Modeling and Analyzing Method for CPS Software Architecture Energy Consumption

Guangquan Zhang\textsuperscript{1,2*}, Kan Zhang\textsuperscript{1}, Xueyang Zhu\textsuperscript{3}, Mingcai Chen\textsuperscript{1}, Chengkai Xu\textsuperscript{1}, Yuzhen Shao\textsuperscript{1}

\textsuperscript{1} School of Computer Science and Technology, Soochow University, Suzhou, China
\textsuperscript{2} State Key Lab. for Novel Software Technology, Nanjing University, Nanjing, China
\textsuperscript{3} State Key Lab. of Computer Science, Institute of Software, Chinese Academy of Science, Beijing, China

*Corresponding author Email: gqzhang@suda.edu.cn

\textbf{Abstract}—CPS is a kind of networked embedded system. Its trustworthiness and cost are impacted by energy consumption. So design a low-power, high trustworthiness CPS has been a major challenge. Modeling and analyzing the energy consumption of CPS software architecture at design stage can help to find the energy consumption design defects. These methods can effectively improve the trustworthiness of the CPS software and reduce development costs. Against this problem, first introduce the concept of the energy consumption time Petri nets; then through analyzing the characteristics of the CPS software energy consumption, use energy consumption time Petri nets to construct the CPS software architecture energy consumption model and discuss the energy consumption problems of CPS software such as lowest energy consumption, largest energy consumption and average energy consumption; Finally, indicate the effectiveness of the proposed method by an instance.

\textbf{Index Terms}—modeling, analyzing, Software Architecture Energy Consumption, CPS

I. INTRODUCTION

With the advocacy of low-carbon energy, how to develop a low-cost, low-power, and high-performance software system has become a hot research point of embedded systems in recent years. The energy consumption is the original concept of hardware. Tiwari and other people think that analyzing the energy consumption only from the perspective of the hardware structure cannot meet the demand for system-level energy consumption analysis. So the concept of software energy consumption is proposed. Software energy consumption means the energy consumption of the instruction of system components during the execution\textsuperscript{[5]}. The purpose to research the software energy consumption is mapped hardware energy consumption to the software architecture and software characteristics. Then study the impact of software on system energy consumption, and use software design optimized technology to improve system energy consumption efficiency by this purpose. CPS is a kind of networked embedded systems \textsuperscript{[2]}, so its software energy consumption issues are critical. CPS’s attributes, such as software reliability, security, performance, and so on, are constraint by energy consumption \textsuperscript{[3]}, so how to design a low-cost, high-trustworthiness CPS software has become the urgent needs and major challenges for CPS development. Existing energy consumption analyzing and assessment methods usually achieved the running nature of the energy consumption of the system through simulation or test. These methods have some defects such as the high experimental cost and the high re-design cost. These defects of existing energy consumption analyzing and assessment methods are particularly evident when analyzing and evaluating the CPS software.

The software architecture provides the design model and guidance for the structure, behavior and key attributes of the software, which are important factors in software development to meet the user's quality demand. As a software quality attribute, low energy consumption has become the requirement which must be considered by architecture designers. Therefore, in order to improve software development efficiency, reduce costs, and avoid the software re-design caused by energy consumption problem in later development stage, expect accurately understand the runtime software energy consumption nature such as maximum, minimum and average energy consumption at the design stage through the software architecture energy consumption model, which can help to find energy consumption design defects in the design stage and re-adjust software architecture. Petri nets is an effective modeling tool for describing and analyzing the complex systems from the view of the process. It can be widely used in describing and studying the concurrent, asynchronous, and distributed characteristics of the system. And it can provide an operational semantics and the analyzing method combining with qualitative and quantitative. Against these problems, the paper first introduced the concept of the ETPN (energy consumption time Petri nets) \textsuperscript{[4]}, which extended time Petri net by energy consumption. It can effectively describe the behavior and the energy consumption of the software; then by analyzing the energy consumption characteristics of CPS software, used energy consumption time Petri nets to construct the CPS software architecture energy consumption model, and discussed the minimum energy consumption, the maximum energy consumption and the average energy consumption of the CPS software based on the energy consumption model; finally, an example illustrates the correctness and validity of the proposed...
method.

II. RELATED WORK

The energy consumption of embedded systems has caused widespread concern of related field researchers. Reference [5] confirmed the main factor of the energy consumption of embedded system is determined by the accesses number of memory in specific embedded systems through modeling the energy consumption of microprocessor-based embedded system. In Non-preemptive real-time systems, the device will take up a lot of transition time when waking up from sleep, which hindered the dynamic energy management of systems. Against this problem, reference [6] proposed a real-time task scheduling algorithm, which can maximize the effectiveness of dynamic energy management of I/O devices. Reference [7] found that increasing the embedded system memory will lead to the increase in system cost, energy consumption. To solve this problem, a software-based embedded system RAM compression algorithm was proposed. This algorithm can significantly improve the validity of memory capacity and maintain the system performance and energy consumption without changing hardware or system design.

The above studies are still stuck in the instruction-level and code-level energy consumption analysis and estimation, while research in the higher abstraction level about energy consumption is relatively few. So energy consumption analysis based on the software architecture has become a hot research topic. Reference [8] can effectively reduce the energy consumption of the system through optimizing process concurrent management process by extending the control / data flow chart, such as the merger process, the transmission of data and mass, calculation migration and / data flow chart, such as the merger process, the concurrent management process by extending the control / data flow chart, such as the merger process, the transmission of data and mass, calculation migration and

The above studies are still stuck in the instruction-level and code-level energy consumption analysis and estimation, while research in the higher abstraction level about energy consumption is relatively few. So energy consumption analysis based on the software architecture has become a hot research topic. Reference [8] can effectively reduce the energy consumption of the system through optimizing process concurrent management process by extending the control / data flow chart, such as the merger process, the transmission of data and mass, calculation migration and / data flow chart, such as the merger process, the transmission of data and mass, calculation migration and / data flow chart, such as the merger process, the transmission of data and mass, calculation migration and / data flow chart, such as the merger process, the transmission of data and mass, calculation migration and / data flow chart, such as the merger process, the transmission of data and mass, calculation migration and

III. ENERGY CONSUMPTION TIME PETRI NETS

In traditional Petri nets theory, the concept of time does not explicitly reflected in the prototype of Petri nets, which limits the application of Petri nets in real-time systems. Therefore, introducing the concept of time into Petri nets can effectively improve the ability of Petri nets for analyzing the time behavior and evaluating the performance of real-time systems. And expending the time Petri net by energy consumption, the generated ETPN model can be further analyzed the energy consumption of the system. The simply introduction about the basic concepts, definitions and terminology of ETPN model is as following.

A. Basic Definitions

Definition 1. Time Petri net is a six-tuple $TPN = (P, T, F, M_0; \alpha, \beta)$:

1. $P = \{p_1, p_2, \ldots, p_m\}$ is a finite, non-empty set of places;
2. $T = \{T_1, T_2, \ldots, T_n\}$ is a finite, non-empty set of transitions;
3. $F \subseteq (P \times T) \cup (T \times P)$ are represented as edges;
4. $M_0 : P \rightarrow N$ is initial mark, $(P, T, F, M_0)$ is a Petri net;
5. $\alpha : T \rightarrow Q^+$ is the earliest fire time function;
6. $\beta : T \rightarrow Q^+$ is the latest fire time function.

In order to describe the characteristic of the energy consumption of the system, on the basis of time Petri nets, form the energy consumption time Petri nets by given transition the energy consumption function. The definition is as follows.

Definition 2. Energy consumption time Petri nets is a seven-tuple $ETPN = (P, T, F, M_0; \alpha, \beta, e)$:

1. $(P, T, F, M_0; \alpha, \beta)$ is a $TPN$;
2. $e : T \rightarrow Q^+$ is the transition energy consumption function, it represents the energy consumption changes in per unit of time.

Next to discuss ETPN semantic model, and introduce the concept of energy consumption time transition system.

Definition 3. The semantic of ETPN is an energy consumption time system, which is a five-tuple, representing as $ETTS_{ETPN} = (S, \Sigma, s_0, \rightarrow, \omega)$:

1. $S$ is a set of all ETPN mark states;
2. $\Sigma \subseteq (T \times N)$ is a set of $(t, \sigma)$, and $\sigma$ represents the execution time of transition $t$;
3. $s_0 = m_0$ represents the initial mark state;
4. $\rightarrow \subseteq (S \times \Sigma \times S)$ represents the relationship between transition;
5. $\omega$ represents the energy consumption of fired transition. $\forall t \in T, m \xrightarrow{1 \sigma} m'$, then $\omega = \sigma \times e(t)$.

The behavior of energy time Petri net is defined as an energy transition sequence as the behavior of time Petri net. The energy transition sequence represents as $\delta = (t_1, \sigma_1, \omega_1) \ldots (t_i, \sigma_i, \omega_i) \ldots (t_n, \sigma_n, \omega_n)$. $(t_i, \sigma, \omega)$ represents the execution time of transition $t_i$ is $\sigma$, units time. The energy consumption is $\theta_i = \sigma \times e(t_i)$. $e(t_i)$
is the energy consumption in unit time of transition \( t_i \).

So \( \sum_{j=1}^{m} \sigma_j \) is the total time after the execution of \( \delta \).

\[ AE = \sum_{j=1}^{m} \sigma_j \]

is the total energy consumption after the execution of \( \delta \). If there is an energy transition sequence \( \delta \), which can deduce \( m_i \) from \( m_0 \), then \( m_i \) is reachable.

**Definition 4.** Let two ETPN models represent as \( A \) and \( B \). Suppose \( P_A \) and \( P_B \) represent the place set of two ETPN models, \( P_A^I \) and \( P_B^I \) represent the input place sets of two ETPN models, \( T_A^I \) and \( T_B^I \) represent the input transition sets of two ETPN models, \( P_A^O \) and \( P_B^O \) represent the output place sets of two ETPN models, \( T_A^O \) and \( T_B^O \) represent the output transition sets of two ETPN models, \( P_A^H \) and \( P_B^H \) represent the inner place sets of two ETPN models. When two ETPN models meet following conditions:

1. \( P_A^H \cap P_B = \emptyset \), \( P_A^I \cap P_B^I = \emptyset \);
2. \( P_A^O \cap P_B^O = \emptyset \), \( P_A \cap P_B^H = \emptyset \);

Then \( A \) and \( B \) is composable, the communication set is

\[ Communication(A,B) = (T_A^I \cap T_B^I) \cup (T_A^O \cap T_B^O) \]

The above definition is based on the component thought. Two different components in the system perform different functions. They should have different input and output. The same input should get the same output. Meanwhile, the output of the one component is often the input of another component, or vice versa.

**Definition 5.** If ETPN model \( A \) and \( B \) is composable, then the composition model \( A \otimes B \) is also an ETPN model:

1. \( P_{A \otimes B} = P_A \cup P_B \);
2. \( T_{A \otimes B} = T_A^I \cup T_B^I \cup Communication(A,B) \);
3. \( F_{A \otimes B} = (P_{A \otimes B} \times T_{A \otimes B}) \cup (T_{A \otimes B} \times P_{A \otimes B}) \);
4. \( M_{A \otimes B} : P_{A \otimes B} \rightarrow N \);
5. \( \alpha_{A \otimes B} : T_{A \otimes B} \rightarrow Q^+ \);
6. \( \beta_{A \otimes B} : T_{A \otimes B} \rightarrow Q^+ \);
7. \( e_{A \otimes B} = T_{A \otimes B} \rightarrow Q^+ \).

B. Reachable State Graph

Reachable state graph is one of important methods to analyze the feature of Petri nets. Firstly introduce the definition of the interval of energy consumption, and then discuss the definition of ETPN state class and the definition of reachable state graph with the extension of the energy consumption.

**Definition 6.** Let \( E = [a,b] \). If \( a,b \in Q^+ \), then \( E \) is an interval of energy consumption. Let \( EI \) represent a set of all energy consumption interval. Suppose \( E_i, E_j \in EI \), \( E_i = [a,b] \), \( x \in Q^+ \), let

\[ \downarrow E_i = a \quad \uparrow E_i = b \quad E_i + E_j = [\downarrow E_i + \downarrow E_j, \uparrow E_i + \uparrow E_j] \]

Thus, \( E_i + x = [\downarrow E_i + x, \uparrow E_i + x] \) be the primary and secondary computing of an energy consumption interval. Then \( E_i + E_j + x, E_i + x \) is also the energy consumption interval.

Then define the state classes of ETPN. The detail of definition is as follows.

**Definition 7.** For an ETPN, a state class \( C = (M, \theta, E) \) is a three-tuple:

1. \( M \) is marks set;
2. \( \theta \) is the total time interval of fired transition;
3. \( E \) is the total energy consumption interval of fired transition;

Reachable state graph is the digraph combined with all state classes in transition fired order. The specific definition is as follows.

**Definition 8.** Reachable state graph \( G = (CS, TS) \) is a digraph. It is obtained from \( ETTSTP \):

1. \( CS \) is the set of all nodes, each node represents a state class;
2. \( TS : CS \rightarrow CS \) is the set of all edges, each edge represents the relationship between two states.

After extending the energy consumption for the state class, the status of the reasonableness and completeness should be discussed. Reasonable means any state class calculated by definition 3 always with an energy interval; Completeness means the total energy consumption from the initial state class to any state class certainly falls within the interval of the energy consumption.

**Theorem 1.** For a state class \( (M_n, \theta_n, E_n) \) reached from initial state class \( (M_0, \theta_0, E_0) \), \( E_n \) must be an energy consumption interval; and the total energy consumption of any transition fire sequence from \( (M_0, \theta_0, E_0) \) to \( (M_n, \theta_n, E_n) \) certainly falls within \( E_n \).

**Proof.** Firstly, use mathematical induction to prove \( E \) must be an energy interval. By the known of condition, there is a transition fire sequence \( t_1 t_2 \cdots t_{n-1} \) from \( (M_0, \theta_0, E_0) \) to \( (M_n, \theta_n, E_n) \). If \( n = 0 \), \( (M_0, \theta_0, E_0) \) is \( (M_0, \theta_0, E_0) \). Then holds. Suppose \( n \leq k - 1 \) theorem holds, \( E_k \) is an energy consumption interval. If \( n = k \), \( E_{k+1} = E_k + [\alpha(t_k) \times e(t_k), \beta(t_k) \times e(t_k)] \). By the definition 4, \( [\alpha(t_k) \times e(t_k), \beta(t_k) \times e(t_k)] \) is an energy consumption interval. And by the known of definition 4 and \( E_k \), \( E_{k+1} \) is also an energy consumption interval.

Similarly, use mathematical induction to prove the total energy consumption of any transition fire sequence from \( (M_0, \theta_0, E_0) \) to \( (M_n, \theta_n, E_n) \) certainly fall within \( E_n \). If \( n = 0 \), then total energy consumption \( AE_0 = 0 \). Theorem holds. Suppose \( n \leq k - 1 \) theorem holds, then
the total energy consumption $AE_{k+1}$ of transition fire sequence $t_1,t_2,...,t_k$ certainly falls within $E_k$. If $n=k$, then the fire energy consumption $C_i$ of transition $t_i$ certainly falls within $[\alpha(t_i) \times e(t_i), \beta(t_i) \times e(t_i)]$. So $AE_{k+1} + C_i$ certainly falls within $E_k + [\alpha(t_i) \times e(t_i), \beta(t_i) \times e(t_i)]$. The total energy consumption $AE_i$ of transition fire sequence $t_1,t_2,...,t_k$ certainly falls within $E_k$, while $E_{k+1} = E_k + [\alpha(t_i) \times e(t_i), \beta(t_i) \times e(t_i)]$.

IV. MODELING AND ANALYSIS OF CPS SOFTWARE ARCHITECTURE ENERGY CONSUMPTION

CPS is the system closely integrated computing system with the physical environment. So the energy consumption of computing and physical processes must be taken into account by constructing and analyzing the energy consumption model. Firstly, analyze the features of CPS software energy consumption.

A. Features of CPS Software Energy Consumption

There are three modules in any CPS: computing module, communication module and the physical module. The computing module is responsible for the implementation of the computing work of CPS, including data processing, control decisions; the communication module is responsible for information exchange between the various modules; the physical module is responsible for the perception and control of the physical environment. Figure 1 is an abstract structure of CPS energy consumption, which can define the energy consumption of the CPS software $\rho^{pCPS}$.

Definition 9. The CPS software energy consumption depends on the computing and non-computing energy consumption. The energy consumed in the computing process is called computing energy consumption, including computing module and communication module energy consumption. Energy consumption in the non-computing process is called non-computing energy consumption, mainly refers to the physical module's energy consumption. The $\rho^{pCPS}$ can be represented as follows:

$$\rho^{pCPS} = \text{computing}_\text{energy} + \text{non-computing}_\text{energy}.$$ 

B. Modeling CPS Software Architecture Energy Consumption

After analyzing the energy consumption features of the CPS software, use the energy consumption time Petri nets to construct CPS software architecture energy consumption model according to definition 9. CPS contains computing module, communication module and the physical module, so the CPS software ETPN model is defined as follows.

Definition 10. Energy consumption time Petri of CPS is $\text{ETPN}^{CPS} = (\text{C}_{\text{ETPN}}, \text{CM}_{\text{ETPN}}, \text{NC}_{\text{ETPN}})$, $\text{C}_{\text{ETPN}}$, $\text{CM}_{\text{ETPN}}$ and $\text{NC}_{\text{ETPN}}$ are the ETPN models of computing module, communication module and the physical module.

According to definition 8, construct the CPS software ETPN model. The construction process is as follows.

1. First, analyze the parts of the system. CPS is composed by independent embedded devices. Each device has a computing, communications or control module. Each module is responsible for a certain function, and has a certain degree of autonomy, but it depends on the computing results of the other modules of the system. The structure model of CPS is shown in Figure 2. There are numerous devices in CPS. Each device is composed by the computing module, communication module, or (and) the control module. Effective and reliable communication between devices through the communication module depends on the network.

2. Abstract each module into an ETPN model. Add the input or output places or transition for each module needed communication as the communication interface with other modules. By definition 4, determine the composable ETPN model. In practical systems, sending and receiving information will consume energy. Therefore, add energy consumption function to the input/output transition.

3. After established ETPN model of each module, connect each module by the communication interface according to the communicating order. As
the information transmission process also consuming energy (such as the energy consumption of the network device), add energy consumption function to transition connecting two ETPN model, which represents the transmission energy consumption.

C. Analyzing CPS Software Architecture Energy Consumption

After constructing the CPS software architecture energy consumption model, reachable state graph of the model can be constructed according to definition 6. Then CPS software architecture energy consumption problems can be analyzed by reachable state graph. The CPS software architecture energy consumption problems include the software maximum energy consumption, minimum energy consumption and the average energy consumption analysis. Here can be converted to searching the energy consumption path from initial state to the target state in reachable state graph. The analysis of these questions is as follows. Because of the real-time of CPS software, the desired behavior sequence within a certain time period is generally limited. So we will not consider the energy consumption problems in the case of infinite behavior sequence.

- Minimum Energy Consumption Path Analysis

**Theorem 2.** In reachable state graph, the minimum energy path reaching any node must be a simple path.

**Proof.** Use proof by contradiction to prove. Let the minimum energy path reaching a node in reachable state graph is a simple path, which is represented as \( \epsilon = C_0 \rightarrow C_1 \rightarrow C_2 \rightarrow \ldots \rightarrow C_j \). The energy consumption of the path is \( \epsilon \). There are at least a sub-path contains the ring in this path. Let the ring as \( \epsilon_1 = C_j \rightarrow C_{j+1} \rightarrow \ldots \rightarrow C_1 \). And then let the rest of the sub-path as \( \epsilon_2 = C_0 \rightarrow C_1 \rightarrow \ldots \rightarrow C_j \). Then the rest of the sub-path \( \epsilon_2 \) is certainly limited after given the time deadline to reach \( C_m \). If not, the time will be unlimited. So the energy consumption and the number of path is limited. So after determining all paths and energy consumption to reach \( C_m \) in a given time period, it can certainly find the energy consumption of a path is greater than or equal to the energy consumption of the other paths. That means the minimum energy consumption path to reach \( C_m \) exists.

Then the algorithm to search reachable state graph for the maximum energy consumption path reaching a node can be constructed by theorem 2. Since there may be different paths with the same energy consumption, the minimum energy path may be more than one. Because this article only focus on the energy consumption of path, and are not concerned about the path itself. So let one of paths as search result. Since each state has an energy interval, let the minimum value of the energy consumption interval as the minimum energy consumption reaching the state.

**Algorithm 1.** Minimum Energy Consumption Path

**Input:** reachable state graph \( G \), target node \( C_m \);

**Output:** the minimum energy consumption path \( \min_{\text{path}}, \) the minimum energy consumption \( \min_{\text{AE}}; \)
\( \min_{\text{AE}} = \infty, \ AE = 0 \); \( \text{// AE is the energy consumption of current path} \)
\( \min_{\text{path}} = \emptyset; \)
\( \text{current}_{\text{path}} = \langle C_0 \rangle; \)

\begin{verbatim}
do
node = the last node of current_path;
if there is no new node, then delete the last node of current_path;
else
node = new node of current node;
if node == \( C_m \) then
compute the AE of current_path;
if AE < \( \min_{\text{AE}} \) then
\( \min_{\text{AE}} = \text{AE}; \)
\( \min_{\text{path}} = \emptyset; \)
\( \min_{\text{path}} = \text{current_path} \cup \ C_m; \)
else current_path = current_path \cup node;
while current_path = < >
return \( \min_{\text{path}}; \)
\end{verbatim}

- Maximum Energy Consumption Path Analysis

**Theorem 3.** If a period of time to reach any node was given, then there is the maximum energy consumption path of all path to reach the node.

**Proof.** Without loss of generality, let the reach node be \( C_m \). Because each transition consumes time, the length of any path to reach \( C_m \) is certainly limited after given the time deadline to reach \( C_m \). If not, the time will be unlimited. So the energy consumption and the number of path is limited. So after determining all paths and energy consumption to reach \( C_m \) in a given time period, it can certainly find the energy consumption of a path is greater than or equal to the energy consumption of the other paths. That means the maximum energy consumption path to reach \( C_m \) exists.

Then the algorithm to search reachable state graph for the maximum energy consumption path reaching a node can be constructed by theorem 2. Since there may be different paths with the same energy consumption, let one of paths as search result. Since each state has an energy interval, let the maximum value of the energy consumption interval as the maximum energy consumption reaching the state.

**Algorithm 2.** Maximum Energy Consumption Path

**Input:** reachable state graph \( G \), target node \( C_m \), time deadline \( T_D \);

**Output:** the maximum energy consumption path \( \max_{\text{path}}, \) the maximum energy consumption \( \max_{\text{AE}}; \)
\( \max_{\text{AE}} = \infty, \ AE = 0 \); \( \text{// AE is the energy consumption of current path} \)
\( T = 0; \) \( \text{//T is the time consumption of current path} \)
\( \max_{\text{path}} = \emptyset; \)
current_path = < C₀ >;
do
  node = the last node of current_path;
  if there is no new node, then delete the last node of current_path;
  else
    node = new node of current_path;
    if T = T + t(node) ≤ T₉ then
      if node = = Cₙ then
        compute AE of current_path;
        if AE > max_AE then
          max_AE = AE;
          max_path = φ;
          max_path = current_path ∪ Cₙ;
        else current_path = current_path ∪ node;
      return max_path = < >
    return AveE;

Algorithm 3. Compute Average Energy Consumption

Input: cachable state graph G, system path execution probability array F[n];
Output: Average Energy Consumption AveE;
AveE = 0, AveE = 0; // AE is the energy consumption of current path
path_num = 0; //path_num is the number of path
current_path = < C₀ >;
do
  node = the last node of current_path;
  if there is no new node, then
    compute AE of current_path;
    AveE = AveE + AE * F[path_num];
    path_num++;
  else current_path = current_path ∪ node;
  if T = T + t(node) ≤ T₉ then
    if node = = Cₙ then
      compute AE of current_path;
      if AE > max_AE then
        max_AE = AE;
        max_path = φ;
        max_path = current_path ∪ Cₙ;
      else current_path = current_path ∪ node;
    return max_path = < >
  return AveE;

V. CASE STUDY

TCAS (Traffic Alert and Collision Avoidance System) is an information system not depending on any land-based air traffic control. It is the typical applications of CPS in the field of aviation. It can help the pilot quickly identify the relative position between the aircraft.

The TCAS generally consists of three modules: distance detection module, collision management module and flight control module. Distance detection module is mainly responsible for detecting the distance with other aircraft. And send distance information to the collision management module. Collision management module judges whether the distance safety after received the distance information. And send warning or safety information to the flight control module. The flight control module judges after the receipt of the information, and send avoidance or recovery action to the collision management module. At last, the collision management module is responsible for the implementation of avoidance or recovery action.

Model the TCAS by the above proposed modeling approach. The concrete steps are as follows:
(1) Analyze TCAS, including the distance detection module, collision management module and flight control module;
(2) Build ETPN model for each module, and identify the communication interface of each module;
(3) According to the order of communication between modules, connect the respective interface.

According to the above described modeling steps, obtain the ETPN model of TCAS shown in Figure 3. The name, execution time and the energy consumption of each transition is shown in Table 1.
TABLE 1.

THE INFORMATION OF EACH TRANSITION

<table>
<thead>
<tr>
<th>Transition</th>
<th>Name</th>
<th>Execution time</th>
<th>Energy consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>Detecte distance</td>
<td>[3,8]</td>
<td>20</td>
</tr>
<tr>
<td>T2</td>
<td>send dangerous distance</td>
<td>[1,3]</td>
<td>8</td>
</tr>
<tr>
<td>T3</td>
<td>send safety distance</td>
<td>[1,2]</td>
<td>7</td>
</tr>
<tr>
<td>T4</td>
<td>receive dangerous distance</td>
<td>[2,3]</td>
<td>9</td>
</tr>
<tr>
<td>T5</td>
<td>receive safety distance</td>
<td>[1,2]</td>
<td>8</td>
</tr>
<tr>
<td>T6</td>
<td>send warming information</td>
<td>[2,4]</td>
<td>8</td>
</tr>
<tr>
<td>T7</td>
<td>send safety information</td>
<td>[1,3]</td>
<td>7</td>
</tr>
<tr>
<td>T8</td>
<td>avoid</td>
<td>[3,7]</td>
<td>21</td>
</tr>
<tr>
<td>T9</td>
<td>recover</td>
<td>[3,6]</td>
<td>17</td>
</tr>
<tr>
<td>T10</td>
<td>receive warming information</td>
<td>[2,5]</td>
<td>8</td>
</tr>
<tr>
<td>T11</td>
<td>receive safety information</td>
<td>[2,4]</td>
<td>7</td>
</tr>
<tr>
<td>T12</td>
<td>Send avoidance</td>
<td>[1,4]</td>
<td>9</td>
</tr>
<tr>
<td>T13</td>
<td>Send recovery</td>
<td>[1,3]</td>
<td>7</td>
</tr>
<tr>
<td>T14, T15, T16</td>
<td>communication</td>
<td>[1,2]</td>
<td>10</td>
</tr>
<tr>
<td>T17, T18, T19</td>
<td>communication</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

According to ETPN model of Figure 3, construct the reachable state graph of TCAS, which is shown in Figure 4. Due to space constraints, and the mark set is not the focus of this article. Therefore, the specific content of each mark set in the reachable state graph is not given.

After constructed Figure 4, the relevant energy consumption problems of TCAS can be analyzed by using the above proposed analytical method. After applying the algorithm 1 and algorithm to Figure 4, the minimum energy consumption path is <M₀, M₁, M₁₀, M₁₁, M₁₂, M₁₃, M₁₄, M₁₅, M₁₆, M₁₇, M₀>). The minimum energy consumption is 184. The minimum energy consumption path is <M₀, M₁, M₂, M₃, M₄, M₅, M₆, M₇, M₈, M₉, M₀>). The maximum energy consumption is 526. Assumed that the execution probability of T2 and T3 were 0.3 and 0.7, and then the average energy consumption of the system is 323.6. In addition to the minimum energy consumption, largest energy consumption and average energy consumption, sometimes need analyze the energy consumption of specific certain period process according to requirements. Then the energy consumption of the process can be analyzed as long as finding the corresponding part of the reachable state graph of the process.

VI. CONCLUSIONS

In the aspect of software trustworthiness analysis and evaluation research, some related methods have been proposed [12-15]. As a networked embedded system, the trustworthiness, cost and other attributes of CPS are constrained by energy consumption. How to find design defects of the energy consumption of CPS software in the design phase, effectively reduce development costs, and improve system trustworthiness, has become requirements and challenges for CPS development. To solve this problem, we firstly introduced the concept of energy consumption time Petri nets. Then used energy consumption time Petri nets to construct the CPS software architecture energy consumption model through analyzing the energy consumption features of the CPS software, and proposed analysis methods for CPS software minimum energy consumption, maximum energy consumption and the average energy consumption.

The next work will consider how to use the analysis results such as minimum energy consumption, maximum energy consumption and average energy consumption to evaluate the trustworthiness of the CPS software energy consumption. It can help understand how energy consumption impact on the trustworthiness of CPS software.

ACKNOWLEDGMENT

This research was sponsored by the Natural Science Foundation of Jiangsu Province (BK2011281), Applied Basic Research Program of Suzhou (SYG201241), Scientific Research Innovation of College Graduated in Jiangsu Province under Grant (CXLX12-0809, CXLX13-820) and College Students' Practice and Innovation Training Project of Soochow University (2012yb010).

REFERENCES


Guangquan Zhang received the MS and PhD degrees in computer science from Chongqing University, in 1988 and 1999, respectively. He is currently a professor in the School of Computer Science and Technology, Soochow University, China, and is the senior member of CCF. His research interests include networking software engineering, formal methods, cloud computing and Cyber Physical Systems, et.al.

Kan Zhang was born in Jiangsu, China. He received his Bachelor degree in Software Engineering in 2010 from School of Computer science and Technology, Soochow University, China. Now he is a graduate student at Soochow University, China. His main research interests include Embedded system, CPS and software trustworthiness evaluation.