоеволюціонуючої сервіс-орієнованої web-системи

Середовищем системи WS (та подібних систем WS',…, WS'') є несистематизована множина різних сервісів $S = \{s_i,s_j,\ldots, s_m\}$, доступних через Інтернет або безпосередньо, або з використанням спеціальної підсистеми U, яка забезпечує прискорений пошук необхідних сервісів і надає інформацію щодо їх характеристик (функціональності, продуктивності, гарантоздатності) завдяки проведенню періодичного моніторингу цих компонент шляхом тестування або фіксації результатів запитів від інших систем (користувачів). Такі функції зараз частково виконує система UDDI [17].

Еволюційні зміни системи WS можуть проходити у такий спосіб:

- при зміні вимог до функціональності r_f підсистема MW через систему U або безпосередньо інкорпорує необхідний сервіс $s_k \in S$ (необхідні сервіси $\Delta S_f \subset S$) для розширення структури відповідного вертикально-інтегрованого сервісу SV_h або утворення нового;

- при зміні вимог до гарантоздатності r_c аналогічно інкорпорується сервіс $s_m \in S$ (необхідні сервіси $\Delta S_c \subset S$) для корегування структури відповідного горизонтально-інтегрованого сервісу SH_v шляхом заміщення негарантоздатного (недостатньо надійного, недостатньо безпечного,…) сервісу або включення до відповідної множини додаткового сервісу для забезпечення нових вимог;

- при зміні характеристик середовища r_{ej} можуть реалізовуватися аналогічні операції або додатково перебудовуватися підсистема MW.

Система WS може конкурувати за ресурси

(сервіси множини S) з системами WS',…, WS''. Але ресурси Інтернет-середовища постійно поповнюються за рахунок природної появи нових сервісів або розширюються за рахунок штучно створюваних (клонованих) сервісів, які розміщуються на різних територіально розташованих серверах. У цьому випадку маємо своєрідну мегасистему масового обслуговування з динамічно змінюваною кількістю обслуговуючих пристроїв.

Еволюціонуючі та самоеволюціонуючі системи, зрозуміло, є системами зі змінними параметрами. Розглянута сервіс-орієнтована web-система є яскравим прикладом такої складної динамічної системи зі змінними компонентами (їх кількістю, характеристиками та визначеністю цих характеристик).

3. Багатоверсійні обчислення

3.1. Таксономія багатоверсійних обчислень

Попередні пояснення. Багатоверсійність або диверсність, як відомо, є одним з принципів, за яким може вибудовуватися механізм відмовостійкості. Частіше використовується термін диверсність, який є калькованим перекладом з англійської – diversity. В україно- та російськомовній літературі йому відповідають також різноманітність, версійність, N-версійність, багатоваріантність або багатоальтернативність. Найбільш поширеними серед них і такими, що однозначно ідентифікують відповідні поняття, є диверсність і багатоверсійність.

Термін диверсність має подвійне тлумачення. У більш широкому сенсі він визначає факт використання процесно-продуктної різноманітності, у вузькому – фіксує число версій, що використовуються. З урахуванням цього більш загальним є термін багатоверсійність, який співпадає з терміном диверсність при двох версіях.

Таксономічна схема. Таксономічна схема багатоверсійності складається з таких елементів (рис.3) [7]:

- *версія* – варіант адекватної за задачею, що вирішується, реалізації продукту (архітектури, набору апаратних, програмних засобів та інш.) або процесу; відповідно маємо *версію-продукт* або *версію-процес*;

- *версійна надмірність (ВН)* – вид надмірності, який характеризується використанням різних версій; ВН може мати, у свою чергу, кілька підвидів (типів); використання ВН для підвищення гарантоздатності називається *версійним резервуванням*;

- *багатоверсійність (БВ) (диверсність)* – принцип, який передбачає наявність кількох (двох) версій для виконання однієї задачі (реалізації продукту або процесу) з метою застосування надмірних даних для контроля, вибору або формування кінцевого чи проміжного результатів і прийняття рішення про подальше їх використання;

- *багатоверсійна система* (БВС) – система, у якій використовується кілька версій-продуктів і спеціальні засоби для їх генерування (ініціювання) та обробки результатів виконання версій; у двохверсійних системах одна з підсистем іноді має назву *основної* (primary system), друга – *диверсною* (secondary або diverse system);

- *мультидиверсна система* (БДВС) – БВС, у якій використовуються два або більше видів ВН; такі системи мають також назву (n,m)-систем (n – кількість версій, m – кількість видів ВН);

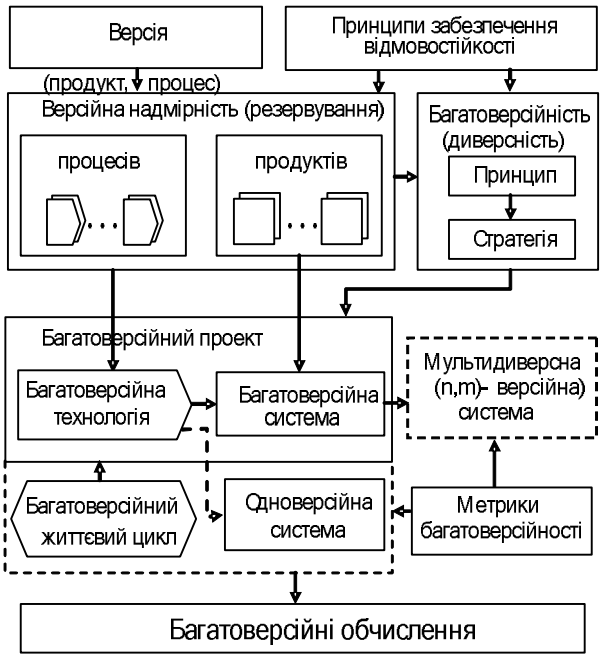

Рис. 3. Таксономічна схема багатоверсійності

- *багатоверсійна технологія* (БВТ) – сукупність взаємопов'язаних правил і проектних дій, у якій відповідно з стратегією БВ використовується кілька версій-процесів, що дозволяють отримати два або більше проміжних або кінцевих продуктів; для роз-робки БВС обов'язково використовується БВТ, для розробки одноверсійної системи – можуть бути використані як багатоверсійна, так і одноверсійна технологія;

- *багатоверсійний проект* (БВП) – проект, у якому застосовується БВТ, що приводить до створення одно- або багатоверсійної системи;

- *стратегія багатоверсійності* – сукупність загальних критеріїв і правил, визначаючих принципи формування та вибору БВТ;

- *багатоверсійний життєвий цикл* – життєвий цикл БВП;

- *метрика багатоверсійності (диверсності)* – показник ступеня незалежності версій.

Крім того, можуть використовуватися терміни, аналогічні тим, що поширені для резервування:

- *кратність версійності* (кратність версійного резервування);
- *загальна* та *роздільна версійність*;
- *повна* та *часткова версійність.*

Означені поняття можуть бути об'єднані поняттям *багатоверсійних обчислень* – гарантоздатних обчислень (обробки інформації), які виконуються з використанням принципу багатоверсійності.

3.2. Вплив багатоверсійності на гарантоздатність

Ефект багатоверсійності. Багатоверсійність як принцип побудови гарантоздатних систем зорієнтована, перш за все, на підвищення безвідмовності та функціональної безпеки. Порівняно зі структурним резервуванням версійне має особливості.

Якщо резервуються апаратні засоби і використовується, наприклад, мажоритарна схема, що поєднує три ідентичних канали (рис.4,а), то вона захищає від фізичних дефектів f_p з урахуванням того, що при експлуатації малоймовірна однотипна відмова за цими дефектами різних каналів.

Якщо таке саме резервування провести з урахуванням дефектів проектування f_d, матимемо той самий ефект для складової апаратних засобів (дефектів f_p) і незахищеність від програмної і апаратної складових за дефектами f_d (рис.4,б). Це пояснюється тиражуванням проектних дефектів у ідентичних версіях.

Якщо застосовується версійне резервування (рис.4,в), то ефект мажоритування розповсюджується і на проектні дефекти (рис.4,в). В ідеальному випадку, коли версії не мають спільних дефектів, а характеризуються індивідуальними дефектами f_{d1}, f_{d2}, f_{d3}, вони парируються мажоритарним елементом.

Що стосується дефектів взаємодії, то для них ситуація буде ще більш складною. Для дефектів, обумовлених фізичними впливами f_{ap}, ефект може бути схожим на ситуацію з версійним резервуванням для проектних дефектів залежно від моделі впливів. Для дефектів внаслідок інформаційних втручань f_{ai} при використанні версійної надмірності можливі різні ситуації – від позитивного до максимально негативного ефекту.

Якщо використовуються, наприклад, різні варіанти цифрових підписів, це призводить то нелінійного підвищення цілісності, тобто ефект зменшується з підвищенням кратності версійності [18]. Якщо застосовуються три криптографічні шифри для захисту блоку інформації і передачі від передавача до приймача, то це може призвести до зворотного ефекту. Це обумовлено тим, що атакуюча сторона, маючи відповідний ресурс, здійснить вторгнення і порушить працездатність системи (її інформаційну складову ІТС) за принципом найслабкішої ланки (рис.4,г).

Для запобігання цієї ситуації слід використовувати фрагментування даних, що захищаються, і зміну різних шифрів для кожного з фрагментів. При такому версійному резервуванні маємо ефект мажоритування і для забезпечення конфіденційності. Він ілюструється умовною схемою на рис.4,д.

Багатоверсійність може мати ефект і для інших складових гарантоздатності:

- для готовності, при застосуванні різних засобів відновлення та надання ресурсів;
- для достовірності, при використанні різних засобів контролю (багатоверсійному контролі);
- для живучості, при застосуванні різних ресурсів і процедур для керованої багатоступеневої деградації та інш.

Багатоверсійність для еволюціонуючих систем. У контексті еволюції багатоверсійні обчислення можуть мати динамічну складову, яка полягає:

- по-перше, у розширенні номенклатури продуктів і процесів, що можуть розглядатися як ресурс для побудови БВС або реалізації БВТ;
- по-друге, у появі нових видів версійної надмірності завдяки еволюційним змінам комп'ютерних та інформаційних технологій;
- по-третє, в удосконаленні процедур відбору диверсних компонент, механізмів обробки результатів роботи версій (каналів).

Це дозволяє покращувати відповідні характеристики системи, насамперед, зменшувати імовірність відмови за загальною причиною, а також оптимізувати витрати на забезпечення її припустимого рівня.

Слід зазначити, таке покращання може здійснюватися для деяких систем (наприклад, систем, розглянутих у п.2.3, рис.2) у динамічному режимі, тобто в процесі еволюції.

3.3. Оцінка диверсності та гарантоздатності

Методи оцінки диверсності. Оцінка диверсності залишається ключовою проблемою багатоверсійних обчислень. Проблема полягає у визначенні ступеня кореляції проектних дефектів для різних версій при недостатній інформації про них.

Для оцінки диверсності застосовуються різні методи, які можуть доповнювати один одного.

Теоретико-множинна метрична оцінка здійснюється з використанням [19]: метрик відносних, групових і абсолютних дефектів; метрик дефектів, що відрізняються і співпадають за проявом; метрик уразливостей та інш.

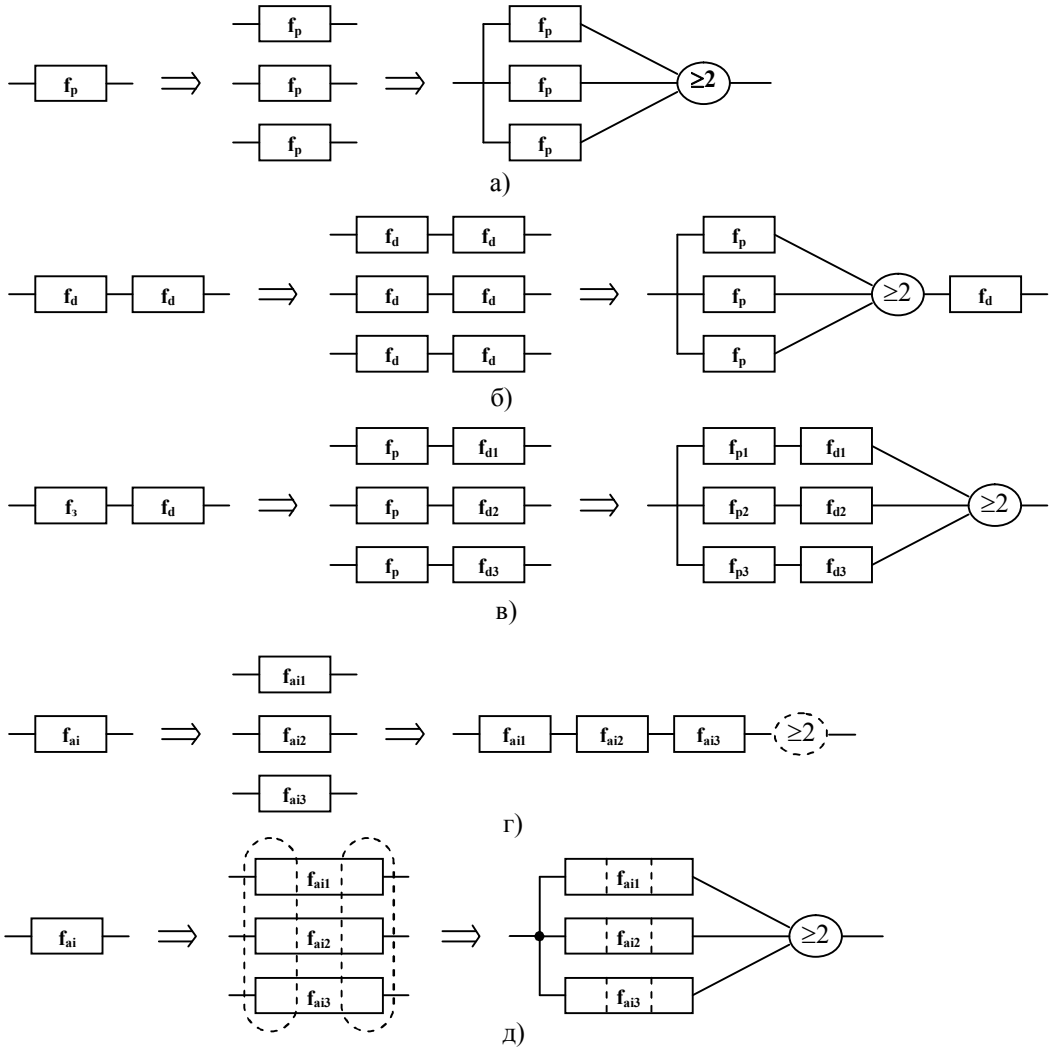

Рис. 3. Структурні схеми надійності (а-в) і інформаційної безпеки (г,д) багатоверсійних систем

Ці методи дають найбільш фізичну оцінку, але їх використання обмежено, з одного боку, недостатньою інформацією про дефекти, з іншого, – тим, що вони надають, як правило, апостеріорну оцінку і є лише базою для подальших оцінок.

Імовірнісна оцінка базується на байєсовському методі [20] або визначенні ймовірностей з використанням метрик диверсності [19]; такі методи є більш ґрунтовними у математичному відношенні та полегшують визначення показників гарантоздатності у звичному виді, але мають обмеження для застосування, також пов'язані з недостатньою інформацією про дефекти.

Статистичні методи використовують інформацію про тренди дефектів при тестуванні системи [21]. Вони можуть давати найбільш точну оцінку за умов наявності об'єктивної інформації про дефекти різних версій за кількістю, типами та у часі, що, у свою чергу, потребує високого рівня технологічної зрілості фірм-розробників програмного забезпечення та КС за шкалою SEI та відповідних інструментальних засобів обробки даних тестування.

Метод засіву дефектів базується на спеціальних способах засіву (ін'єкції) дефектів, подальшому тестуванні та обробці результатів виявлення засіяних та власних дефектів різних версій з додатковою класифікацією за множинними ознаками (аналогічно метричному методу). Реалізація такого методу пов'язана з необхідністю розробки складних інструментальних засобів для підтримки всіх технологічних операцій. На сьогодні невідомі детально розробленні засоби засів-орієнтованого оцінювання багатоверсійних систем.

Крім того, використання таких засобів потребує визначення профілів дефектів для засіву, тобто певної статистики про дефекти аналогічних проектів і відповідних типів версій.

Експертна оцінка диверсності може організовуватися за класичною схемою і має всі переваги та

недоліки цього підходу. Вона ус-кладнюється унікальністю багатоверсійних проект-тів, отже і «унікальністю» експертів, які можуть на-давати відповідну оцінку.

При оцінюванні експерти використовують інформацію про інші характеристики компонент, зокрема, *методи непрямих (побічних) оцінок.* Такі методи базуються, наприклад, на даних про метрики складності, які можуть надавати, з одного боку, пряму (безпосередню) інформацію про надійність версій (наприклад, метрика Холстеда), з іншого, - побічну інформацію («портрет версії»), яка є корисною для власне оцінки диверсності.

Інтервальна оцінка [22], на нашу думку, можуть бути певним компромісом між означеними вище методами, використовуючи сильні сторони кожного з них. Такі методи вже використовуються для оцінки надійності програмного забезпечення і можуть бути поширені на багатоверсійні проекти.

Аспект еволюційності. Зрозуміло, що оцінка диверсності пов'язана з оцінкою гарантоздатності, для якої застосовуються такі класичні методи як метод структурних схем надійності (безпеки, живучості [23]), марковський аналіз та інш. Для БВС оцінка диверсності є вхідною для розрахунків різних показників гарантоздатності.

Еволюційність ускладнює процес оцінювання диверсності, оскільки система змінюється, а інформація про характеристики і кореляцію версій старіє. З іншого боку, якщо зміна системи здійснюється за рахунок використання компонент з відомим характеристиками гарантоздатності (у тому числі, наприклад, з відомими уразливостями програмних компонент), це може полегшити процес оцінювання, а його результати зробити більш достовірними.

3.4. Багатоверсійні технології та рішення

Граф БВТ та класифікація версійної надмірності. БВТ фіксує кортеж елементів – варіантів вибору видів версійної надмірності по N етапах життєвого циклу і кількості відповідних поточних версій [23]. Задача вибору сформульована і вирішується як оптимізаційна задача пошуку шляхів у біполярному N-рівневому графі за критерієм «диверсність (надійність-безпека) – вартість. Основними проблемами є побудова графа БВТ виходячи з можливих видів версійної надмірності та їх сумісності, а також обчислення метрик диверсності і вартості по етапах.

Для побудови графа багатоверсійних технологій використовуються різні класифікаційні схеми версійної надмірності та матриці відповідних проектних рішень.

У роботах [7,24] надано огляд класифікаційних схем версійної надмірності:

- на системному рівні (суб'єктна, проектна, програмна, функціональна, сигнальна, апаратурна версійна надмірність);
- на програмному рівні (версійна надмірність моделей життєвого циклу, принципів і процесів розробки, ресурсів і засобів, проектних рішень);
- для систем на програмованій логіці (версійна надмірність за ступенем охоплення та глибиною багатоверсійності, версійна надмірність елементної бази, інструментальних засобів і мов програмування САПР, мов специфікацій та інш.).
- на концептуальному рівні (версійна надмірність систем счислення, підходів до розробки («біла» та «чорна» скринька); внутрішня та зовнішня версійна надмірність).

Аналіз класифікаційних схем показав, що: вони є по-перше, схемами фасетно-ієрархічного або матричного типів; по-друге, найбільш змістовною є класифікація ВН на системному рівні [24]; по-третє, складність і різноманітність версійної надмірності продуктів і процесів обумовлюють велику розмірність задачі вибору варіантів реалізації КС.

Куб багатоверсійності. Множину видів версійної надмірності зручно представити матрицею $VR = || vr_{ijk} ||$ в трьохмірному просторі, яка отримала назву *куба багатоверсійності* [7] (рис.5).

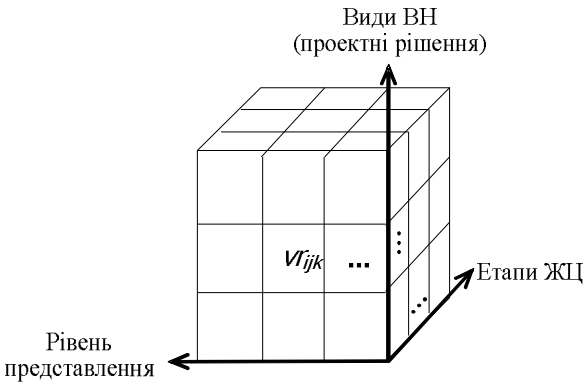

Рис. 5. Куб багатоверсійності

Куб багатоверсійності має координати: «етапи і процеси життєвого цикла системи (від специфікації до валідації та використання за призначенням) – рівень проектних рішень (від концептуального до компонентного) – конкретні види версіної надмірності (проектні рішення)», які відповідають нижнім індексам елементів vr_{ijk} матриці (куба).

На базі куба багатоверсійності можуть бути отримані двомірні матриці проектних рішень для різних рівнів (концептуального, системного, програмного, апаратного) або етапів життєвого циклу системи. Варіанти таких матриць надано у [7, 8, 23].

Ці матриці та куб в цілому акумулюють інформацію щодо БВТ і проектних рішень в процесі

еволюції відповідних технологій і є методичною основою для вирішення задач розробки гарантоздатних систем з використанням принципу багатоверсійності.

4. Еволюція технологій для гарантоздатних систем

4.1. «Колообіг» COTS-CrOTS

Технології, які використовуються для створен-ня гарантоздатних систем, досить динамічно онов-люються і можна визначити певні закономірності їх розвитку та еволюції, які носять сталий характер. Одна з таких закономірностей пов'язана з використанням раніше розроблених продуктів і технологій, які отримали назву «Off-The-Shelf» (OTS)-компонент, дослівно продуктів «з полиці».

OTS-компоненти, як відомо, мають класифікацію за кількома ознаками [25], з яких найважливішими, нашу думку, є їх походження (OTS-компоненти власної розробки або запозичені OTS-компоненти) та сфера первинного застосування (комерційні (тобто Commercial OTS) або COTS-компоненти, та компоненти, які розроблялися для критичних застосувань (Critical OTS) або CrOTS-компоненти).

У [26] була сформульована теза про те, що має місце «колообіг» OTS-компонент між критичними (CrAp) і комерційними (CAp) застосуваннями. Цей колообіг має часовий і змістовний вимір і підтверджується досвідом розвитку компонент, технологій та ідей за останні роки.

Часовий, більш сталий і загальний, вимір полягає у тому, що у другій половині 80-х на початку 90-х років сформувалася конверсійна політика, внаслідок якої критичні технології (CrOTS-компоненти) активно впроваджувалися у комерційних проектах. Завдяки масовому застосуванню у комерційній сфері вони відпрацьовувалися і після певної модифікації далі використовувалися в критичних системах.

Тобто має місце цикл (рис.6,б): вихідна компонента Pr (критичне застосування $CrAp_v$) → компонента Pr'(комерційне застосування CAp_v) → компонента Pr''(модифікація і можливе поширення на застосування CAp_μ) → компонента Pr*(критичне застосування $CrAp_\mu$). Можливий інший цикл руху та модифікації OTS-компонент (рис.6,б): з комерційної сфери у критичну і назад: Pr (CAp_i) → Pr* ($CrAp_i$) → Pr' ($CrAp_j$) → Pr'' (CAp_j).

На змістовному рівні можна також привести приклади такого COTS-CrOTS-колообігу. Це стосується, зокрема, технологій та компонент для систем високої готовності (High Availability Systems).

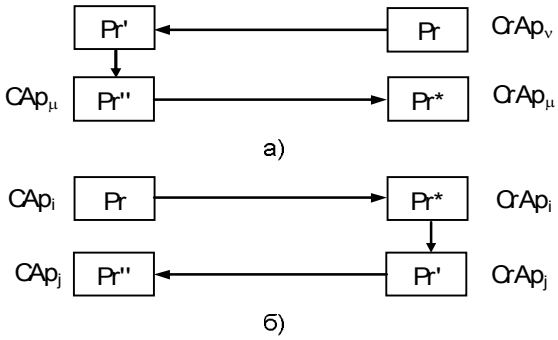

Рис. 6. «Колообіги» COTS-CrOTS-компонент

Наприклад, комерційна технологія STRATUS фірми HP, яка забезпечує високу готовність завдяки трьох- або чотирьохкратному резервуванню [27], була похідною від розробок для критичних застосувань. Зараз вона застосовується в таких бізнес-критичних системах. Інші приклади пов'язані з використанням багатоверсійності та колообігом БВТ між критичними та бізнес-критичними застосуваннями.

4.2. Диверсність: закон заперечення заперечення

Існують певні закономірності у еволюції самих багатоверсійних систем і технологій. Одна з них стосується розвитку та реалізації принципу багатоверсійності у інформаційно-управляючих системах (ІУС) АЕС. Вона відбиває дію закону «заперечення заперечення» [28].

Можна виділити такі етапи розвитку цього принципу та технологій його реалізації для систем аварійного захисту (рис.7):

1) перехід від апаратно-реалізованих комплек-тів систем к структурі, у якій один (основний) комп-лект реалізовувався апаратно (на «жорсткій» логіці, HW), а другий (диверсний) – як програмне рішення (SW) на мікропроцесорах (МП) (80-ті роки, перща фаза «заперечення»: від пари HW1-HW2 до пари HW-SW);

2) розробка основного і диверсного комплектів на МП з використанням різних мов програмування (наприклад, МП Intel і Motorola, мови C++ та Ada) (90-ті роки, друга фаза «заперечення»: від пари HW-SW до пари SW1-SW2);

3) використання ПЛІС від різних виробників, різних технологій виготовлення кристалів, різних мов опису апаратури для розробки основного та диверсного комплектів (початок 2000-х років, третя фаза «заперечення»: від пари SW1-SW2 до пари HW(FPGA)1-HW(FPGA)2; вона завершує цикл «заперечення заперечення»);

4) використання різних технологій інтелектуальних ядер (IP-core) та «м'яких» процесорів (soft processor) на базі ПЛІС для реалізації основного та диверсного комплектів (друга половина першого десятиліття 2000-х років, перша фаза «заперечення» нового циклу: від пари HW(FPGA)1-HW(FPGA)2 до пари HW(FPGA-SP)1-HW(FPGA-SP)2).

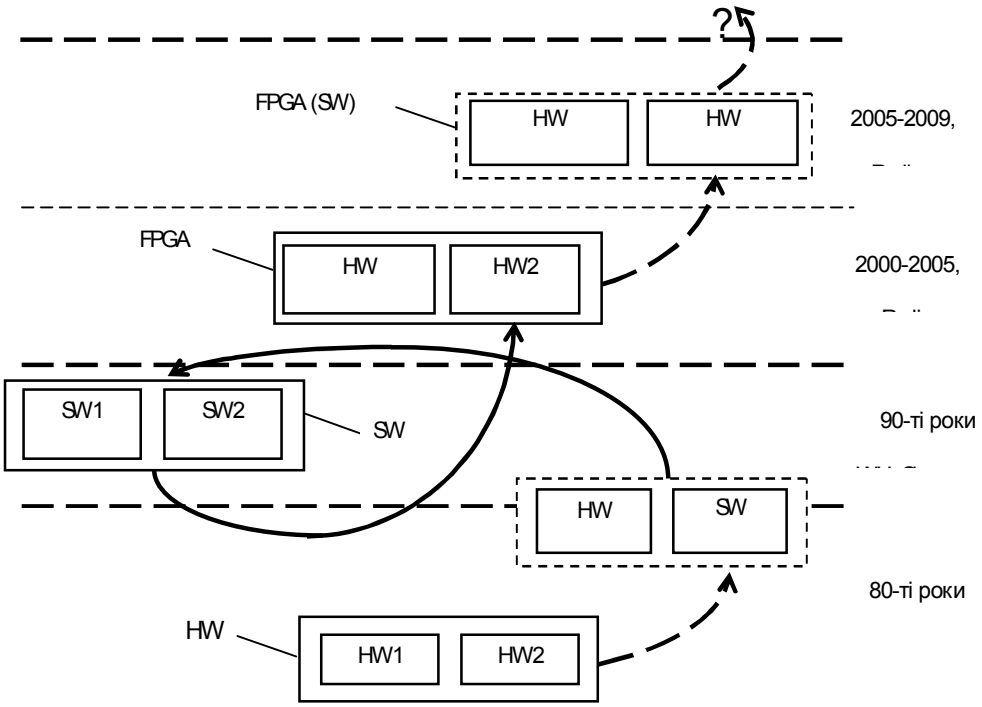

Рис. 7. Етапи розвитку технологій реалізації багатоверсійності реалізації для систем аварійного захисту

Логічним є запитання: яким буде наступний етап? Можливо, він буде пов'язаний з можливостями, які нададуть нанотехнології, і створенням природно надійних програмованих структур.

Висновки

Аспект еволюції надає новий цікавий вимір проблематиці гарантоздатних систем і багатоверсійності як одного з ключових принципів забезпечення гарантоздатності. Еволюція в контексті га-рантоздатності та багатоверсійності має дві скла-дові: сталу і динамічну. Перша полягає у суттєвих та поступових змінах технологій, які впливають на різні системи, друга – у змінах, які стосуються конкретної системи на протязі її життєвого циклу.

У другому випадку йдеться про динамічно еволюціонуючі гарантоздатні системи (самоеволюціонуючі системи реального часу, які здатні ураховувати зміни вимог до системи і характеристик середовища та використовують власні або зовнішні ресурси). Приклади таких систем вже мають місце і базуються на сервіс-орієнтованих web-системах, сервіс-орієнтованому програмному забезпеченні та «системах систем» [8], а також технологіях «grid» і «clauding computing». Доцільно провести опис та аналіз таких утворень з урахуванням загальної моделі еволюціонуючих систем.

В статті узагальнено таксономічну схему гарантоздатності з урахуванням факторів еволюції. Сформульовано єдиний підхід до формування операційного циклу відмовостійкості для різних типів дефектів і рівнів ієрархії, на яких забезпечується стійкість до різних типів дефектів. Аналогічний операційний цикл сформований для урахування змін вимог і середовища. Він потребує наповнення його конкретним змістом і розвитку концепції природно гарантоздатних та здатних до еволюції систем.

Удосконалено таксономію багатоверсійних обчислень та проаналізовано її ефект для властивостей гарантоздатності. Застосування багатоверсійності для забезпечення інформаційної безпеки, зокрема, конфіденційності, потребує спеціального аналізу і має сенс тільки при фрагментації даних при шифруванні. Аналіз методів оцінки диверсності дозволяє зробити висновок про необхідність їх комплексного застосування, тобто розробки технології «багатоверсійної» оцінки багатоверсійності, зокрема, з використанням інтервального методу.

Стала складова еволюції технологій для гарантоздатних систем має цікаві прояви, які стосуються COTS і CrOTS-компонент і реалізації принципу диверсності у критичних застосуваннях. Подальші дослідження у цьому напрямі потребують системного аналізу і прогнозування розвитку таких технологій, принципів оцінки та забезпечення гарантоздатності.

Окремої уваги потребує аналіз еволюції та конкретизації змісту парадигми «гарантоздатних систем з негарантоздатних компонент» [29] з використанням різних видів багатоверсійності [30].

Література

1. Avizienis A. Basic Concepts and Taxonomy of Dependable and Secure Computing / Avizienis A., Laprie J.-C., Randell B., Landwehr C. // IEEE Transactions on Dependable and Secure Computing. – 2004. - vol. 1, № 1. - P. 11-33.

2. Харченко В.С. Гарантоспособность и гарантоспособные системы: элементы методологии / В.С. Харченко // Радіоелектронні і комп'ютерні системи. – 2006. - №. 5. – С. 7-19.

3. TR 026764, ReSIST: Resilience for Survivability in IST. Deliverable D12. Resilience-Building Technologies: State of Knowledge, September 2006. – 345 p.

4. Douglas J. Futuyma. Evolution / Douglas J. Futuyma. - Sunderland, Massachusetts: Sinauer Associates, Inc, 2005. – 340 p.

5. Смирнов А.К. Управление жизненными циклами сложных систем / Смирнов А.К., Твердохлебов В.А. - РАН, Ин-т проблем точной механіки и управления: Изд-во Саратовского ун-та, 2000. – 122 с.

6. Gorbenko A. On Composing Dependable Web Services Using Undependable Web Components /Gorbenko A., Kharchenko V., Romanovsky A. // Int. Journal Simulation and Process Modelling. - 2007. - Vol. 3, 1-2. – P. 45-54.

7. Сиора А.А. (n,m)-версионные системы: таксономия, модели, технологии / А.А. Сиора, В.В. Скляр, В.С. Харченко // Вісник Харківського Національного університету №833. Серія "Математичне моделювання, інформаційні технології автоматизованих систем управління". - 2008. - Вип.10. - С.231-241.

8.Sousa-Poza A. System of Systems Engineering: an Emerging Multidiscipline / A. Sousa-Poza, S. Kovacic, C. Keating // Int. J. System of Systems Engineering - 2008. - Vol. 1, № ½. – P.1-17.

9. Харченко В.С. Гарантоздатність комп'ютерних систем: межа універсальності в контексті ін-формаційно-технічного стану / В.С. Харченко // Радіоелектронні і комп'ютерні системи. – 2007. - № 8. – С. 8-16.

10. Одарущенко О.Н. Многофрагментные марковские модели и их использование для оценки надежности обслуживаемых программно-технических комплексов / О.Н. Одарущенко .// Сб. науч. труд. НАН Украины. ПАНУ. – 1997. - Вып.1(5). – С. 102-105.

11. Поночовный Ю.Л. Моделирование надежности обновляемых программных средств нерезервированных информационно-управляющих систем постоянной готовности / Ю.Л. Поночовный, Е.Б. Одарущенко // Радіоелектронні і комп'ютерні системи. – 2004. - № 4(8). – С. 93-97.

12. Gorbenko A. Dependable Composite Web Services with Components Upgraded Online / A. Gorbenko, V. Kharchenko, P. Popov, A. Romanovsky // LNCS 3549, Architecting Dependable Systems III / R. de Lemos et al. (eds.). - Springer, 2005. – P. 92 - 121.

13. Ushakov A.A. Fault-Tolerant On-Board PLD-Systems: a Space-Structural Simulation and Methods of Adaptation / A.A. Ushakov, V.S. Kharchenko // Radioelectronics & Informatics. Proc. of East-West Design & Test Conference. - 2003. - No.3. - P.100-106.

14. Ushakov A. Fault-tolerant Embedded PLD-systems: Structures, Simulation, Design Technologies / A. Ushakov, V. Kharchenko, V. Tarasenko // Proc. of the 12th Intern. Conf. Mixed Design of Integrated Circuits and Systems, Krakow, Poland, June 12-15, 2003. – P.546-551.

15. Sterritt R. Self- properties in NASA missions / R. Sterritt, C.A. Rouff, J.L. Rash, W.F. Truszkowski, M.G. Hinchey // In 4th International Workshop on System/Software Architectures (IWSSA'05) in Proc. 2005 International Conference on Software Engineering Research and Practice (SERP'05), Las-Vegas, Nevada, USA, June 27 2005. P. 66-72.*

16. Gorbenko A. F(I)MEA-Technique of Web-services Analysis and Dependability Ensuring / A. Gorbenko, V. Kharchenko, O. Tarasyuk, A. Furmanov // LNCS 4157, Rigorous Development of Complex Fault-Tolerant Systems / M. Butler et al. (eds.). - Springer, 2006. – P.153-168.

17. Gorbenko A. How to Enhance UDDI with Dependability Capabilities / A. Gorbenko, A. Romanovsky, V. Kharchenko // Proc. 32nd Annual IEEE Int. Computer Software and Applications Conference (COMPSAC'2008). – Turku (Finland), 28 Jul. - 1 Aug., 2008. – P. 1023-1028.

Харченко В.С. Многоверсионность для обеспечения конфиденциальности и целостности информации: модели и методы / В.С. Харченко, М.А. Халин // Информационные технологии и системы. – К.: НАУ. – 2004. - № 1. – С.45-50.

19. Харченко В.С. Метрики диверсности: классификация, анализ и применение для оценки надежности и безопасности компьютерных систем управления / В.С. Харченко, И.В. Пискачева, В.В. Скляр // Открытые информационные и компьютерные интегрированные технологии. – Харьков: Нац. аэрокосмический ун-т «ХАИ». – 2001.– Вып. 9.– С. 194-214.

20. Littlewood B. Modelling the Effects of Combining Diverse Software Fault Removal Techniques / Littlewood B., Popov P. // IEEE Transactions on Software Engineering. – 2000, SE-26(12). – P. 1157-1167.

21. Lyu M.R. (edit.) Handbook of Software Reliability Engineering. - McGraw-Hill Company, 1996. - 805 p.

22. Жолен Л. Прикладной интервальный анализ / Л. Жолен, М. Кифер, О. Дидро, Э. Вальтер - М.: Ин-т компьютерных исследований, 2007. - 468 с.

23. Бахмач Е.С. Отказобезопасные информационно-управляющие системы на программируемой логике / Е.С. Бахмач, А.Д. Герасименко, В.А. Головир и др. / Под ред Харченко В.С., Скляра В.В. – Харьков: Нац. аэрокосмический ун-т «ХАИ», НПП «Радий». – 2008. – 380 с.

24. Preckshot G. Method for Performing Diversity and Defense-in-Depth Analysis of Reactor Protection Systems / G. Preckshot. - NUREG/CR-6303. - USA, Livermore: LLNL, 1994.

25. Харченко В.С. COTS– и CrOTS–подходы к повышению эффективности критических и коммерческих IT–проектов / В.С. Харченко, К.В. Харченко // Системи обробки інформації.– Харків: НАНУ, ПАНМ, ХВУ.–2002. - №2(18).- С.252-258.

26. Харченко В.С. Класифікація і профілювання OTS– продуктів для комп'ютерних систем управління / В.С. Харченко, В.В. Скляр, В.Г. Кожемяченко // Системи обробки інформації. – 2003. - №4 (24). – С. 67-73.

27. Харченко В.С. Базовые структуры многоканальных дублированных систем, устойчивые к отказам аппаратных и программных средств / В.С. Харченко, Ф.А. Асидех // Вісник Технологічного університету Поділля. – 2002. - № 3, т. 1. - С. 55-59.

28. Бахмач Е.С. Обеспечение и оценка безопасности информационных и управляющих систем АЭС на базе ПЛИС / Е.С. Бахмач, А.А. Сиора, В.В. Скляр, В.И. Токарев, В.С. Харченко // Радіоелектронні і комп'ютерні системи. – 2007. - № 7 (26). – С. 75-82.

29. Kharchenko V.S. Multi-version Information Technologies and Development of Dependable Systems out of Undependable Components / V.S. Kharchenko, V.V. Sklyar, A.V. Volkovoy // Proc.of IEEE DepCoS Conference. - Szklarska Poreba, Poland, June 14-16, 2007. – P. 18-24.

30. Сиора А.А. Отказоустойчивые системы с версионно-информационной избыточностью / А.А. Сиора, В.А. Краснобаев, В.С. Харченко / Под ред. Харченко В.С. – Х.: Министерство образования и науки Украины, Нац. аэрокосмический ун-т «ХАИ»., 2009. – 321 с.

Поступила в редакцию 2.03.2009

Рецензент: д-р техн. наук, проф., декан факультету Г.М. Жолткевич, Харківський національний університет им. В.Н. Каразіна, Харків, Україна.

ГАРАНТОСПОСОБНЫЕ СИСТЕМЫ И МНОГОВЕРСИОННЫЕ ВЫЧИСЛЕНИЯ: АСПЕКТЫ ЭВОЛЮЦИИ

В.С. Харченко

Предложена модифицированная таксономическая схема гарантоспособности с учетом факторов эволюции компьютерных систем (изменения функциональных требований, требований к гарантоспособности и характеристикам среды), а также операционного цикла и уровней обеспечения отказоустойчивости. Обобщена таксономия многоверсионности и проанализированы методы ее оценки. Определно понятие самоэволюционирующей системы. Проанализированы некоторые эволюционные аспекты технологий, используемых при создании гарантоспособных компьютерных систем.

Ключевые слова: гарантоспособность, многоверсионные вычисления, таксономия, операционный цикл отказоустойчивости, эволюция, самоэволюционирующие системы.

DEPENDABLE SYSTEMS AND MULTI-VERSION COMPUTING: ASPECTS OF EVOLUTION

V.S. Kharchenko

The modified dependability taxonomy scheme is proposed taking into account evolution aspects of computer-based systems (variation of dependability and functionality requirements, environment parameters), operational cycle and hierarchical levels of ensuring fault-tolerance. The taxonomy of multi-version computing is generalized and methods of assessing diversity are analyzed. A self-evolvable system concept is defined. Evolution aspects of technologies used to develop dependable compute-based systems are analyzed.

Key words: dependability, multi-version computing, taxonomy, computer systems, evolution.

Харченко В'ячеслав Сергійович – д-р техн. наук, професор, завідувач кафедри комп'ютерних систем та мереж, Національний аерокосмічний університет ім. М.Є. Жуковського «ХАІ», Харків, Україна, e-mail: V.Kharchenko@khai.edu.