CONTEXT-AWARE WORKFLOW MANAGEMENT FOR INTELLIGENT NAVIGATION APPLICATIONS IN PERVERSIVE ENVIRONMENTS

FEILONG TANG\textsuperscript{1}, ILSUN YOU\textsuperscript{2}, MINYI GUO\textsuperscript{1}, SONG GUO\textsuperscript{3}

\textsuperscript{1}Department of Computer Science and Engineering
Shanghai Jiao Tong University
Shanghai 200240, China

\textsuperscript{2}School of Information Science
Korean Bible University
South Korea

\textsuperscript{3}School of Computer Science and Engineering
The University of Aizu
Fukushima 965-8580, Japan

ABSTRACT—Pervasive computing is a user-centric distributed computing paradigm, allowing users to access their preferred services even while moving around. To make this vision a reality, context-aware workflow management is one of key issues because the context of pervasive applications is changing instantly. In this paper, we propose a context model for intelligent navigation applications and then present a context-aware workflow management algorithm (CAWM) which can dynamically adjust workflow execution behaviours based on current context information. The correctness of the CAWM algorithm has been also verified theoretically by formulating it as a Petri-net model. Furthermore, the proposed context model and workflow management algorithm can apply to other applications by simply revising the corresponding context structures only.

Key Words: Intelligent computing, pervasive computing, workflow, context awareness, Petri nets

1. INTRODUCTION

PERVASIVE computing is a new distributed computing paradigm that provides mobile users their preferred services while moving around. Due to the high mobility of users, pervasive software runs in an extremely heterogeneous and dynamic environment, where many kinds of network nodes and communication protocols co-exist, and network topology and bandwidth change frequently. As a result, context awareness and context-driven self-adaptation are key

\* This work was supported by the National Natural Science Foundation of China (Grant Nos. 60773089, 60533040 and 60725208) and the National High Technology Research and Development Program (863 Program) of China (Grant Nos. 2006AA01Z172 and 2008AA01Z106) and Shanghai Pujiang Program (Grant No.07pj14049).
characteristics distinguishing pervasive environments from traditional distributed systems. Pervasive software has to be aware of the context changes for dynamically adjusting execution behaviors and supporting seamless roaming to make the user-centric vision a reality. Along with this direction, there have been many active application-driven researches [1-5].

Context-aware workflow management is an important enabling technology to achieve the desired goals of pervasive computing. It should consider the characteristics of both static and dynamic topologies, wired and wireless subnets, mobile and disconnected users, etc. However, none of existing workflow management proposals can sufficiently address the combined and new requirements for pervasive applications.

Context-aware navigation service is one of significant pervasive applications. In this paper, we consider a context-aware workflow management for pervasive navigation in a campus, in which there are many libraries, restaurants and bus stops, each of them in a different location. Alex plans to first visit a library for borrowing some books (A1), then have a lunch in a restaurant (A2), and finally take a bus to go home (A3). Figure 1 illustrates such a tour, where A1, A2 and A3 are these three sub-activities in Alex’s campus tour. We assume he is not familiar with the campus environment and just holds a PDA (personal digital assistant) with GPS (global positioning system) function. To make the tour convenient and efficient, the pervasive campus should provide him an intelligent navigation based on current context. When Alex wants to take a bus, for example, a campus navigation system should find the shortest bus stop from his current location and send the navigation information (e.g., routing path) to his PDA. Furthermore, the navigation system should automatically transmit requests and responses via qualified network links that have stable connection and sufficient communication bandwidth. From the above scenario, we find that navigation applications are highly context-dependent, especially in the following dimensions:

- **current location.** When Alex hopes to have a lunch, for example, the navigation system should find a restaurant closest to his current location.
- **connected network.** The navigation system has to select a qualified network to transmit requests and responses for Alex, based on current network topology and bandwidth.

Workflow management aims at providing the process transparency for a group of activities. As a result, workflow management for intelligent navigation applications in wireless network environments has to be aware of current context, e.g. network connectivity, bandwidth and user location, for reliable and intelligent navigation services. So, context-aware workflow management is a valuable and important service for mobile and pervasive environments.

![Figure 1. A tour in a pervasive campus.](image)

This paper is motivated to solve such an issue for providing a context-aware management service characteristic to navigation applications in pervasive campuses. We propose a context model for the campus navigation and then present a context-aware workflow management algorithm (CAWM). The correctness of the CAWM algorithm is also verified theoretically by formulating it as a Petri-net model.

The remainder of this paper is organized as follows. Section 2 provides an overview of related work. Section 3 proposes a context model for campus navigation applications. Section 4 presents
a context-aware workflow management algorithm CAW. Section 5 validates the correctness of CAWM through Petri nets. The implementation and case study are presented in Section 6. Finally, Section 7 concludes this paper with a discussion on our future work.

2. RELATED WORK

Workflow management technologies have been extensively researched in recent years. From the perspective of adaptation to execution environments, existing workflow models fall into two categories: non-context-aware workflow management [6-10] and context-aware workflow management [11-13] (see Table I).

Non-context-aware workflow management models are not adaptive to the context of workflow execution. DISCOBOLE [6] is a service architecture for interconnecting workflow processes. This model supports heterogeneous workflow process interconnection for inter-enterprise cooperation, where several heterogeneous workflow management systems co-exist. McRunjob [8] is a grid workflow manager used to generate large number of production processing jobs in high energy physics. It converts core metadata into jobs submitted in a variety of environments. Krishnan et al. [9] presented a workflow framework for Grid services (GSFL) to manage workflows within the Open Grid Services Architecture (OGSA) framework. The GSFL focuses on the definition of an XML-based workflow language and the description of an implementation of a workflow engine. Nichols et al.[10] proposed a model for autonomic workflow management (AWM) in Grids. The model integrated dynamic fault tolerance mechanism with Grid-based workflow management architecture, providing awareness and resilience to failures. Task net (TN) [7] is transactional workflow model based on colored Petri net. It defined a workflow specification language that can express the dependency relationship of task states and enable users to express both the transaction- and application-oriented requirements.

Table I. Some related workflow models

<table>
<thead>
<tr>
<th>Models</th>
<th>Main function</th>
<th>Context-aware</th>
<th>Applicable environments</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISCOBOLE</td>
<td>Process cooperation</td>
<td>No</td>
<td>Heterogeneous systems</td>
</tr>
<tr>
<td>McRunjob</td>
<td>Job generation</td>
<td>No</td>
<td>High energy physics</td>
</tr>
<tr>
<td>GSFL</td>
<td>Grid workflow management</td>
<td>No</td>
<td>Grids</td>
</tr>
<tr>
<td>AWM</td>
<td>Fault tolerance</td>
<td>No</td>
<td>Grids</td>
</tr>
<tr>
<td>TN</td>
<td>Transactional workflow</td>
<td>No</td>
<td>Distributed systems</td>
</tr>
<tr>
<td>CAWE</td>
<td>Web services workflow</td>
<td>Yes</td>
<td>Web services environments</td>
</tr>
<tr>
<td>CAWDRA</td>
<td>Automatic resource allocation</td>
<td>Yes</td>
<td>Distributed healthcare systems</td>
</tr>
<tr>
<td>ACAS</td>
<td>Autonomous context-aware service</td>
<td>Yes</td>
<td>Mobile computing</td>
</tr>
</tbody>
</table>

There also have been reports on context-aware workflow management. CAWE (Context-Aware Workflow Execution) [11] is a framework for management of context-aware Web Services-based workflow systems. The CAWE framework introduced a hierarchical workflow representation, specifying context-dependent actions and a declarative specification of the conditions determining how to select the appropriate context-dependent workflow part at runtime. The context in CAWE proposal is based on Web Services systems, with no consideration of the features of wireless and mobile environments. In CAWDRA (Context-aware Workflow Driven
Resource Allocation) [12], Hsieh presented a workflow model to describe the medical service processes in healthcare systems, with the goal of automatic resource allocation. ACAS (Autonomous Context-Aware Services) [13] is a model for the discovery and execution of services on connected and partially autonomous mobile devices. Discovery and execution procedures are sensitive to context changes, by running a service registry on the mobile device itself. However, this framework did not use workflow technologies to manage the discovery and execution of services. In conclusion, the above workflow systems have not yet included sufficient functions to support context-aware services for pervasive computing. Distinguishing from these proposals, this paper investigates, by the first time, how to automatically manage workflows for navigation applications in a context-aware way in pervasive environments.

3. CONTEXT MODEL FOR CAMPUS NAVIGATION

Context awareness enables systems automatic configurations and adjustments to provide better services for users. Context is the representation of the information that is relevant to the individuals and devices within the activity space [18]. A context adaptive system typically enables the user to maintain a certain application (in different forms) while roaming between different wireless access technologies, locations, devices and even simultaneously executing everyday tasks. Context-aware campus navigation systems focus on providing navigation information (e.g., text, photo) of targets for users with mobile devices (e.g., PDA). As a result, the following context dimensions have to be considered for campus navigation applications (see Table II).

- **Person**: profile, preferences and requirements of a user,
- **Location**: longitude and latitude (or relative position) of the user,
- **Network**: connection and performance of networks, and
- **Mobile device**: computing and storage capacity of devices

### Table II. Context of pervasive campus

<table>
<thead>
<tr>
<th>Entity</th>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person</td>
<td>Name</td>
<td>Name description</td>
</tr>
<tr>
<td></td>
<td>Sex</td>
<td>Male, female</td>
</tr>
<tr>
<td></td>
<td>Age</td>
<td>Value of age</td>
</tr>
<tr>
<td></td>
<td>Identity</td>
<td>Teacher, student, guest etc.</td>
</tr>
<tr>
<td></td>
<td>Requirement</td>
<td>Requirement description</td>
</tr>
<tr>
<td></td>
<td>Preference</td>
<td>Behaviour preference</td>
</tr>
<tr>
<td></td>
<td>Connectivity</td>
<td>Connected, disconnected</td>
</tr>
<tr>
<td></td>
<td>Bandwidth</td>
<td>High, medium, low</td>
</tr>
<tr>
<td></td>
<td>Cost</td>
<td>Expensive, cheap, free</td>
</tr>
<tr>
<td></td>
<td>Stability</td>
<td>Good, medium, bad</td>
</tr>
<tr>
<td>Network</td>
<td>Available battery</td>
<td>Full, half, low</td>
</tr>
<tr>
<td></td>
<td>Available data</td>
<td>Available, unavailable</td>
</tr>
<tr>
<td></td>
<td>Computing capacity</td>
<td>High, medium, low</td>
</tr>
<tr>
<td></td>
<td>Available memory</td>
<td>Full, half, low</td>
</tr>
<tr>
<td></td>
<td>Available cache</td>
<td>Full, half, low</td>
</tr>
<tr>
<td>Location</td>
<td>Longitude</td>
<td>Value of longitude</td>
</tr>
<tr>
<td></td>
<td>Latitude</td>
<td>Value of latitude</td>
</tr>
</tbody>
</table>
We abstract the context information using entities, attributes and relationships among entities, and model them in a Entity Relationship (ER) diagram. Note that some attributes (e.g., preference) can not be directly abstracted from entity profiles and data mining-like technologies will be used for this purpose. The graphic context model for pervasive campus navigation is shown in Figure 2, where we only illustrate attributes of the entity Person. Attributes of other entities can be easily included in the same way according to Table II. An entity is fully captured by means of its attributes. For example, a user Alex can be described like Person.Name = “Alex,” Person.Identity = “student,” Person.Age = “20,” Person.Sex = “male,” Person.Requirement = “he needs to visit a library, a restaurant and a bus stop,” and Person.Preference = “he likes computer books.”

On the other hand, there are relationships between two entities. For example, the relationship denoted as “Person n m Location” in Figure 2 means that a person may appear at many locations while a location can be visited by many persons.

**Figure 2. Context model for pervasive campus navigation.**

### 4. CONTEXT-AWARE WORKFLOW MANAGEMENT

In this section, we define Petri net model for a context-aware workflow, and then present the context-aware workflow management algorithm CAWM.

#### 4.1 Context-aware workflow based on Petri nets

A Petri net is an effective modeling tool applicable to many systems, especially to distributed and parallel systems. It can model systems’ events, conditions and relationships, expressing control flow as well as data flow in a set of activities. The occurrence of these events may change the state of the system, causing some of the previous conditions to cease holding and other conditions to begin to hold [14-17]. We model a context-adaptive pervasive workflow in Petri nets as follows.

**Definition 1.** A context-adaptive workflow (CAW) is a 7-tuple CAW=(P,T,I,O,IT,CO, M0), where

- \( P = \{p_1, p_2, \ldots, p_m\} \) is a finite set of places,
- \( T = \{t_1, t_2, \ldots, t_n\} \) is a finite set of transitions such that \( P \cap T = \emptyset \) and \( T \cap P = \emptyset \),
- \( I \subseteq P \times T \) is a finite set of input arcs from \( p_i \) to \( t_j \) (\( 1 \leq i \leq m, 1 \leq j \leq n \))
- \( O \subseteq T \times P \) is a finite set of output arcs from \( t_j \) to \( p_i \) (\( 1 \leq i \leq m, 1 \leq j \leq n \))
IT=\{IT_1, IT_2, \ldots, IT_k\} is a finite set of user’s interested targets,

Table III. Workflow composite operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Priority</th>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANY</td>
<td>1</td>
<td>ANY(m, E_1, E_2, \ldots, E_n)</td>
<td>M out of n events E_i (1 \leq i \leq n) occur(s)</td>
</tr>
<tr>
<td>EXC</td>
<td>1</td>
<td>EXC(E_1, E_2, \ldots, E_n)</td>
<td>Only 1 out of n events E_i (1 \leq i \leq n) occurs</td>
</tr>
<tr>
<td>NOT</td>
<td>2</td>
<td>NOT E_1</td>
<td>E_1 does not occur</td>
</tr>
<tr>
<td>AND</td>
<td>3</td>
<td>E_1 AND E_2</td>
<td>Both E_1 and E_2 occur</td>
</tr>
<tr>
<td>ALL</td>
<td>3</td>
<td>ALL(E_1, E_2, \ldots, E_n)</td>
<td>All events E_i (1 \leq i \leq n) occur</td>
</tr>
<tr>
<td>OR</td>
<td>4</td>
<td>E_1 OR E_2</td>
<td>At least one of E_1 and E_2 occur(s)</td>
</tr>
<tr>
<td>REP</td>
<td>5</td>
<td>REP E_1 _n</td>
<td>E_1 occurs n times</td>
</tr>
<tr>
<td>PRE</td>
<td>6</td>
<td>E_1 PRE E_2</td>
<td>E_1 occurs before E_2</td>
</tr>
<tr>
<td>SEQ</td>
<td>6</td>
<td>SEQ(E_1, E_2, \ldots, E_n)</td>
<td>E_1 PRE E_2 PRE \ldots PRE E_n</td>
</tr>
</tbody>
</table>

CO is a finite set of context information, and 
M_0 is the initial marking.

CAWM manages workflows in the event-driven way, including atomic and composite events. We defined five types of atomic events: context change event, method call event, state transition event, (absolute and relative) time event and user-defined event. The composite events are composed of the atomic events or lower-level composite events, using the composite operators with specified priorities listed in Table III, where E_i (1 \leq i \leq n) indicates an atomic or a composite event. The highest priority is 1.

Context-Aware Workflow Management Algorithm
Input: a set of expected targets (IT);
Output: navigation information of IT;
{ i=1;
  while (IT \neq \emptyset) do
    C_i= candidates corresponding to IT_i;
    j=1;
    suc=true;
    while (C_i \neq \emptyset) and (suc) do
      k=1;
      repeat
        T = bandwidth of network N_k;
        if (T is low) \_k++;
        until (\_k>K) or (T is high or medium);
        send request for C_{i,j} to server via network N_{\_k};
        wait for navigation information of IT_{i,j};
        if (the user does not satisfy) {
          C_i=C_i\{-C_{i,j}\};
          j++;
        } else
        suc=false;
        endwhile
    suc=false;
  endwhile
  IT=IT-\{IT_i\};
  i++;
}
4.2 Context-aware workflow management algorithm

Context-aware campus navigation targets on providing expected navigation information based on current context information. Let \( IT = \{ IT_i \ | \ IT_i \) is one of expected targets, \( 1 \leq i \leq n \) \} be the set of a user’s interested targets (e.g., library, restaurant) in a campus. \( IT \) is an ordered set with the dependence \( IT_i < IT_{i+1}, \) i.e., specifically, \( IT_{i+1} \) is not executed until \( IT_i \) finishes. \( C_i \) is a set of candidates corresponding to an interested target \( IT_i \) such that \( C_i = \{ C_{i,j} \ | \ C_{i,j} \) is a candidate of \( IT_i, \ 1 \leq j \leq m \} \). \( C_i \) is also an ordered set, for example, \( C_{i,j} \) is closer to a user’s current location than \( C_{i,j+1} \). Note that current location is the latest position where the user is located when last sub-activity finishes. At the beginning of a campus navigation workflow, for example, current location means the place where Alex initiates a request just after Alex arrives at a library. In other words, current location refers to that library. Moreover, we assume a request can be sent to servers via \( K \) different network links \( N = \{ N_k \ | \ N_k \) is a network link between the request node and the server, \( 1 \leq k \leq K \} \).

Our context-aware workflow management algorithm (CAWM) adopts an event-context-action (ECoA) mechanism. As illustrated in Figure 3, the algorithm can dynamically adjust workflow execution policies based on current context information. Attributes of each entity can take the values given in Table II. The algorithm requires that the value of a network bandwidth for transmitting workflow requests and results must be high or medium. Figure 4 illustrates the execution flow of the algorithm CAWM.

```
Figure 3. Context-aware workflow management algorithm for campus navigation (CAWM).

Figure 4. Flow diagram of the CAWM.

5. CORRECTNESS VALIDATION OF THE ALGORITHM CAWM

In this section, we formulate the CAWM algorithm as a graphic Petri net model and then validate its correctness using the reachability tree analysis technology of Petri nets.
```
5.1 Activity diagram and graphic Petri net

Compared with other validation tools, Petri net can model systems or protocols with instinctive graphics. Before illustrating graphic Petri nets of the CAWM, we introduce possible workflow execution sub-structures, as shown in Figure 5, where \( A_i \) denotes a sub-activity in a workflow, and \( p_i \) and \( t_i \) mean corresponding system states and actions, respectively. In each sub-figure, the left is the diagram of a sub-activity and the right is corresponding Petri nets. To express selective executions, in particular, we introduce a **selective transition** concept. Let \( O(t) \) be the set of output arcs of a transition \( t \) such that \( O(t)=\{O(t)^{(i)} | O(t)^{(i)} \text{ is one of output arcs, } 1 \leq i \leq n\} \), where \( n \) is the number of \( t \)’s output arcs.

**Definition 2.** A transition \( t \) is **selective** if (1) any \( O(t)^{(i)} \in O(t) \) associates with a condition \( O(t)^{(i)}.\text{cond}(i) \), and (2) when a firing occurs, only \( O(t)^{(i)} \) with \( O(t)^{(i)}.\text{cond}(i)=\text{true} \) \( (1 \leq i \leq n) \) is fired.

The **selective transition** concept extends modeling ability of Petri nets by introducing firing conditions for each output arc. Let a selective transition \( t_2 \) in Figure 5(2) denote the data transmission via a network link \( N_k \) and the corresponding conditions be network performance. When \( t_2 \) is fired, \( O(t)^{(2)} \) is actually fired only if the network link \( N_k \) is qualified, i.e., \( O(t)^{(2)}.\text{cond}(2)=\text{true} \). Otherwise, only \( O(t)^{(1)} \) is fired when the network link \( N_k \) is not qualified.

![Diagram](image-url)
(7) Exclusive resource: workflows A and B exclusively use the resource R

Figure 5. Workflow execution structures and corresponding Petri nets.

5.2 Petri nets of CAWM

Our workflow management CAWM is responsible for finding the navigation information of targets, requested by a user, based on his/her current context. Generally, there are multiple candidates for a requested target. As a result, CAWM first selects a qualified network link between the user and a server, then finds a candidate for a requested target IT, and finally gets the navigation information of IT. Figure 6 illustrates such an activity diagram of CAWM, where each activity respectively denotes: (1) A - CAWM initiates IT; (2) A - CAWM finds a candidate; (3) A - CAWM finds a qualified network link; (4) A - CAWM gets navigation information of the candidate; (5) A - CAWM judges whether the candidate is satisfied; and (6) A - CAWM judges whether the workflow is finished.

Figure 6. Activity diagram of CAWM.

Figure 7 depicts the graphic Petri net of the CAWM, a simplified PN-CAWM, where places refer to states of sub-activities while transitions mean the execution of corresponding actions (see Table IV). Three selective transitions t, t and t denote inspecting network status, checking the navigation information for a target IT, and judging whether a workflow finishes, respectively. More specifically, O(t).cond(1) denotes that network N has good performance (e.g., enough bandwidth), O(t).cond(1) denotes that a user does not accept the candidate C, and O(t).cond(1) denotes a workflow does not finish. Similarly, O(t).cond(2), i = 3, 5 and 6, denote otherwise, respectively.

According to Figure 7, we have the Petri net PN-CAWN=(P,T,I,O,IT,CO,M), where P={p, p, ... , p}; T={t , t , ..., t }; I={I(t ), I(t ), I(t ), I(t ), I(t ), I(t )}; O={O(t ), O(t ), O(t ), O(t ), O(t ), O(t )}. Note that for a selective transaction t, only qualified output arcs are actually fired when t is fired;
Finally, IT is the set of visiting targets in a workflow, for example, IT={library, restaurant, bus stop} in Alex’s tour; and CO is the set of context during the execution of a workflow.

### Table IV. Places and transitions of PN-CAWM

<table>
<thead>
<tr>
<th>Elements</th>
<th>Status</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>Active(IT&lt;sub&gt;i&lt;/sub&gt;)</td>
<td>Sub-activity IT&lt;sub&gt;i&lt;/sub&gt; is initiated</td>
</tr>
<tr>
<td>p2</td>
<td>Executing(IT&lt;sub&gt;i&lt;/sub&gt;)</td>
<td>IT&lt;sub&gt;i&lt;/sub&gt; is executing</td>
</tr>
<tr>
<td>p3</td>
<td>Found(C&lt;sub&gt;i,j&lt;/sub&gt;)</td>
<td>Candidate C&lt;sub&gt;i,j&lt;/sub&gt; is found</td>
</tr>
<tr>
<td>p4</td>
<td>Checking(N&lt;sub&gt;k&lt;/sub&gt;)</td>
<td>N&lt;sub&gt;k&lt;/sub&gt; is been checking</td>
</tr>
<tr>
<td>p5</td>
<td>Qualified(N&lt;sub&gt;k&lt;/sub&gt;)</td>
<td>N&lt;sub&gt;k&lt;/sub&gt; is qualified</td>
</tr>
<tr>
<td>p6</td>
<td>Finished(IT&lt;sub&gt;i&lt;/sub&gt;)</td>
<td>A user gets navigation information of IT&lt;sub&gt;i&lt;/sub&gt;</td>
</tr>
<tr>
<td>p7</td>
<td>Satisfied(IT&lt;sub&gt;i&lt;/sub&gt;)</td>
<td>A user satisfies the navigation information</td>
</tr>
<tr>
<td>p8</td>
<td>End</td>
<td>A workflow finishes</td>
</tr>
<tr>
<td>t1</td>
<td>IT&lt;sub&gt;i&lt;/sub&gt; is initiated</td>
<td></td>
</tr>
<tr>
<td>t2</td>
<td>CAWM finds C&lt;sub&gt;i,j&lt;/sub&gt; for IT&lt;sub&gt;i&lt;/sub&gt;</td>
<td></td>
</tr>
<tr>
<td>t3</td>
<td>CAWM inspects whether N&lt;sub&gt;k&lt;/sub&gt; is qualified</td>
<td></td>
</tr>
<tr>
<td>t4</td>
<td>CAWM requests navigation information of C&lt;sub&gt;i,j&lt;/sub&gt; via N&lt;sub&gt;k&lt;/sub&gt;</td>
<td></td>
</tr>
<tr>
<td>t5</td>
<td>A user satisfies navigation information for IT&lt;sub&gt;i&lt;/sub&gt; or not</td>
<td></td>
</tr>
<tr>
<td>t6</td>
<td>CAWM checks whether a workflow finishes</td>
<td></td>
</tr>
</tbody>
</table>

### 5.3 Correctness validation of the CAWM

Petri nets can analyze the behavioral properties, which depend on the initial marking, including reachability, boundedness, liveness, coverability, reversibility, persistence and so on. For a bounded Petri net, the coverability tree is called the reachability tree and all above problems can be solved by the reachability tree [15]. Peterson [14] also pointed out that in Petri nets, many questions can often be reduced to the reachability problem.

![Petri net diagram](add petri net image here)
A marking $M$ in a Petri net is an assignment of tokens to each place. The movement of tokens expresses the status conversion of workflow systems. $M$ is reachable from another marking $M'$ if $M'$ may be transformed to $M$ through a sequence of firings. The set of all possible markings reachable from a initial marking $M_0$ in a Petri net $(N, M_0)$ is expressed by $R(M_0)$. For the PN-CAWM, $M=(m_1, m_2,\ldots, m_8)$ is a marking and $m_i$ indicates the number of tokens in place $p_i$ ($i=1,2,\ldots,8$).

The boundedness and liveness of a Petri net are often used as correctness criteria in protocol validation. This paper mainly discusses these two properties by analyzing the reachability tree of the PN-CAWM. Reachability tree intuitively describes all the states from the initial state $M_0$, where nodes denote $M_0$ and its successors. In a reachibility tree, the $M_0$ is the root and leaf nodes correspond to final states. A path from the $M_0$ to a leaf node means an execution sequence. Figure 8 is the reachability tree of the PN-CAWM. In the following, we prove the PN-CAWN is bounded and L1-live.

**Theorem 1.** PN-CAWM is bounded.

**Proof:** A Petri net $(N, M_0)$ is said to be k-bounded (or simply bounded) if the number of tokens in each place does not exceed a finite number $k$ for any marking reachable from the initial marking $M_0$, i.e., $m_i \leq k$ for every place $p_i$ and every marking $M \in R(M_0)$ [15]. By examination of the reachability tree of the PN-CAWM, $\omega$ (a symbol to represent an arbitrarily large value) does not occur anywhere, and the number of tokens in each place is never more than 1. Therefore, the PN-CAWM is bounded and $k$ equals to 1.

**Theorem 2.** PN-CAWM is L1-live.

**Proof:** A transition $t$ is L1-live if $t$ can be fired at least once in some firing sequences. Furthermore, a Petri net $(N, M_0)$ is said to be L1-live if each transition in the Petri net is L1-live [15]. For a bounded Petri net, the reachability tree contains all possible reachable markings. By examining the reachability tree of the bounded Petri net PN-CAWM, each marking is reachable and each transition can be fired at least once from $M_0$. Thus, PN-CAWM is L1-live.

Theorem 2 indicates that PN-CAWM can be deadlock-free as long as the firing starts from $M_0$. According to theorem 1 and theorem 2, we can conclude that the workflow management algorithm CAWM is correct.

### 6. INTELLIGENT CAMPUS NAVIGATION SYSTEM

We have developed an intelligent pervasive campus system (iCampus) with the architecture shown in Figure 9, which integrates various heterogeneous networks (wired and wireless networks, sensor network, self-organized network, Bluetooth etc.), and covers libraries, restaurants, meeting rooms, offices and sports fields. iCampus provides users intelligent personalized services and software migration and roaming in the campus.

Intelligent campus navigation is the major function of our iCampus system, which can provide navigation information for users in the graphical and textual way based on user’s current locations, network states and device capacity. Our workflow management algorithm CAWM is at the centre of the intelligent campus navigation subsystem. When a user enters the campus, he needs to input his interested targets only and the navigation subsystem will automatically connect an appropriate server and show the path information and brief introduction of targets on his PDA. For example as illustrated in Figure 10, the navigation information of a library is shown in a PDA. Moreover, we have implemented a Petri-net-based automatic verification tool in our campus navigation system.
We present a case study to explain how CAWM provides navigation information for the campus tour mentioned in Section 1. The set of interested targets is $\text{IT} = \{\text{IT}_1, \text{IT}_2, \text{IT}_3\}$, where $\text{IT}_1$, $\text{IT}_2$ and $\text{IT}_3$ mean to visit a library, have a lunch and finally take a bus, respectively. $\text{IT}$ is an ordered set such that $\text{IT}_1 < \text{IT}_2 < \text{IT}_3$. In the pervasive campus shown in Figure 11, each interested target $\text{IT}_i$ has two candidates, i.e., $\text{C}_1 = \{\text{library A, library B}\}$, $\text{C}_2 = \{\text{restaurant A, restaurant B}\}$ and $\text{C}_3 = \{\text{bus stop A, bus stop B}\}$. In addition, let there be K wireless links between Alex’s PDA and the server. CAWM automatically discovers navigation information of the three interested targets in the following steps (see Figure 11):

1. Alex requests the navigation information of $\text{IT}_1$, $\text{IT}_2$ and $\text{IT}_3$ through his PDA.
2. PDA selects a qualified wireless link that can be used to communicate with a server and then connects with the server.
3. The server returns the navigation information of both libraries A and B. Alex selects library B that has his preferred books.
4. CAWM returns the navigation information of restaurant B and bus stop B that are closer to the library B than their counterparts.
7. CONCLUSIONS AND FUTURE WORK

We have presented a context model and a context-aware workflow management algorithm CAWM for pervasive campus navigation. Advantageous to the existing workflow management technologies, our algorithm can adjust workflow execution behaviors based on current context information. We also modeled the workflow management algorithm and analyzed its correctness through Petri nets. We implemented CAWM in our intelligent pervasive campus system. Experimental testing has demonstrated that CAWM is able to guide campus users intelligently and transparently. Although we currently focus on navigation applications, the CAWM can be extended to other pervasive applications, for example, traffic information and traveling guidance, by enriching context entities and attributes.

As part of the future work, we shall investigate how to model users’ behavior, using data mining and reasoning, and further predict the users’ actions based on their profiles and history data. These results shall be integrated to our system and extend it into a context knowledge based intelligent navigation system for many application areas.

ACKNOWLEDGEMENTS

The author Dr. Feilong Tang would like to thank The University of Aizu (UoA), Japan for providing the excellent research environment for his JSPS (The Japan Society for the Promotion of Science) Postdoctoral Fellowship for Foreign Researchers in Japan. Thanks are also given to Professor Zixue Cheng in UoA, Japan for his precious help.

REFERENCES


ABOUT THE AUTHORS

F. Tang received his Ph.D degree in Computer Science and Technology from Shanghai Jiao Tong University (SJTU), China in 2005. Currently, he works with the Department of Computer Science and Engineering at SJTU, China. His research interests include grid and pervasive computing, reliability computing, wireless sensor networks and distributed computing, especially on distributed transaction processing and workflow management.
I. You received his MS and Ph.D. degrees in the Division of Information and Computer Science from the Dankook University, Seoul, Korea in 1997 and 2002, respectively. He is now an assistant professor in the School of Information Science at the Korean Bible University. His research interests include MIPv6 security, key management, authentication and access control. He is a member of the IEEK, KIPS, KSII, and IEICE.

M. Guo received his Ph.D. degree in computer science from University of Tsukuba, Japan. He is now a full professor at the Department of Computer Software, The University of Aizu, Japan. His research interests include pervasive computing, parallel and distributed processing and parallelizing compilers. He is a member of ACM, IEEE, IEEE Computer Society, and IEICE.

S. Guo received the PhD degree in computer science from the University of Ottawa, Canada in 2005. From 2006 to 2007, he was an Assistant Professor at the University of Northern British Columbia, Canada. He is currently an Assistant Professor at School of Computer Science and Engineering, the University of Aizu, Japan. His research interests are in the areas of protocol design and performance analysis for communication networks, with a special emphasis on wireless ad hoc and sensor networks.