УДК 681.3

Формальна модель координаційно-орієнтованої мережі для системи дистанційної освіти

М.М. Глибовець, Д.К. Гломозда

В статті розглядається задача побудови формальної моделі координаційної системи мережі для колаборативного середовища. Запропоновано автоматну модель, описану за допомогою мереж Петрі. На основі мережної моделі здійснено дослідження системи на обмеженість, збереження та активність.

The problem of building a formal model of coordination system for a collaboration environment is considered in the article. The automata model described with Petri nets is presented. The study of system's boundedness, preservation and liveness is conducted on the basis of the net model.

Національний університет „Києво-Могилянська Академія", Київ, вул. Сковороди, 2, корпус 1, тел. 463 6985, e-mail: glib@ukma.kiev.ua, e-mail: dmtrglm@yahoo.com

Першим кроком вирішення проблеми побудови ефективної комп'ютерної системи підтримки середовища колаборативного типу є створення формальної моделі. Це, вочевидь, має бути мережева модель. Втім, незважаючи на стрімкий розвиток мережевих технологій, питання побудови адекватної формальної моделі мережі — середовища взаємодії користувачів для спільного розв'язання конкретних задач (так зване collaboration environment) — все ще залишається відкритим. В даній роботі робиться спроба побудови такої моделі за допомогою скінченних автоматів та мереж Петрі.

За останні роки відбулося значне поширення Інтернет-послуг. Але технологія спільної роботи, що ґрунтується на спеціальному програмному забезпеченні, розвивалася досить повільно через те, що навіть на неформальному рівні досить складно описати дистанційне співробітництво, яке дає змогу людям спільно використовувати і маніпулювати мультимедійною інформацією в реальному часі і на різних рівнях модальності. Цей аспект контрастує зі спадковими "клієнт-серверними" "застосуваннями", такими як "відео-на-замовлення" (*video-on-demand*), і з такими асинхронними, документаційно-орієнтованими інструментами, як e-mail, миттєва передача повідомлень і чат-кімнати. Часто робилася спроба описати конструктивно мережеву синхронну мультимедійну роботу в Інтернет великих груп користувачів. На наш погляд, найбільш прискіпливо вона була зроблена Кіншуком та іншими [1]. Ми будемо спиратися на основні положення цієї роботи.

Проектування системи дистанційного співробітництва достатньо складне, оскільки охоплює користувачів, мережу та проблеми з головними комп'ютерами (хостами), наприклад, різнорідні платформи для "застосувань". Користувачі прагнуть отримати середовище дистанційного співробітництва, яке має забезпечити якість взаємодії, близької до "зустрічі віч-на-віч". В цьому випадку стають необхідними координаційні механізми досягнення консенсусу для спільного й ефективного використання ресурсів. Конфлікти, що блокують перебіг процесу, можуть трапитись як до, так і безпосередньо під час використання. Телеспівробітницькі послуги будуються на забезпеченні групових координаційних механізмів. Такі координаційні механізми необхідні, щоб надати

користувачам можливості досягти індивідуальних цілей в контексті групової дистанційної взаємодії, коли “дистанційна присутність” замінюється на фізичну.

Основні поняття

Комп’ютерну мережу зображатимемо у вигляді графу (V, E), де V — множина вершин графу, роль яких відіграють апаратні засоби мережі: комп’ютери користувачів, сервери, тощо; E — множина ребер графу, які уособлюють канали зв’язку ($E \subset V \times V$).

Середовищем роботи в мережі (Θ) називатимемо кортеж $\Theta = <S, U, R, F>$, де S (session) = (V, E) — набір *сеансів* роботи, U (users) — набір *користувачів*, R (resources) — *ресурси*, F (floors) — *рівні* керування ресурсами.

Сеанс забезпечує інфраструктуру для кооперації і співробітництва в мережі. Для системі дистанційної освіти характерними різновидами сеансів є, зокрема, лекції, семінари, тьюторські заняття, індивідуальна робота з викладачем.

Користувачі (в широкому розумінні) — особи (оператори), процеси, прикладні програми тощо. В системі користувачі репрезентуються своїми профілями. Під профілем користувача ми розуміємо його „особову справу”, що зберігається на головному сервері мережі. Профіль визначає права користувача в межах даної мережі та слугує джерелом інформації про поточні дії користувача. Операторами в рамках нашої предметної області є, наприклад, адміністратори, вчителі та студенти. Прикладом програми може бути комунікаційна оболонка (чат чи відеоконференція), процесу — запущена процедура проведення тестування.

Ресурси — прикладні компоненти координаційної структури (пам’ять, машинний час, канал зв’язку, тощо). Характерними прикладами в системі дистанційної освіти є „класна дошка” (на яку викладач поміщає інформацію, призначену для всіх студентів), індивідуальна скринька викладача (для прийому робіт), тощо.

Рівень — тимчасовий доступ та маніпуляційний привілей для ресурсів в рамках інтерактивної групової роботи. Наприклад, впродовж сеансу „лекція” користувач „викладач” має доступ до спільної змінної „класна дошка” в режимі „зчитування/запис”, а користувач „слухач” — лише в режимі „зчитування”. Як наслідок, маємо два рівні — „викладацький” та „слухацький” [1].

Автоматна модель системи

1.1 Модель контролера сеансу

Контролер сеансу (далі КС) регламентує створення сеансу та користування ним, забезпечує стабільність роботи, уможливлює доступ користувачів до потрібних їм ресурсів. Він же несе відповідальність за коректне завершення сеансу та звільнення зайнятих ресурсів.

Початковим станом контролера є стан бездіяльності (idle). Коли надходить запит на встановлення сеансу зв’язку, контролер опрацьовує його і ухвалює рішення. В разі, якщо у користувача немає права на таке з’єднання, або задані ним параметри не відповідають можливостям системи, КС відмовляє йому. В іншому випадку сеанс створюється.

У разі виникнення помилки часу виконання КС робить все можливе для того, щоб мінімізувати її вплив на роботу користувачів. В ідеалі користувачі взагалі не мають знати, що протягом певного часу щось було не так. Це стосується таких випадків як неухвалення запиту про виділення певного ресурсу чи аварійне завершення роботи одного з користувачів.

Якщо ж помилка занадто серйозна, контролер розпочинає процедуру коректного завершення сеансу, в рамках якої звільняються всі виділені ресурси, коректно від’єднуються всі користувачі, прибирається „сміття”. Якщо і на цьому етапі виникає невиправна позаштатна ситуація, контролер вимикає сеанс та повідомляє про помилку всім задіяним

сторонам (профілям користувачів, контролерам ресурсів), щоб вони зробили „прибирання” в себе. Після цього він повертається до початкового стану.

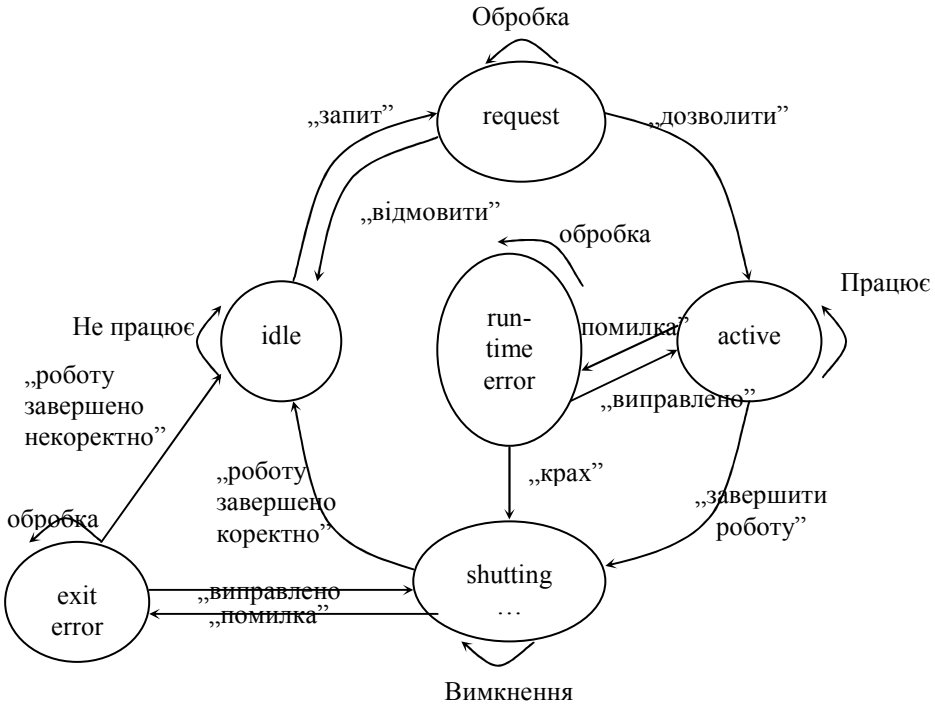

Рис. 1. Схема контролера сесії

Автоматна модель контролера сесії:

Стани: q1 = ‘idle’ (сесія вільна), q2 = ‘request’ (обробка запиту на створення або приєднання до сесії), q3 = ‘active’ (сесія використовується), q4 = ‘shutting …’ (процедура закриття сесії), q5=’run-time error’ (помилка в процесі роботи), q6=’exit error’ (помилка процедури завершення роботи).

Абетка (можливі повідомлення):

„запит”=1-1 (запит на створення або приєднання до сесії),
„відмовити”=2-1 (відхилити запит),
„дозволити”=2-2 (ухвалити запит),
„помилка”= #-1 (помилка),
„виправлено”= + (помилку виправлено),
„крах” = #-2 (невиправна помилка — аварійне завершення роботи),
„завершити роботу”=3-2 (активація процедури коректного завершення роботи),
„роботу завершено некоректно” = #-3 (невиправна помилка — аварійне завершення роботи),
„роботу завершено коректно” = 4-1 (коректне завершення роботи).

Таблиця 1.

	1-1	2-1	2-2	3-1	3-2	4-1	#-1	#-2	#-3	+
Q1	Q2	-	-	-	-	-	-	-	-	-
Q2	-	Q1	Q3	-	-	-	-	-	-	-
Q3	-	-	-	Q1	Q4	-	Q5	-	-	-
Q4	-	-	-	-	-	Q1	Q6	-	-	-
Q5	-	-	-	-	-	-	-	Q4	-	Q3
Q6	-	-	-	-	-	-	-	-	Q1	Q4

1.2 Модель профілю користувача

Поки користувач не увійшов у систему, профіль перебуває у стані „поза системою”. Увійшовши в систему, користувач приєднується до вже існуючої сесії або створює свою якщо має для цього повноваження (наприклад, розпочати семінар може лише викладач). Потім працює в мережі (цю „роботу” слід розуміти в найширшому сенсі, як довільну послідовність передбачених дій). Якщо користувач робить щось „не те”, система може примусово від’єднати його.

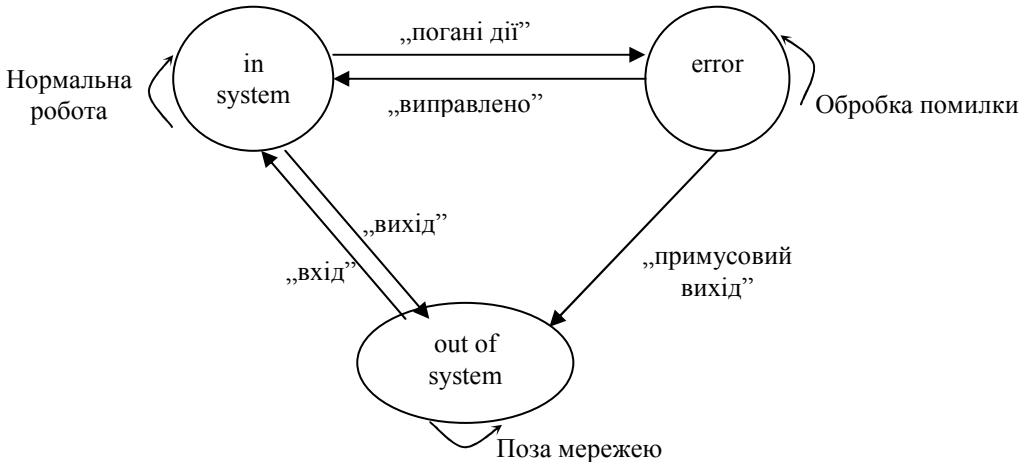

Рис. 2. Схема функціонування профілю користувача.

Автоматна модель профілю користувача:

Стани: q1 = ‘out of system’ (поза системою), q2 = ‘in system’ (у системі), q3 = ‘error’ (помилкові дії).

Абетка (можливі повідомлення):

„вхід”=1-1 (вхід в систему),

„вихід”=2-1 (вихід з системи),

„погані дії”=2-2 (випадкові або навмисні дії, що не відповідають правилам мережі),

„виправлено”=3-1 (наслідки помилки ліквідовано без наслідків),

„примусовий вихід”=3-2 (примусове від’єднання користувача від системи за грубе порушення).

Таблиця 2.

	1-1	2-1	2-2	3-1	3-2
Q1	Q2	-	-	-	-
Q2	-	Q1	Q3	-	-
Q3	-	-	-	Q2	Q1

1.3 Модель контролера ресурсу

Контролер ресурсу відповідає за надання ресурсу за запитом та за коректне його звільнення (в разі аварійного завершення роботи відповідного контролера сесії).

Життєвий цикл контролера ресурсу починається зі створенням ресурсу. Після цього контролер переходить в пасивний стан очікування на запити на використання ресурсу. Коли ресурс виділено, контролер слідкує за його використанням, і в разі вичерпання ресурсу (якщо йдеться, наприклад, про певну кількість машинного часу) повертається у пасивний стан. Якщо ресурс передбачає можливість паралельного використання кількома користувачами, контролер бере на себе функції забезпечення взаємовиключення/синхронізації використання ресурсу з метою уникнення взаємного блокування (на малюнку 3 відповідні стани та повідомлення позначено пунктиром). Ще одним завданням контролера є прибирання „сміття”: коректне звільнення пам’яті, каналів зв’язку тощо.

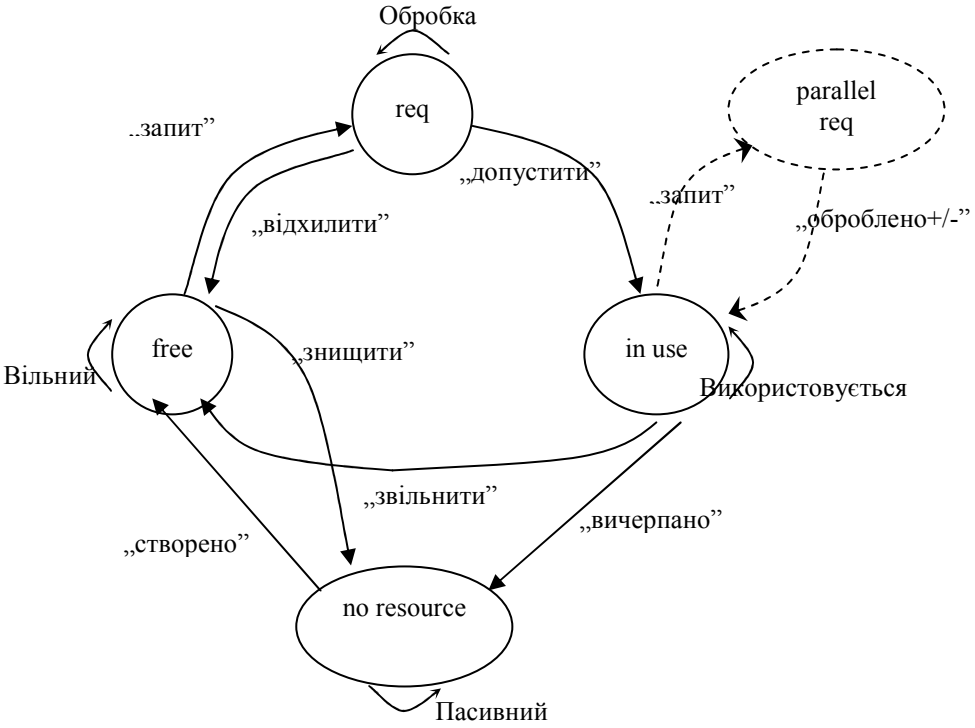

Рис. 3. Схема контролера ресурсу.

Автоматна модель контролера ресурсу:

Стани: q1 = ‘no resource’ (ресурсу нема), q2 = ‘free’ (ресурс вільний), q3 = ‘req’ (обробка запиту на використання ресурсу), q4 = ‘in use’ (ресурс використовується), q5 = ‘parallel req’ (обробка запиту на спільне використання ресурсу).

Абетка (можливі повідомлення):

„створено”=1-1 (поява ресурсу в системі),
„запит”=2-1 (запит на використання ресурсу),
„знищити”=2-2 (видалення ресурсу),
„відхилити”=3-1 (відмова у наданні ресурсу),
„допустити”=3-2 (згода на надання ресурсу),
„звільнити”=4-1 (звільнення ресурсу),
„вичерпано”=4-2 (сигнал про вичерпання ресурсу),
„оброблено +”=5-1+ (позитивний результат обробки запиту на спільне використання ресурсу).

„оброблено -”=5-1- (негативний результат обробки запиту на спільне використання ресурсу).

Таблиця 3.

	1-1	2-1	2-2	3-1	3-2	4-1	4-2	5-1+/-
Q1	Q2	-	-	-	-	-	-	-
Q2	-	Q3	Q1	-	-	-	-	-
Q3	-	-	-	Q2	Q4	-	-	-
Q4	-	Q5	-	-	-	Q2	Q1	-
Q5	-	-	-	-	-	-	-	Q4

1.4 Модель контролера рівня

Протокол рівневого контролю домагається доступу до спільно використовуваних (розподілених) об'єктів, надаючи доступ до рівнів відповідно до політики, визначеної для групи обслуговування («слухачі лекції», «група, що виконує лабораторну роботу», «екзаменаційна комісія» тощо).

Можливі стани рівня такі. *Вільний* (*Free*) — доступний, невикористовуваний рівень. *Очікування* (*Idle*) — задіяний, але неактивний рівень. *Req* (*є запит*) — рівень, на який надійшли запити (створення або приєднання до рівня, тощо). *Зайнятий* (*Busy*) — вже наданий і зайнятий рівень. „Скасування” рівня означає примусове скорочення терміну служби рівня контролером (наприклад, через помилку, що може призвести до падіння всієї мережі).

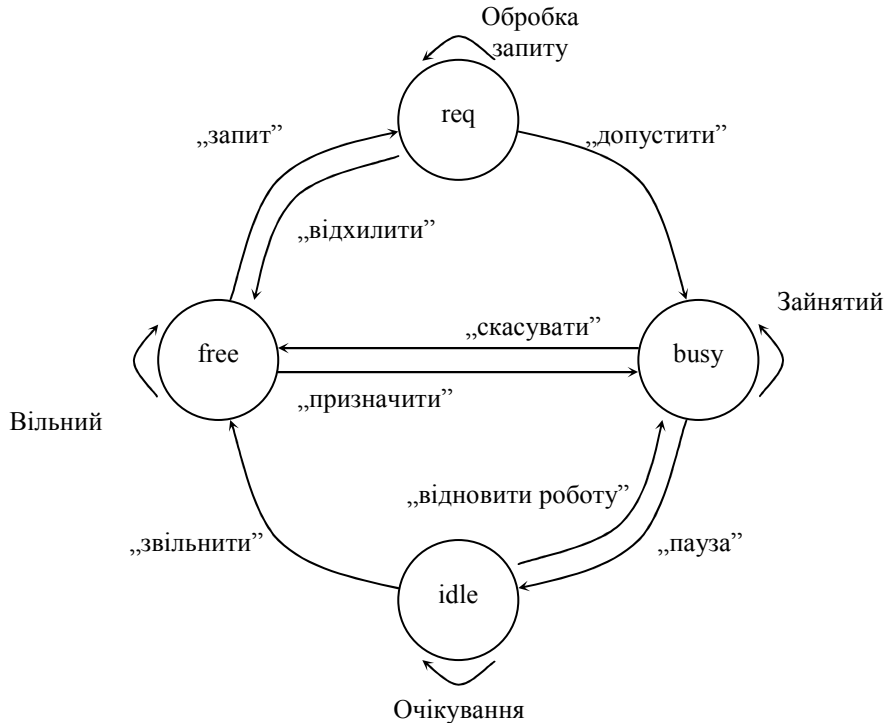

Рис. 4. Схема контролера рівня.

Автоматна модель контролера рівня:

Стани: q1 = ‘free’ (рівень вільний), q2 = ‘req’ (обробка запиту), q3 = ‘busy’ (рівень зайнято), q4 = ‘idle’ (перерва в роботі, але без звільнення рівня).

Абетка (можливі повідомлення):

„запит”=1-1 (запит на використання рівня),

„відхилити”=2-1 (запит відхилено),

„допустити”=2-2 (запит ухвалено),
„скасувати”=3-1 (невідкладно і примусово звільнити рівень),
„призначити”=1-2 (невідкладно надати рівень),
„пауза”=3-2 (пауза в роботі без звільнення рівня),
„відновити роботу”=4-1 (відновлення роботи після паузи),
„звільнити”=4-2 (звільнення рівня після завершення роботи).

Таблиця 4.

	1-1	2-1	2-2	3-1	1-2	3-2	4-1	4-2
Q1	Q2	-	-	-	Q3	-	-	-
Q2	-	Q1	Q3	-	-	-	-	-
Q3	-	-	-	Q1	-	Q4	-	-
Q4	-	-	-	-	-	-	Q3	Q1

Мережна модель системи

Тепер на базі автоматної моделі побудуємо модель мовою мереж Петрі. Для скінченного автомата (Q, A, b, q_0, F) мережа Петрі (P, T, I, O) визначається так:

$$P = Q \cup A \cup b,$$

$$T = \{t_{q,\sigma} \mid q \in Q, \sigma \in A\},$$

$$I(t_{q,\sigma}) = \{q, \sigma\},$$

$$O(t_{q,\sigma}) = \{q_0(q, \sigma), F(q, \sigma)\}$$ [2, С. 47].

Зразу хочемо пояснити, що „зайві” повідомлення, тобто такі, що відсутні в абетці автоматної моделі, введено для полегшення подальшого об’єднання мереж в єдину систему.

Отже:

2.1 Мережна модель контролера сеансу

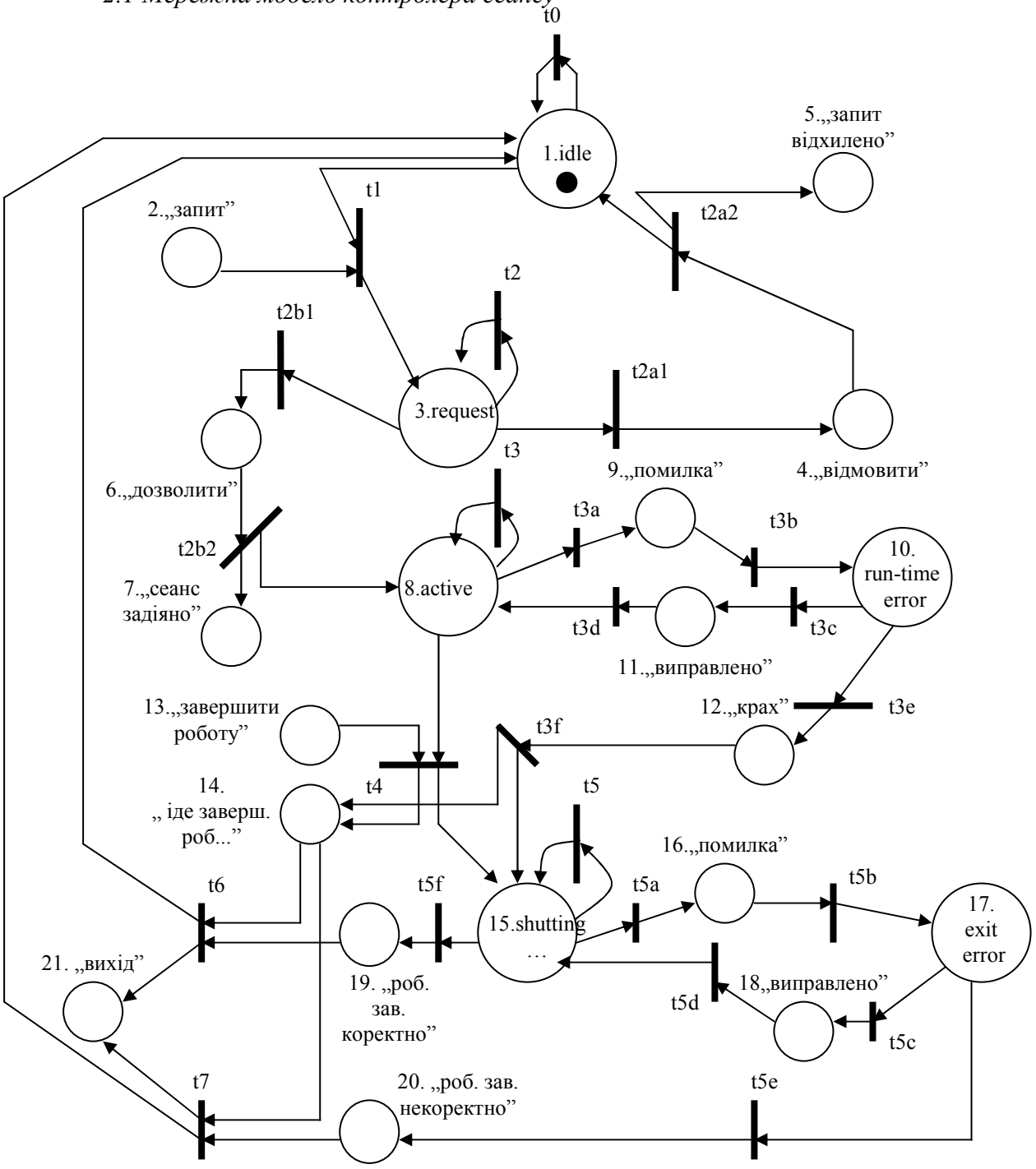

Рис. 5. Мережна модель контролера сесії.

Таблиця 5.

I(t0) = {idle}	O(t0) = {idle}
I(t1) = {idle, „запит”}	O(t1) ={request}
I(t2) = {request}	O(t2) ={request}
I(t2a1) = {request}	O(t2a1) ={„відмовити”}
I(t2a2) = {„відмовити”}	O(t2a2) ={idle, „запит відхилено”}
I(t2b1) = {request}	O(t2b1) ={„дозволити”}
I(t2b2) = {„дозволити”}	O(t2b2) ={active, „сеанс задіяно”}
I(t3) = {active}	O(t3) ={active}
I(t3a) = {active}	O(t3a) ={„помилка”}
I(t3b) = {„помилка”}	O(t3b) ={run-time error}
I(t3c) = {run-time error}	O(t3c) ={„виправлено”}
I(t3d) = {„виправлено”}	O(t3d) ={active}
I(t3e) = {run-time error}	O(t3e) ={„крах”}
I(t3f) = {„крах”}	O(t3f) ={shutting …, „іде заверш. роб...”}
I(t4) = {active, „завершити роботу”}	O(t4) ={shutting…, „іде заверш. роб...”}
I(t5) = {shuttting …}	O(t5) ={shutting…}
I(t5a) = {shuttting …}	O(t5a) ={„помилка”}
I(t5b) = {„помилка”}	O(t5b) ={exit error}
I(t5c) = {exit error}	O(t5c) ={„виправлено”}
I(t5d) = {exit error, „виправлено”}	O(t5d) ={shutting …}
I(t5e) = {exit error}	O(t5e) ={„роб. зав. некоректно”}
I(t5f) = {shuttting …}	O(t5f) ={„роб. зав. коректно”}
I(t6) = {„іде заверш. роб...”, „роб. зав. коректно”}	O(t6) ={idle, „вихід”}
I(t7) = {„іде заверш. роб...”, „роб. зав. некоректно”}	O(t7) ={idle, „вихід”}

2.2 Мережна модель профілю користувача

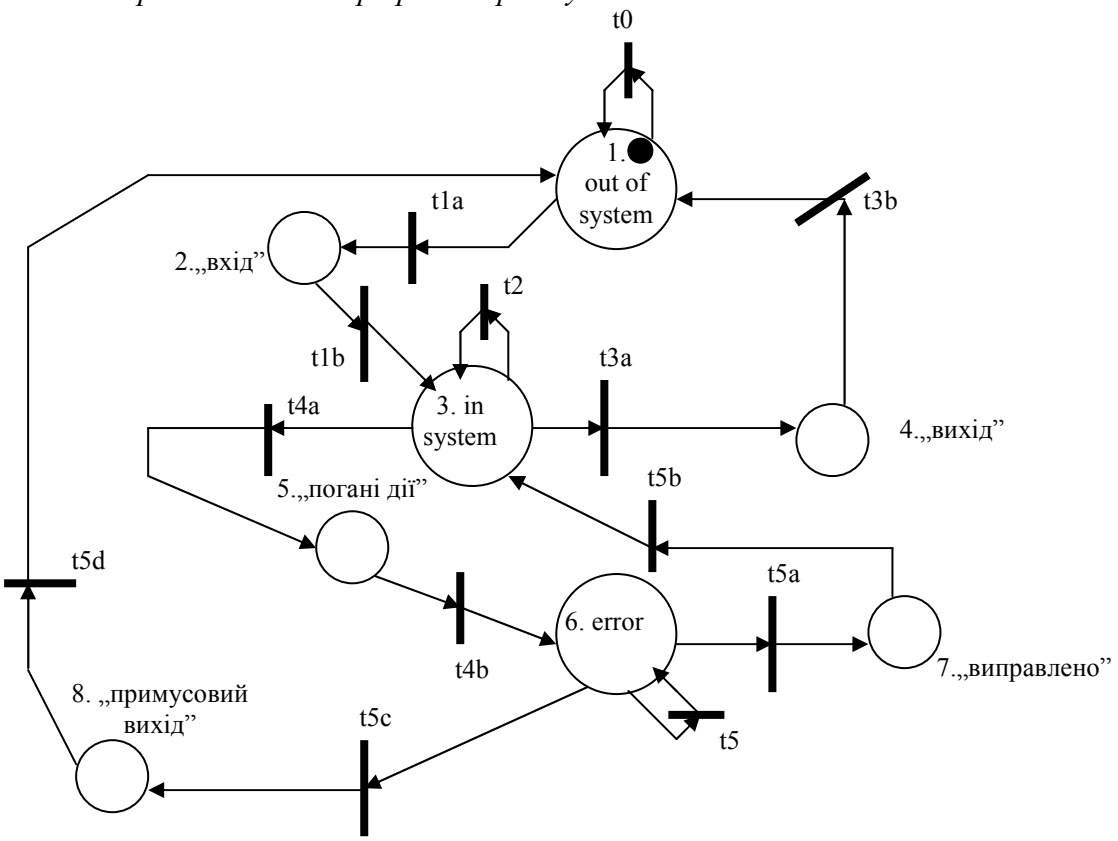

Рис. 6. Мережна модель профілю користувача.

Таблиця 6.

I(t0) = {out of system}	O(t0) = {out of system}
I(t1a) = {out of system}	O(t1a)={„вхід”}
I(t1b) = {„вхід”}	O(t1b)={in system}
I(t2) = {in system}	O(t2)={in system}
I(t3a) = {in system}	O(t3a)={„вихід”}
I(t3b) = {„вихід”}	O(t3b)={out of system}
I(t4a) = {in system}	O(t4a)={„погані дії”}
I(t4b) = {„погані дії”}	O(t4b)={error}
I(t5) = {error}	O(t5) = {error}
I(t5a) = {error}	O(t5a)={„виправлено”}
I(t5b) = {„виправлено”}	O(t5b)={in system}
I(t5c) = {error}	O(t5c)={„примусовий вихід”}
I(t5d) = {„примусовий вихід”}	O(t5d)={out of system}

2.3 Мережна модель контролера ресурсу

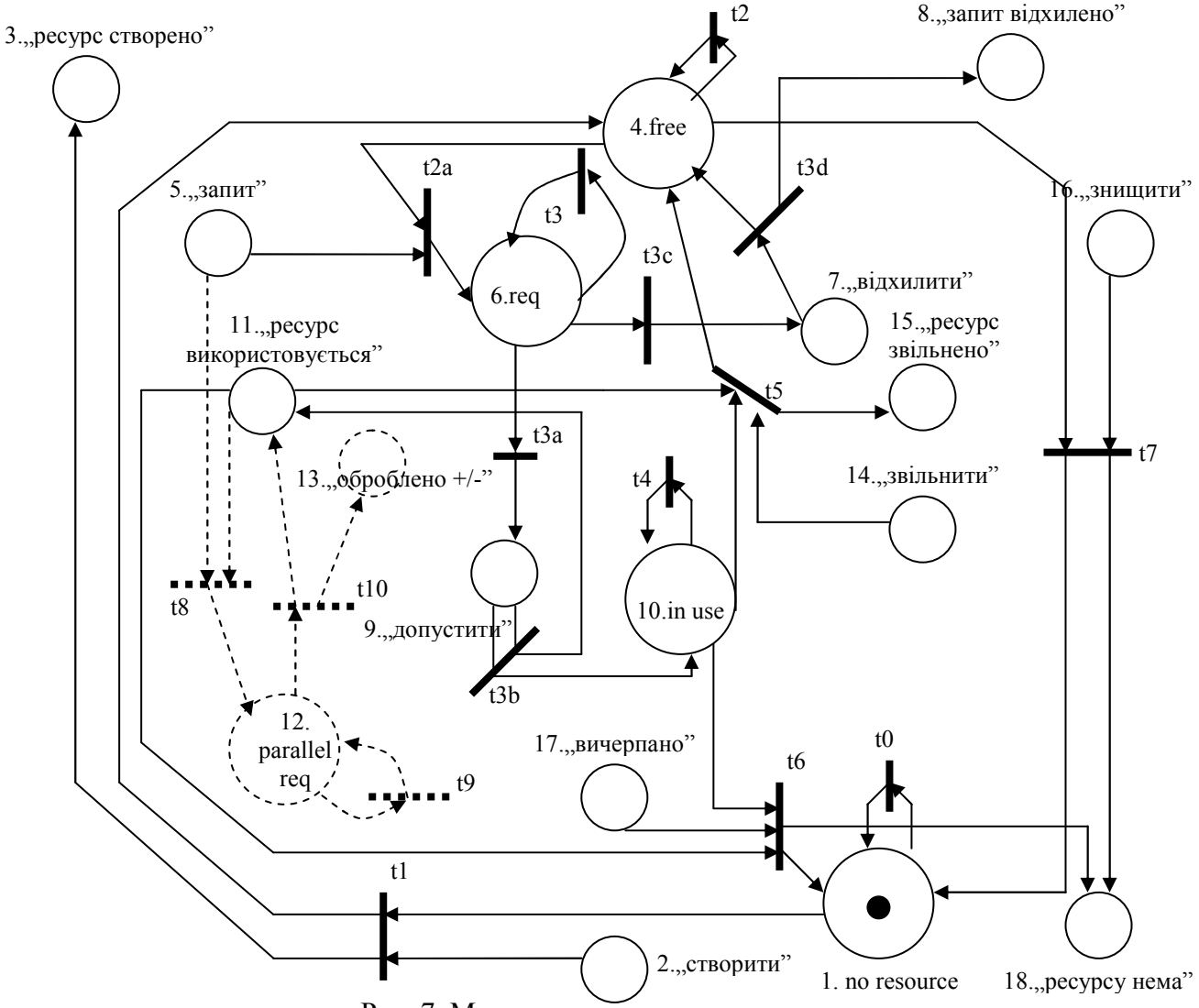

Рис. 7. Мережна модель контролера ресурсу.

Таблиця 7.

I(t0) = {no resource}	O(t0)={no resource}
I(t1) = {no resource, „створити”}	O(t1)={free, „ресурс створено”}
I(t2) = {free}	O(t2)={free}
I(t2a) = {free, „запит”}	O(t2a)={req}
I(t3) = {req}	O(t3)={req}
I(t3a) = {req}	O(t3a)={„допустити”}
I(t3b) = {„допустити”}	O(t3b)={in use, „ресурс використовується”}
I(t3c) = {req}	O(t3c)={„відхилити”}
I(t3d) = {„відхилити”}	O(t3d)={free, „запит відхилено”}
I(t4) = {in use}	O(t4)={in use}
I(t5) = {in use, „звільнити”, „ресурс використовується”}	O(t5)={free, „ресурс звільнено”}
I(t6) = {in use, „вичерпано”, „ресурс використовується”}	O(t6)={no resource, „ресурсу нема”}
I(t7) = {free, „знищити”}	O(t7)={no resource, „ресурсу нема”}
I(t8) = {„запит”, „ресурс використовується”}	O(t8)={parallel req}
I(t9) = {parallel req}	O(t9)={parallel req}
I(t10) = {parallel req}	O(t10)={„оброблено +/-”, „ресурс використовується”}

2.4 Мережна модель контролера рівня

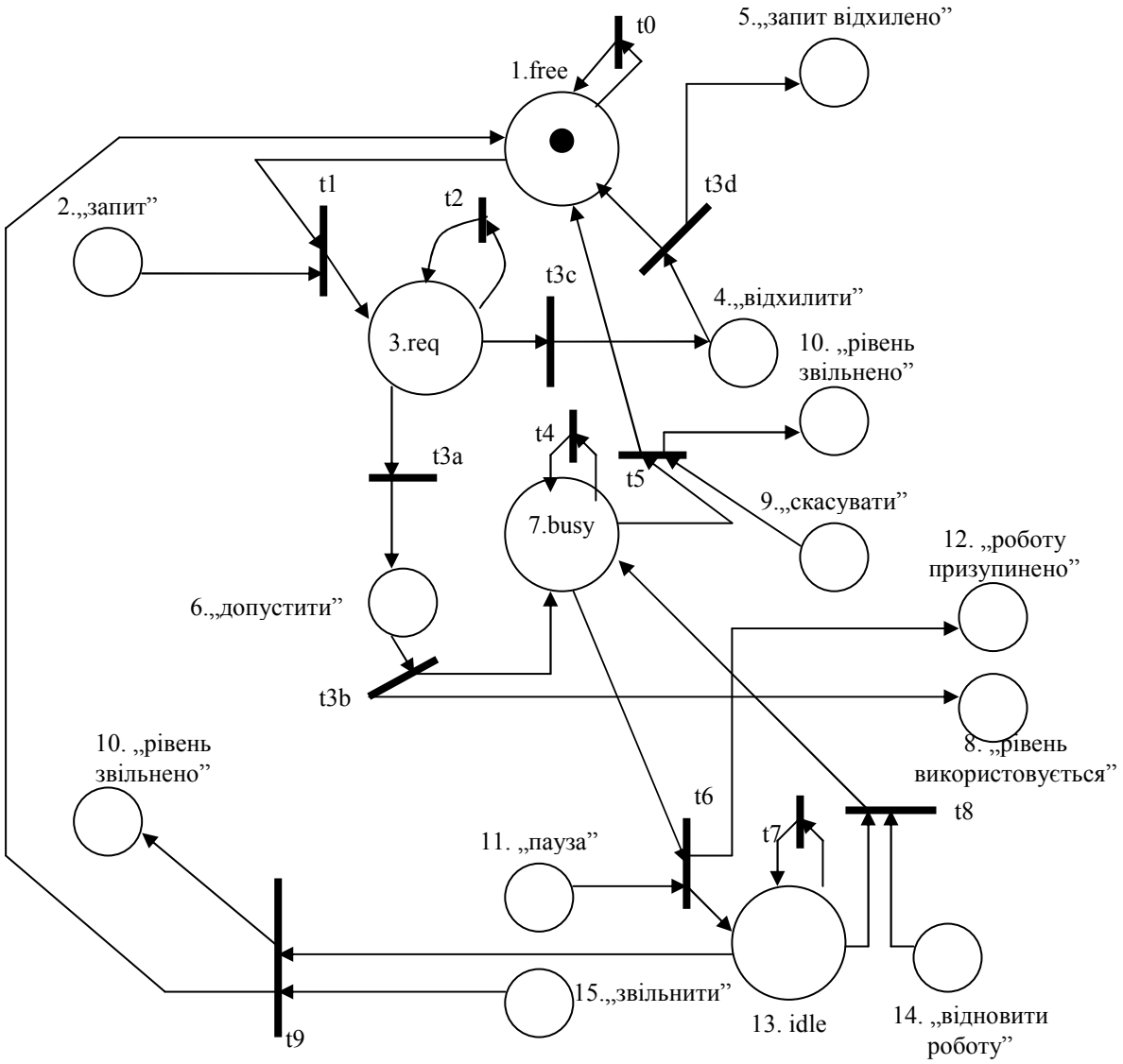

Рис. 8. Мережна модель контролера рівня.

Таблиця 8.

I(t0) = {free}	O(t0)={free}
I(t1) = {free, „запит”}	O(t1)={req}
I(t2) = {req}	O(t2)={req}
I(t3a) = {req}	O(t3a)={„допустити”}
I(t3b) = {„допустити”}	O(t3b)={busy, „рівень використовується”}
I(t3c) = {req}	O(t3c)={„відхилити”}
I(t3d) = {„відхилити”}	O(t3d)={free, „запит відхилено”}
I(t4) = {busy}	O(t4)={busy}
I(t5) = {busy, „скасувати”}	O(t5)={free, „рівень звільнено”}
I(t6) = {busy, „пауза”}	O(t6)={idle, „роботу призупинено”}
I(t7) = {idle}	O(t7)={idle}
I(t8) = {idle, „відновити роботу”}	O(t8)={busy}
I(t9) = {idle, „звільнити”}	O(t9)={free, „рівень звільнено”}

Хоча опис за допомогою мережі Петрі більш громіздкий, ніж мовою скінченних автоматів, опис мовою мереж має свої переваги, головною з яких є порівняна легкість

об’єднання мереж, що описують окремі елементи системи, в одну загальну мережу, що описуватиме систему загалом. Крім того, мережі Петрі краще пристосовані для опису паралельних процесів. Нарешті, аналізуючи мережу Петрі, можна багато чого сказати про властивості системи, яку вона моделює. Зокрема дослідити систему на активність (відсутність тупикових (термінальних) вершин) та досяжність всіх станів за допомогою дерева досяжності [2, С. 79—106].

3.1 Дослідження моделі контролера сеансу

Твердження 1. Мережа моделі контролера сеансу обмежена, незберігаюча та активна.

Доведення. Побудуємо дерево досяжності для мережі контролера сеансу (рис. 9).

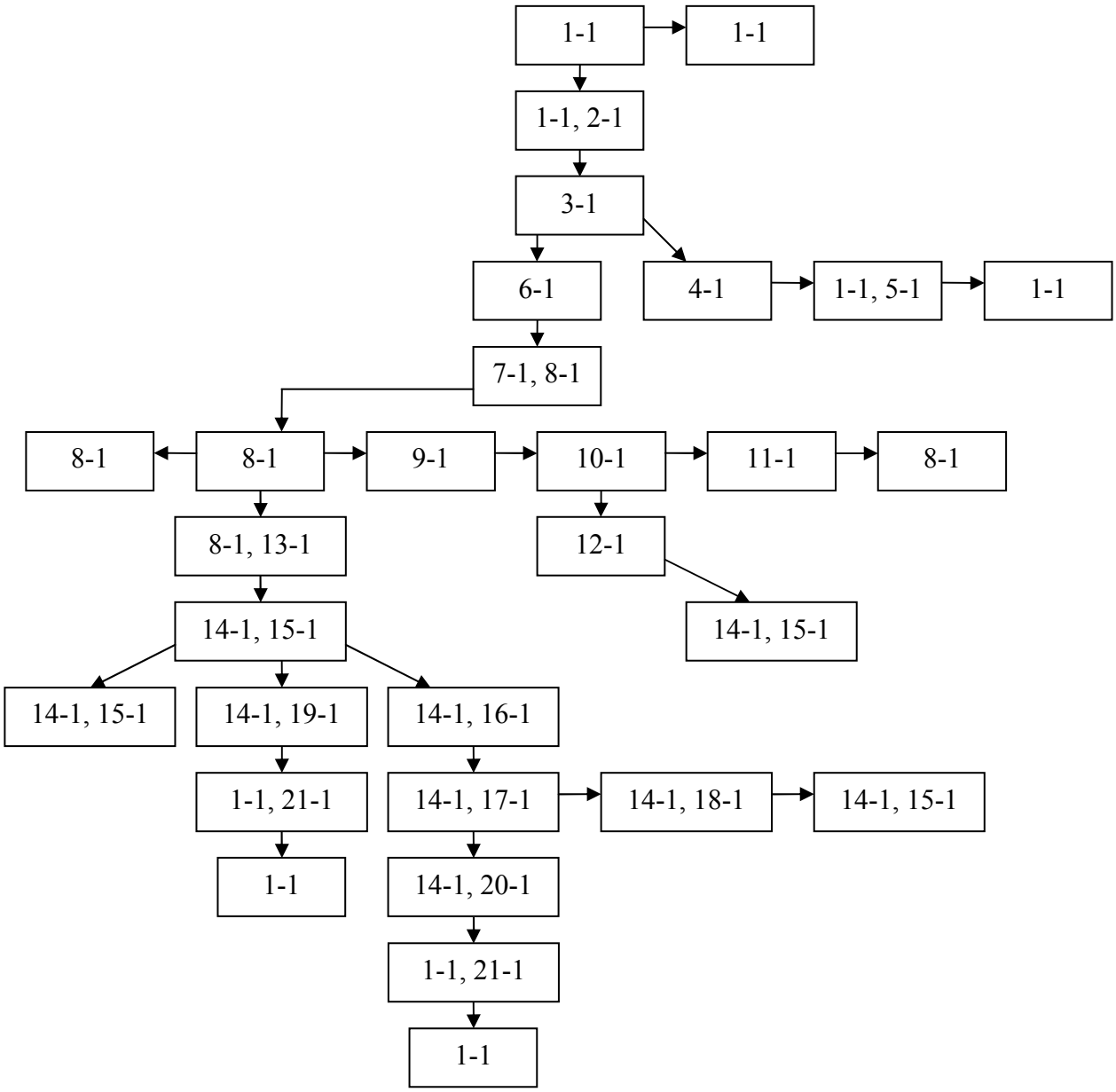

Рис. 9. Дерево досяжності для мережної моделі контролера сеансу.

Цифри на вершинах відповідають певному маркуванню мережі. Наприклад, (14-1, 16-1) означає, що в вершинах з номерами 14 та 16 розташовано по одній фішці.

Аналізуючи це дерево, ми можемо сказати, що мережа обмежена (кількість фішок не може сягнути нескінченності); незберігаюча (кількість фішок може змінюватися за рахунок

додавання та вилучення фішок, що відповідають вхідним та вихідним сигналам); активна (термінальні вершини відсутні).

Провівши аналогічні дослідження інших наведених нами моделей, одержимо такі результати:

Твердження 2. Мережа моделі профілю користувача є обмеженою, зберігаючою та активною.

Твердження 3. Мережа моделі контролера ресурсу є обмеженою, незберігаючою та активною.

Твердження 4. Мережа моделі контролера рівня є обмеженою, незберігаючою та активною.

Висновок

Запропонований нами варіант моделі не є остаточним, і отримає розвиток в ході подальшої праці. Але його можна вважати базисом, користуючись яким можна досліджувати різноманітні координаційні системи.

Представлена модель обслуговує як теоретичні, так і практичні цілі. Вона забезпечує більш складну структуру для формальної специфікації і перевірки правильності спільних систем, наприклад, системи перевірки прототипу. Вона також враховує описи можливостей процесу для оперативної специфікації, встановлення і запиту членства та стану координації активної конференції. Цю можливість можна вважати ресурсною чи системною характеристикою, що впливає на вибір корисних конфігурацій окремих компонентів.

Література

1. *Chen N. S., Ko H.-C., Kinshuk, Lin T.* Synchronous Learning Model over the Internet // Innovations in Education and Teaching International. — 2005. — Vol. 42 (2). — P. 181—194.
2. *Питерсон Дж.* Теория сетей Петри и моделирование систем. — М.: Мир, 1984. — 264 с.