Bartłomiej Gaweł*

PROCESS MINING: USPRAWNIENIE PRZEBIEGÓW PROCESÓW BIZNESOWYCH W OPARCIU O ANALIZĘ LOGÓW SYSTEMU ZARZĄDZANIA PRZEPŁYWEM PRACY – ANALIZA PRZYPADKU

Proces biznesowy składa się z wykonywanych przez agentów czynności podlegających ograniczeniom kolejnościowym oraz czasowym, których przekroczenie powoduje uruchomienie odpowiednich scenariuszy ratunkowych lub zakończenie procesu. W artykule dokonano próby zastosowanie process miningu w metodzie dynamicznego ustalania żądanych czasów zakończenia zadań w systemie informatycznym koordynującym przepływ procesów usługowych w średnim przedsiębiorstwie.

Słowa kluczowe: *process mining, żądane czasy zakończenia zadań, systemy zarządzania przepływem pracy*

PROCESS MINING: USING PROCESS MINING AS A TOOL FOR BUSINESS PROCESS IMPROVEMENT – CASE STUDY

Business processes consist of multiple activities, and their enactment is carried out by human agents and software systems. Typically, business processes and activities constituting them have deadline. When activity misses its deadline, special action may be triggered. In this article I try to apply process mining to deadline assignment problem in workflow management systems in middle size company

Keywords: *process mining, deadline, workflow management systems*

1. WPROWADZENIE

Wdrożenie systemu informatycznego powinno opierać się na dostępnej dokumentacji procesów biznesowych w przedsiębiorstwie. W praktyce, szczególnie w małych i średnich przedsiębiorstwach, dokumentacja taka nie istnieje, a przebieg konkretnych procesów opiera się na tworzonych *ad hoc* kanałach komunikacyjnych pomiędzy członkami organizacji. W takich przypadkach do efektywnego wdrożenia wymagane jest wcześniejsze zbudowanie prototypu, którego zadaniem będzie automatyzacja i zapis komunikacji pomiędzy użytkownikami. Process mining polega na analizie logów systemu w celu uzyskania wzorców, z których jak z klocków można zbudować model referencyjny procesu lub wyznaczyć efektyw-

* Katedra Informatyki Stosowanej, Wydział Zarządzania, Akademia Górniczo-Hutnicza, Kraków

ność pracy tworzących ten proces zasobów. W tym artykule pokażę, że narzędzia process miningu można również wykorzystać do usprawnienia przebiegu procesu.

Nie we wszystkich gałęziach działań biznesowych stosowanie systemów wspomagających zarządzanie przepływem pracy jest równie łatwe. Jedną z trudniejszych są procesy usługowe prowadzone w skali masowej, takie jak telefoniczna obsługa reklamacji klientów w firmach telekomunikacyjnych, realizacja roszczeń ubezpieczeniowych od osób fizycznych, masowa windykacja należności lub obsługa konsumpcyjnych kredytów bankowych. Wdrożenie systemu wspomagającego przepływ pracy standaryzuje proces. Z kolei częsta komunikacja z otoczeniem w przebiegach procesów usługowych wymusza stosowanie wyrafinowanych mechanizmów wspomagających ich elastyczność [1], rozumianą tu jako właściwa szybkość odpowiedzi na zmiany zachowania, wykazywanie inicjatywy oraz umiejętność współdziałania ze znajdującymi się na zewnątrz organizacji autonomicznymi agentami (np. klientami, innymi organizacjami) [2]. Dlatego też automatyzacja i standaryzacja procesów usługowych jest wciąż zadaniem stanowiącym poważne wyzwanie przed badaczami. Istnieje wiele prac opisujących teoretycznie (m.in. [3, 4]) i praktycznie [5], problemy związane z wdrożeniem i działaniem systemów systemów informatycznych koordynujących takie procesy.

Jednym z najpoważniejszych problemów, który wciąż nie doczekał się efektywnego rozwiązania, jest przydział żądanych czasów zakończenia zadań w procesach usługowych. W artykule przedstawiłem próbę rozwiązania tego problemu w oparciu o procedurę, którą wdrożyłem w systemie zarządzania przepływem pracy wspomagającym procesy windykacji należności. W przedstawionej tutaj metodzie wykorzystałem odpowiednio zmodyfikowane algorytmy wykorzystywane w systemach czasu rzeczywistego oraz process miningu. Rozpocznę od krótkiego przedstawienia problemu oraz dostępnych w literaturze jego rozwiązań. Następnie zbuduję odpowiednie modele, które posłużą mi do ustalenia strategii przydziału żądanych terminów zakończenia zadań. W kolejnym kroku pokażę, jak do tych strategii można wykorzystać algorytmy process miningu. Artykuł kończy krótkie podsumowanie.

2. OPIS PROBLEMU

Automatyzacja procesu usługowego rozpoczyna się od zapisu jego *modelu referencyjnego* porządkującego zbiór zadań T pewną funkcją, zwaną *następnikiem* [6], w taki sposób, że przyporządkowuje ona każdemu elementowi $T_i \in T$ zbiór zadań, które mogą zostać po nim wykonane. Wszystkim elementom T_i w powstałej w ten sposób strukturze przypisuje się ograniczenia czasowe, które w dalszej części tego artykułu będę nazywał *żądanymi czasami wykonania zadań* i określał funkcją $dl(T_i)$. Odpowiednio zapis $dl(P_j)$ oznaczać będzie czas, po jakim powinien zakończyć się przebieg procesu j. Opisany powyżej model będę nazywał *statycznym sposobem przydziału* $dl(T_i)$.

Jeżeli któreś z zadań T_i przekroczy przydzielony mu $dl(T_i)$ lub też jeżeli zachodzi uzasadniona obawa, że nie zakończy się ono w terminie, wtedy konieczna jest reakcja *opiekuna procesów* (w badanym systemie – windykatora). Rolę tę może pełnić system komputerowy,

który albo wywołuje odpowiedni, alternatywny scenariusz przygotowany na wypadek przekroczenia czasu przeznaczonego na wykonanie zadania T_i, albo kończy dany przebieg procesu. Niewątpliwą zaletą tego rozwiązania jest szybkość reakcji na zakłócenie, a wadą konieczność opracowania w fazie modelowania wszystkich ewentualnych wyjątków w procesie. Dlatego rolę opiekuna procesu powierza się pracownikom posiadającym największą wiedzę i doświadczenie w organizacji. Każdy z nich staje się odpowiedzialny za monitoring pewnej grupy przebiegów procesu. Opiekun może, jeśli uzna to za konieczne, zmienić marszrutę przebiegu procesu, który jest zapisany w grafie modelu referencyjnego, np. może ominąć jakieś zadanie, zmodyfikować $dl(T_i)$ lub nakazać powtórzenie pewnego zadania T_i (z powodu np. pojawienia się nowych informacji). Takie podejście zapewnia wysoką elastyczność (człowiek może wziąć pod uwagę swoje doświadczenie, odczucia, niepewne informacje), ale wymaga zatrudnienia bardziej wykwalifikowanych, a więc droższych, pracowników. Co gorsza, człowiek potrzebuje zdecydowanie więcej czasu na podjęcie decyzji i odpowiednią reakcję niż system komputerowy. Z tego powodu przy wyznaczaniu liczby podległych opiekunowi przebiegów procesów należy tak ją dobrać, aby zminimalizować liczbę nieobsłużonych przez niego wyjątków.

Kluczem do rozwiązania tego problemu jest likwidacja „fałszywych" alarmów. Mam tu na myśli sytuacje, w których system alarmuje o przekroczeniu żądnego czasu $dl(T_i)$, mimo że wykonanie zadań poprzedzających zadanie T_i zakończyło się przed terminem, a więc w rzeczywistości nie istniało zagrożenie przekroczenia czasu $dl(P_j)$ zakończenia przebiegu procesu j. Tak więc konieczne jest wypracowanie narzędzi, które pozwolą opiekunowi procesu na efektywne monitorowanie podległych mu przebiegów procesu oraz na takie ustalania żądanych czasów wykonywania zadań $dl(T_i)$, aby uwzględniały one zarówno bieżący stan poszczególnych elementów organizacji realizującej proces, jak i ewentualne zmiany zachodzące w otoczeniu tej organizacji. W artykule proponuję metodę rozwiązania tak postawionego problemu.

Przedstawione wyniki uzyskałem w oparciu o system zarządzający przepływem pracy w procesie windykacji masowej. Windykacja masowa polega na odzyskiwaniu niewielkich kwot należności od dużych (kilka tysięcy) grup klientów, najczęściej osób fizycznych. Odzyskiwaniem należności zajmuje się windykator, który prowadzi zwykle równocześnie kilkaset przebiegów procesów. W przebieg procesu, oprócz windykatora, zaangażowane są także: dział prawny (reprezentacja w sprawach sądowych oraz ekspertyzy prawne), dział obsługi korespondencji (zarządzanie przepływem korespondencji w firmie), dział monitoringu płatności (wpłaty na konto od wierzycieli, naliczanie odsetek) oraz dział wywiadu gospodarczego.

3. STAN WIEDZY

Badania rozpocząłem od przeanalizowania, w jaki sposób w najpopularniejszych systemach zarządzania procesami pracy rozwiązany został problem ustalania żądanych czasów zakończenia zadań. Okazało się, że w znakomitej większości [7–10] jedynie w fazie projektowania procesu referencyjnego istnieje możliwość statycznego ustalania czasów zakończenia zadań. Jedynie w [11, 12] istnieje możliwość zmiany predefiniowanych w modelu referencyj-

nym czasów przez opiekuna procesów, aczkolwiek decyzję tę musi on podejmować w oparciu o własne doświadczenie, ponieważ system nie pozwala na dynamiczne ustalanie $dl(T_i)$.

Bogata literatura, m.in. [13–17], dotycząca budowy systemów monitorowania procesów i obsługi wyjątków w procesach pracy, albo opiera skomplikowane systemy decyzyjne na statycznym modelu przydziału żądanych czasów zakończenia zadań, albo skupia się na problemie występowania awarii zasobów realizujących procesy pracy (choroby pracowników, uszkodzenia infrastruktury informatycznej, itp.).

Dziwi niewielka liczba publikacji dotycząca dynamicznego określania żądanych czasów zakończenia zadań. Właściwie warte wspomnienia są tylko dokonania grupy badaczy koreańskich [18–19], którzy w oparciu o sieci PERT oraz systemy kolejkowe próbują określić ścieżkę krytyczną w grafie procesu referencyjnego, by na tej podstawie dokonywać predykacji przekroczenia żądanych czasów zakończenia zadań. Z kolei w [20] opisany został sposób dynamicznego przydziału czasów zakończenia zadań w oparciu o koszty, jakie generuje obsługa wyjątku. Niestety, w obie te prace zakładają *a priori*, że wszystkie wyjątki zapisane zostały w procesie referencyjnym, czyli rolę opiekuna procesu może pełnić komputer.

Termin „process mining" jest pojęciem dość nowym (po raz pierwszy użyto go w [21]). Wynika to przede wszystkim z faktu, że dopiero niedawno pojawił się wspólny standard języka opisu procesów biznesowych wykorzystujący język XML. Ogólnie można wyróżnić dwa kierunki analizy logów z systemu przepływu pracy. Pierwsza grupa metod skupia się na budowie modelu procesu referencyjnego w oparciu o logi procesu, m.in. [22] (obszerne podsumowanie stanu badań [23]). Przykładem takiego rozwiązania jest α-algorytm zaproponowany w [24]. Druga grupa metod stara się tak standaryzować dane o poszczególnych przebiegach procesu, aby móc otrzymać zbiorcze informacje o procesie (np. parametry opisujące czas trwania, wykorzystanie zasobów, rozkłady zaawansowania poszczególnych procesów, etc.). Ten zbiór metod doczekał się już wielu komercyjnych rozwiązań, takich jak SAS Enterprise Miner, HP Process Manager [11] czy też ARIS Process Performance Manager [25].

Proponowana w tym opracowaniu metoda łączy zalety dynamicznego przydziału żądanych czasów zakończenia zadań oraz zalety modeli obsługi wyjątków zbudowanych w oparciu o statyczne ograniczenia czasowe. Do budowy metody zostanie wykorzystałem zmodyfikowany α-algorytm.

4. MODEL PRZEWIDYWANIA WYJĄTKÓW W PROCESIE PRACY

Metodę przewidywania wyjątków w procesie pracy przedstawimy w oparciu o zmodyfikowany przez nas model zaczerpnięty z prac Kao i Garcii-Moliny [26, 27], który autorzy wykorzystali do opisu ogólnego modelu systemu czasu rzeczywistego.

Na podstawie definicji modelu referencyjnego, który przedstawiłem w rozdziale drugim, można powiedzieć, że dowolny proces przebiegu pracy składa się z m zadań $T_1, ..., T_m$, z których każde może być albo niepodzielną częścią procesu, albo może zawierać w sobie n niepodzielnych podzadań wykonywanych równolegle. W tym drugim przypadku do wyznaczania ograniczeń czasowych przyjmuje się czas najdłużej trwającego zadania, a następnie

obliczony na tej podstawie $dl(T_i)$ narzuca się na wszystkie podzadania. Dzięki temu dowolny przebieg procesu można potraktować jako ciąg zadań.

Dowolnemu zadaniu T_i można przypisać zbiór atrybutów opisanych funkcjami (notacja zaczerpnięta z [26]):

$Dl(T_i)$ – żądany czas zakończenia zadania T_i,
$Ex(T_i)$ – przewidywany czas zakończenia zadania T_i,
$Sl(T_i)$ – czas, jaki pozostanie po wykonaniu zadania T_i aż do żądanego czasu jego zakończenia,
$Pr(T_i)$ – przewidywany czas trwania czynności.

Oczywiście czas $sl(T_i)$ można obliczyć dopiero po zakończeniu zadania T_i jako różnicę pomiędzy $dl(T_i)$ i $ex(T_i)$.

Opiekunowi procesu podlega k przebiegów procesu. Celem opiekuna procesu jest doprowadzenie do zakończenia danego przebiegu procesu najpóźniej po czasie $dl(P_j)$. Praca opiekuna procesu polega na podejmowaniu decyzji o wykonaniu, opuszczeniu lub powtórzeniu dowolnego zadania w oparciu o model referencyjny procesu. Podjęcie decyzji wiąże się ze zleceniem wykonania zadania jednemu z działów przedsiębiorstwa oraz z przydzieleniem przez system komputerowy takiemu zadaniu pewnego $dl(T_i)$. Ten przydział dokonuje się według algorytmu, który polega na modyfikacji zapisanego dla tego zadania w procesie referencyjnym $dl(T_i)$ o pewną dodatkową liczbę jednostek czasu. Także opiekun procesu – znając priorytety, jakimi kierują się kierownicy działów szeregując przydzielone im zadania oraz biorąc pod uwagę własne obciążenie – może również modyfikować wyznaczone w ten sposób żądane czasy zakończenia zadań $dl(T_{ij})$.

4.1. SYSTEM PRZEWIDYWANIA PRZEKROCZEŃ ŻĄDANEGO CZASU ZAKOŃCZENIA ZADANIA

Budowa systemu przewidywania przekroczeń żądanego czasu zakończenia zadań przebiegała w trzech etapach. W pierwszym etapie, podobnie jak w większości systemów zarządzania procesami pracy, każdemu zadaniu T_i występującemu w grafie procesu referencyjnego w oparciu o dane historyczne przypisano oczekiwany żądany czas zakończenia zadania $dl(T_i)$. Uruchomienie dowolnego przebiegu procesu P_j powoduje przypisanie temu procesowi atrybutu *pozostały_czas*. W momencie uruchamiania przebiegu procesu wartość ta wynosi 0. Zlecenie wykonania zadania T_i pociąga za sobą obliczenie żądanego czasu $dl(T_i)$, które dokonuje się poprzez modyfikację czasu $dl(T_i)$ wynikającego z grafu procesu referencyjnego o część wartości *pozostały_czas* według procedur zapisanych jednym z wzorów (1) lub (2). O tę samą wartość obniża się *pozostały_czas*. W przypadku gdy wykonanie zadania T_i zakończyło się przed żądanym terminem $dl(T_i)$, czas $sl(T_i)$, jaki pozostanie po wykonaniu zadania T_i, dodawany jest do wartości atrybutu *pozostały_czas* dla danego przebiegu procesu j. W dynamicznym obliczaniu żądanego czasu $dl(T_i)$ wykorzystałem procedury zaproponowane w [26, 27]. Procedury te pierwotnie wykorzystano przy ustalaniu żądanych czasów zakończenia czynności w systemach czasu rzeczywistego.

Obie procedury wymagają wiedzy o przewidywanej liczbie i czasach kolejnych zadań dla danego przebiegu procesu. Informacji tych dostarcza zmodyfikowany α-algorytm, którego działanie opisano w dalszej części artykułu. System przydziela dostępny czas proporcjonalnie do czasu trwania zadania (1) lub proporcjonalnie do liczby pozostałych do wykonania zadań (2):

$$dl(T_i) = dl(T_i) + pozostały\_czas \cdot \frac{ex(T_k)}{\sum_{j=i}^{m} ex(T_j)} \tag{1}$$

$$dl(T_i) = dl(T_i) + pozostały\_czas \cdot \frac{1}{\sum_{j=i}^{m} 1} \tag{2}$$

Praktyka pokazała, że algorytm przydziału żądanych czasów wykonania zadań oparty na wzorze (2) zdecydowanie lepiej rozwiązuje problem przydziału niż algorytm realizujący procedurę zapisaną wzorem (1). Podobnie spostrzeżenia przedstawiono w [26].

W oparciu o te algorytmy zbudowałem system monitorujący działanie procesów. W tym celu uszeregowałem sytuacje, w których mogą wystąpić wyjątki spowodowane przekroczeniem żądanego czasu $dl(T_i)$ ze względu na rosnące zagrożenie przekroczenia żądanego czasu zakończenia zadania.

Najniższy poziom obejmuje przypadki, w których zadanie przekroczyło swój żądany czas wykonania. W takiej sytuacji opiekun procesu ma niewiele do zrobienia i musi oczekiwać na zakończenie zadania. Drugi poziom zawiera zadania, których opóźnienie wynika z faktu zbyt późnej realizacji zadań poprzedzających dane zadania, ale atrybuty *pozostały_czas* są dodatnie. Jest więc jeszcze możliwość zakończenia przebiegu procesu w terminie. Takie przypadki są analizowane przez opiekuna procesów, który możne przyśpieszyć lub opuścić część zadań. Trzeci poziom to przypadki zadań, które przekroczyły swoje ograniczenia czasowe, ale ich atrybuty *pozostały_czas* wynoszą 0. Istnieje wysokie prawdopodobieństwo, że dany przebieg procesu nie zostanie zakończony w terminie. Ostatni, czwarty poziom to sytuacja, w której istnieje prawdopodobieństwo, że zadanie przekroczy swój żądany czas wykonania.

Opiekun procesu na bieżąco otrzymuje informacje – tzw. alerty – o pojawianiu się wyjątków w procesie pracy wraz z poziomami zagrożenia, na których one wystąpiły. Opiekun rozwiązuje problemy w kolejności ich wag, tzn. najpierw rozpatruje alerty z poziomu czwartego, potem z trzeciego itd. Dzięki temu podejściu ewentualne opóźnienia w reakcji opiekuna procesu – związane na przykład z jego obciążeniem – w niewielkim stopniu wpływają na opóźnienie działania całego systemu.

Jak widać, w przedstawionym powyżej systemie ustalania żądanych czasów zakończenia zadań oraz monitorowania procesów, konieczna jest dokładna wiedza o przewidywanej liczbie i czasach kolejnych zadań dla danego przebiegu procesu oraz określenie prawdopodobieństwa, że zadanie przekroczy swój czas wykonania. Do wykonania tych zadań wykorzystałem narzędzia process miningu.

4.2. ZASTOSOWANIE PROCESS MININGU DO OSZACOWANIA ODPOWIEDNICH PARAMETRÓW MODELU

Główny problem związany z oszacowaniem odpowiednich parametrów modelu na podstawie logów systemu wynika z dużej elastyczności badanego procesu.

Moim priorytetem było stworzenie prostego w obsłudze systemu, dlatego szacując parametry modelu, przyjąłem mocne założenia upraszczające. I tak przy ustalaniu żądanych czasów zakończenia zadań założyłem, że czas ten nie zależy od pozostałych czasów w danym przebiegu procesu. Następnie dla każdego zadania zbudowałem histogram rozkładu czasów wykonania. W kolejnym kroku dobrałem spośród typowych rozkładów prawdopodobieństwa taki, który najlepiej dopasowywał się do wszystkich danych. Rozkładem tym był rozkład wykładniczy. Przyjąłem, że średni czas wykonania zadań równy jest średniemu czasowi wykonania zadania z rozkładu wykładniczego.

Do ustalenia określonych powyżej parametrów wykorzystano zmodyfikowany α-algorytm. Algorytm ten został pierwotnie wymyślony do budowy struktury modelu referencyjnego procesu w oparciu o logi systemu zapisane w formacie XPDL.

Zgodnie z definicją przedstawioną powyżej, każdy przebieg procesu jest pewnym ciągiem zadań $P_j = (T_i)_m$. Niech T_a oznacza ostatnie wykonane zadanie w przebiegu procesu. Naszym celem jest określenie $dl(T_{a+1})$. W tym celu musimy określić oczekiwaną liczbę zadań pozostałą do końca procesu. Zdefiniuję teraz na ciągu opisującym szereg procesu funkcję $\beta(P_j) = \{(T_1, T_2), (T_2, T_3), ..., (T_{a-1}, T_a)\}$, przekształcającą dowolny przebieg procesu w zbiór dwójek: poprzednik, następnik. Oprócz tego niech funkcje $first((T_{k-1}, T_k)) = T_{k-1}$, a $last((T_{k-1}, T_k)) = T_k$.

Najpierw z badanego ciągu P_j usuwamy wszystkie elementy spełniające warunek $first((T_{k-1}, T_k)) = last((T_{k-1}, T_k))$, dla $k = 1, ..., a$. W ten sposób oczyszczamy ciąg z powtórzeń zadań. Powstały w ten sposób proces nazywał będę w dalszej części *procesem bazowym*.

W zakończonych, historycznych przebiegach procesów oczyszczonych według tego samego algorytmu poszukuję teraz przebiegów podobnych. Metryka określająca podobieństwo została określona w sposób następujący

$$met(P_j, P_i) = \frac{1}{a} \sum_{k=1}^{a} b_{ijk} \tag{3}$$

gdzie

$$b_{ijk} = \begin{cases} 1, & T_{ik} = T_{jk} \\ 0, & \text{inaczej} \end{cases} \tag{4}$$

W oparciu o badania empiryczne przyjąłem, że podobieństwo pomiędzy tymi ciągami jest wystarczające, jeżeli $met(P_i, P_j) \geq 0{,}7$. Dla każdego ciągu podobnego do procesu bazowego obliczana jest wartość $oczek\_dl_i = m_j - a$. Teraz ze wszystkich unikalnych wartości

$oczek\_dl_i$ obliczana jest wartość średnia, która wykorzystywana jest we wzorze (2) jako liczba zadań pozostałych do wykonania. Nie zdecydowałem się na wykorzystanie tutaj średniej ważonej licznością danej wartości $oczek\_dl_i$ ponieważ zauważyłem, że wartości te obarczone są większym błędem obliczonym jako średnie odchylenie bezwzględne pomiędzy wartością rzeczywistą i oczekiwaną.

Łatwo przekształcić wzór (1) tak, aby zamiast konieczności wyznaczania wartości poszczególnych czasów pozostałych do końca procesu wyznaczyć czas pozostały do końca procesu. W tym celu dla wszystkich przebiegów procesów posiadających odpowiedni poziom podobieństwa wyznaczyłem wartość $oczek\_time_i$, a następnie z wartości unikalnych obliczyłem wartość średnią, która podobnie jak $oczek\_dl_i$ była lepszą miarą niż wartość ważona.

Ustalenie przez system prawdopodobieństwa przekroczenia czasu wykonania zadania dokonywane jest według następujących kroków. W oparciu o historyczne informacje dla kolejnych zadań określane są średnie czasy wykonania zadań i oczekiwania zadań w kolejce. Następnie, poprzez ustalenie, jak bieżący stan zadania (zadanie oczekuje w kolejce lub jest realizowane) koresponduje ze średnimi stanami i czasami, system szacuje prawdopodobieństwo przekroczenia przez zadanie żądanego czasu jego wykonania.

Główna wada tego rozwiązania polega na pominięciu bieżącego obciążenia systemu. Zdecydowanie lepszym rozwiązaniem byłoby wykorzystanie narzędzi symulacyjnych, które – korzystając z informacji zawartych w systemie zarządzania procesem pracy – symulują na bieżąco działanie systemu. Dla przykładu: programy symulacyjne Expect [24] lub Protos [10] przechowują wszystkie informacje o historii oraz o bieżącym stanie systemu w języku XPDL, a także udostępniają interfejsy, które pozwalają na bieżące załadowanie danych do systemu przewidywania przekroczeń żądanego czasu zakończenia zadań i przeprowadzenie symulacji procesu w oparciu o aktualny stan systemu. Niestety wysokie koszty wprowadzenia takiego rozwiązania, szczególnie koszty oprogramowania, czynią te systemy niedostępne dla małych firm.

5. PODSUMOWANIE

Systemy zarządzania procesami pracy składają się z czynności wykonywanych przez ludzi lub systemy komputerowe. W wielu przypadkach czynności posiadają ograniczenia czasowe, których przekroczenie wymaga rozwiązania – obsłużenia tzw. wyjątków w procesie. Dążenie do minimalizacji pojawiania się wyjątków w procesach pracy obniża koszty realizacji całego procesu.

W tym artykule przedstawiono model dynamicznego ustalania strategii przydziału żądanych czasów wykonania zadań oraz strategii monitoringu procesów. Zaproponowany system działa w oparciu o rozkład dostępnego naddatku pracy. Praktyka pokazała, że zaprezentowane w artykule metody pozwalają na obniżenie liczby wyjątków w procesie o kilkanaście procent w stosunku do zadań ze statycznymi ograniczeniami czasowymi.

Literatura

[1] Moore C.: *Evaluation Framework: Workflow or Workware?* White paper, Enix Consulting Ltd., http://www.enix.co.uk/workware.htm, 2003

[2] Weiss G.: *Multiagent systems: a modern approach to distributed modern approach to artificial intelligence*. Cambridge, Massachusetts, London, England, The MIT Press 1999

[3] Miers D., Hutton G.: *The Business Case for Case Handling*. White paper, Enix Consulting Ltd., http://www.enix.co.uk/caseman.htm, 1997

[4] Van der Aalst W.M.P., Weske M., Grunbauer D.: *Case Handling: a new paradigm for business process support*. Data & Knowledge Engineering, t. 53, 2005, s. 129

[5] Gaweł B.: *Metoda kontroli elastycznych procesów w systemach zarządzania obiegiem pracy.* Komputerowo zintegrowane zarządzanie, t. 1, WNT 2005, s. 341

[6] Casati F., Ceri S., Pernici B., Sheth A.: *Workflow evolution*. ER'96 International Conference, Berlin 1996, s. 126

[7] *TeamWare Flow. Collaborative workflow system for the way people work*. P.O. Box 780, FIN-00101, Helsinki, Finland

[8] *InConcert. Technical product overview. XSoft, a division of Xerox*. 3400 Hillview Avenue, Paolo Alto, CA 94304, http://www.xsoft.com

[9] *Ultimus. Workflow suite. Business workflow automation*. 4915 Waters Edge Dr., Suite 135, Raleigh

[10] *Pallas Athena. Case Handling with FLOWer: Beyond Workflow.* Positioning paper, http://www.pallas-athena.com/downloads/eng_flower/flowerwp.pdf, 2000

[11] *HP Process Manager Process Design Guide*, wersja 4.4, www.hp.com

[12] *Proces Product Watch, I-Flow ver. 4.01*. Workflow Management Technology Report, Enix Consulting Ltd., http://www.enix.co.uk

[13] Chiu D., Li Q., Karlapalem K.: *A Meta Modeling Approach to Workflow Management, Systems Supporting Exception Handling*. Information Systems, t. 24, nr 2, 1999, s. 159

[14] Georgakopoulos D., Hornick M., Sheth A.: *An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure*. Distributed and Parallel Databases, t. 3, 1995, s. 119

[15] Hwang S.Y., Tang J.: *Consulting Past Exceptions to Facilitate Workflow Exception Handling*. Decision Support Systems, t. 37, nr 1, 2004, s. 49

[16] Klein M., Dellarocas C.: *A Knowledge-Based Approach to Handling Exceptions in Workflow Systems*. Journal of Computer-Supported Collaborative Work, t. 9, nr 3–4, 2000, s. 399

[17] Luo Z., Sheth A., Kochut K., Miller J.; *Exception Handling in Workflow Systems*. Applied Intelligence, t. 13, nr 2, 2000, s. 125

[18] Son J.H., Kim M.H.: *Improving the Performance of Time-Constrained Workflow Processing*. Journal of Systems and Software, t. 58, 2001, s. 211

[19] Son J.H., Kim J.S., Kim M.H.: *Extracting the workflow critical path form the extended well-formed workflow schema*. Journal of Computer And System Sciences, t. 70, 2005, s. 86

[20] Eder J., Panagos E., Rabinovich M.: *Time Constraints in Workflow Systems*, Proceedings of the 11th International Conference on Advanced Information Systems Engineering, 1999, s. 286

[21] Agrawal R., Gunopulos D., Leymann F.: *Mining Process Models from Workflow Logs*. Sixth International Conference on Extending Database Technology, 1998, s. 469

[22] Pinter S., Golani M.: *Discovering workflow models from activities' lifespans*. Computers in Industry, t. 53, 2004, s. 283
[23] Aalst van der W.M.P., Dongen van B., Herbst J., Maruster L., Schimm G., Weijters A.: *Workflow mining: A survey of issues and approaches*. Data & Knowledge Engineering, t. 47, 2003, s. 237
[24] Aalst van der W.M.P., Weijters A., Maruster L.: *Workflow Mining: Discovering Process Models from Event Logs*. IEEE Transactions on Knowledge and Data Engineering (TKDE), 2003, s. 178
[25] *IDS Scheer.* ARIS Process Performance Manager (ARIS PPM), http://www.ids-scheer.com, 2002
[26] Kao B., Garcia-Molina H.: *Deadline assignment in a distributed soft real-time systems*. Proceedings of the 13th International Conference of Distributed Computing Systems, 1993, s. 428
[27] Kao B., Garcia-Molina H.: *Subtask deadline assignment for complex distributed soft real-time tasks*. Technical Report 93-1491, Stanford University 1993