Energy-efficient Workflow Distribution*

Daniel Fischer, Stefan Föll, Klaus Herrmann, Kurt Rothermel
Institute of Parallel and Distributed Systems, Universität Stuttgart
Universitätsstrasse 38, 70569 Stuttgart, Germany
{daniel.fischer, stefan.foell,klaus.herrmann, kurt.rothermel}@ipvs.uni-stuttgart.de

ABSTRACT

Pervasive computing and business process modeling are increasingly joining forces, as mobile human users shall be seamlessly integrated into business processes. In respective scenarios, humans use mobile devices and wireless technology to interact with workflows running in a powerful back-end infrastructure. However, the frequent interaction between humans and workflows causes a high communication overhead and, thus, high energy consumption on mobile devices. This impacts the usability and efficiency of the business process due to rapidly drained batteries and the resulting short life-times of the devices and applications. We present an approach based on a minimum-cut algorithm for reducing costly data transmissions during workflow execution by distributing parts of a workflow to the users' devices. Our motivation is to reduce the energy consumption on the mobile devices and, thus, avoid draining batteries in the field. We prove that our algorithm finds the optimal solution for a given network and workflow, decreasing the energy consumed on mobile devices by 32-37% compared to an approach where the entire workflow is executed in the infrastructure. Thus, in typical domains like logistics and health care, one third of the energy can be saved. This either means that devices have to be charged less frequently, leading to less distraction in the business process, or that mobile device specifications can be lowered. Significant cost reductions result in both cases.

Categories and Subject Descriptors

D.2.4 [**Computer-Communication Networks**]: Distributed Systems; H.4 [**Information Systems Applications**]: Office Automation—*Workflow Management*

General Terms

Algorithms, performance

*This research has been supported by FP7 EU-FET project ALLOW (contract number 213339).

COMSWARE 2011, July 04-07, Verona, Italy

DOI 10.4108/comsware.2011.13

Keywords

Workflow distribution, energy efficiency, minimum cut.

1. INTRODUCTION

By using workflows, organizations are able to automate and optimize their business processes [1]. Workflows are a way of orchestrating services in order to implement the *programming-in-the-large* paradigm. They define a set of activities that call services, and they define constraints on the execution order of these activities. While services are often provided by software components, *human-centric* application scenarios such as pervasive healthcare and logistics require extensive human interaction to execute a workflow.

Driven by the advances in the area of pervasive computing [2], humans are no longer tied to their desk. They can use mobile devices to interact with workflows virtually anywhere. While this enables unobtrusive interactions with the workflow system, mobile devices have a constrained battery lifetime such that the energy consumption is a critical point for their usability. High energy consumption causes batteries to be drained fast, disrupting the business process and leaving users unable to continue their work. This renders the process less efficient and, thus, incurs higher costs. Alternatively, devices could be fitted with larger batteries to avoid such disruptions. However, this also incurs higher costs. Therefore, preserving energy is an important goal with respect to seamless workflow execution and cost reduction.

Today's workflow systems are usually deployed in a back-end infrastructure, such that the workflows need to communicate with mobile humans over a wireless medium. Due to this deployment scenario, extensive data transmission between user devices and workflows running in the infrastructure is required for each interaction with a mobile user. Consequently, this results in high energy costs since sending and receiving data are highly energy-intensive operations with current wireless communication technologies such as GPRS, UMTS or WiFi [3].

In order to reduce these energy costs, we propose an algorithm for distributing fragments of workflows from the infrastructure to mobile user devices. Distributing workflows to mobile devices avoids transmissions of large volumes of data since this data is processed locally. Our algorithm constructs a *cost graph* that is based on the workflow model and takes two sources of energy into account: the data communication costs and the costs of service execution on the mobile device. We partition the cost graph using a minimum cut algorithm such that the workflow execution causes minimum energy consumption. Each partition of the cost graph

contains the workflow activities to be executed either in the infrastructure or on the mobile devices. Our evaluations show that this approach achieves average energy savings of 37% for GPRS and 32% for UMTS communication compared to the centralized infrastructure-based approach. Existing work in the area of workflow distribution [4, 5] only focuses on infrastructure nodes, but does not consider the impact of workflow execution on the energy budget of mobile devices.

The rest of the paper is structured as follows. In Section 2, we present an application scenario for workflow distribution. Section 3 discusses the related work. We then present our system model and give a formal description of the optimization problem underlying our approach in Section 4. In Section 5, we present the algorithm for solving the optimization problem, and we present a proof for the optimality of this algorithm. The evaluation results are discussed in Section 6. Finally, we conclude the paper in Section 7 and give an outlook on future work.

2. APPLICATION SCENARIO

The application scenario depicted in Figure 1 models a quality assurance workflow for car shipment management (logistics domain). In this scenario, we consider a big harbor where ships deliver large numbers of cars. The cars go through a series of treatments that involve e.g. fitting radios, maintenance, and cleaning activities. Cars may be parked for certain periods of time and retrieved for additional treatments or onward shipment. During this whole process, cars may be damaged. Therefore, the personnel that is handling the cars (handling drivers) needs to check them regularly for damages. Based on the results of these checks, cars may be sent back to the manufacturer, they may be sent to a repair station, or insurance cases may have to be filed. Since the handling drivers are no experts in detecting and evaluating damages, this is done by taking photos of the car and sending these photos to some image recognition service that is trained to detect specific kinds of damages.

First, an available worker is chosen by the workflow from a database using the *Worker Service*. The chosen worker gets instructions (*Notify Worker*) and interacts with the workflow by taking pictures of a car and submitting them to a diagnosis service. If the service detects a problem with the car, the worker is instructed to take further steps to fix the problem. For example, this can be an instruction manual or video.

In our example, the mobile device has to transmit 4 MB of data and receives 1 MB. Assuming that the workflow is executed in the infrastructure, this drains about 140J via UMTS and even 189J (150J+39J) via GPRS (shown in the figure) [3] of energy from the mobile device. We can save a significant amount of energy by executing the workflow (including the diagnosis service) on the users' mobile device, since data transmission dominates service execution in terms of energy usage. For the implementation of the diagnosis service on a mobile device, several approaches such as Neuronal Networks or Markov models are feasible. These models can be trained online and enable real-time recognition on mobile devices [6]. Executing the shaded area of the workflow on the mobile device costs only 22J (20J for executing the diagnosis service on the mobile device and 2J for small data transmissions) as opposed to the 189J consumed with an infrastructure-based service. We save 167J. Saving this amount of energy for every car the worker handles has a strong impact on the battery lifetime of his device.

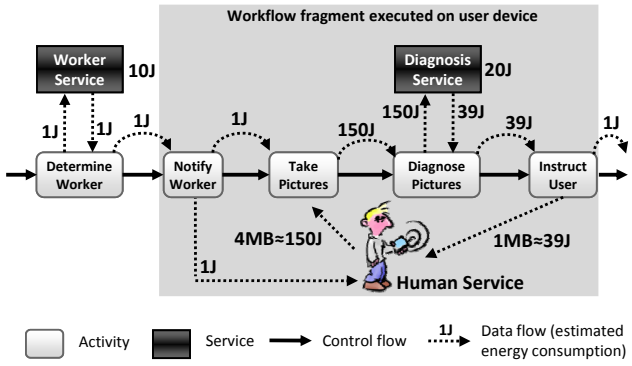

Figure 1: Example workflow with annotated energy costs

Our assumption is that off-the-shelf smartphone technology will be used for applications like the one investigated here. The performance and the form factor of these devices is adequate. Using off-the-shelf phones has several advantages:

- They are generally cheaper than special-purpose devices that have been designed for a very small market segment.
- They are constantly improved, removing the necessity to explicitly invest in costly redesigns.
- They offer a very flexible hardware platform with different communication devices and sensors.
- They offer additional communication functionality and applications that can support the general work process and enable integration.
- They are available in arbitrary numbers such that scaling the system up is not a problem in terms of the hardware.

Of course, precautions in terms of security have to be taken. But these can be implemented in software.

Typical smartphones have a battery capacity of about 17000J and the typical energy consumption of smartphone users is between 144J and 3600J per hour with a median of about 850J [7]. Hence, centralized workflow execution would drain the overall energy of a mobile device heavily in our scenario.

3. RELATED WORK

The task of workflow distribution has been studied in previous research within the workflow community. Each approach is heavily influenced by the class of performance gain to be achieved. Bauer and Dadam [4] propose an algorithm to reduce the network load produced by the workflow system. For this purpose, they have defined a probabilistic model that reflects the communication costs for workflow execution. By means of variable server assignments they create relations between activities which are used to select a suitable server in the infrastructure and distribute the workflow. Son et al. [5] propose an algorithm to distribute a workflow among infrastructure nodes to minimize communication cost in the network. Their approach is based on multi-level graph partitioning that considers different types of wired networks (e.g. LAN, WAN). However, their approach assumes equal costs for communication between hosts in the same wired network. In contrast, we assume the amount of data to be exchanged over the wireless channel to have a big impact on the energy drained on mobile devices. Hiesinger

et al. [8] present an approach to minimize the interaction time experienced by humans during workflow execution. For this purpose, they propose a list-scheduling algorithm to assign workflow activities to different network domains such that the time required for the workflow execution is reduced. However, none of the existing work in the area of workflow distribution focuses on distributing a workflow between infrastructure and mobile nodes in order to save energy.

Workflow distribution requires mechanisms to enforce a decentralized workflow execution model. Baresi et al. [9] use graph partitioning rules to transform a centralized workflow model into a set of distributed fragments that run on different devices. The partitioning rules operate on an abstract graph representation of BPEL [10] to create a set of cooperating workflows. However, while this enables the creation of valid workflow fragments, the selection of these fragment has to be done manually by a workflow designer. Our approach supports a workflow designer to find the best fragments to be executed on mobile devices in terms of energy consumption. The results of our distribution algorithm can be used to apply the rules of Baresi et al. for cutting the workflow into fragments. Therefore, both approaches complement each other for creating a decentralized energy-efficient workflow executuion environment.

MAUI [11] is a system for offloading code from a mobile device to an infrastructure in order to save energy. This is sensible if the energy required for executing the code exceeds the energy needed to transfer its state. However, MAUI works on a fine-grained level of individual functions. A workflow is a coarse-grained orchestration of large pieces of code (services). Thus, MAUI may be used orthogonally to our solution on the level of individual services. AIDE [12] is a distributed platform to offload application code from mobile devices to nearby powerful computers. For this purpose, the execution history of the application is monitored to create a fully connected graph of the application's interaction behaviour. The graph is then partitioned using a modified version of a minimum-cut algorithm, which considers the resource limitations on the mobile device such as the memory available for executing the application. For making distribution decisions it is assumed that applications can be freely migrated among different devices. In contrast, workflows often depend on services that are bound to a specific device, while other services are generally available on all devices. We explicitly deal with these restrictions in our approach for the distribution of a workflow. Furthermore, we have devised an approach for workflow distribution among an arbitrary number of devices, while AIDE can only divide an application into exactly two partitions.

We conclude that workflow distribution in a heterogeneous environment for mobile and infrastructure-based devices is of major importance to conserve scarce resources on mobile devices. Existing approaches deal with workflow distribution and energy conservation under different assumptions and, thus, are not applicable to our problem.

4. SYSTEM MODEL AND PROBLEM DESCRIPTION

In this section, we describe our system model in a formal way. Our goal is to distribute a workflow among the infrastructure and the mobile devices. Hence, we first propose a suitable network model. Then, we describe our workflow model. We conclude this section by describing the optimization problem we solve.

4.1 Network Model

The network consists of a set of mobile devices $D = \{d_1, d_2, \ldots\}$, where each device corresponds to a human user who participates in the execution of the workflow. Furthermore, the network consist of an abstraction of the back-end infrastructure inf. We abstract from the concrete implementation of this infrastructure since it does not influence the energy consumption of mobile devices. For example, the infrastructure may be a standalone server, a server cluster or even a distributed network of servers. The set of all hosts is denoted as $H = D \cup \{inf\}$. Thus, the term *host* may refer to both, infrastructure or mobile device. We assume a communication system between hosts, e.g. cellular communication (UMTS, GPRS) or WiFi via access points, such that each pair of hosts can communicate.

4.2 Workflow model

A workflow is a directed acyclic graph $W = (A, s, F, \rho, \theta_A, \theta_D)$. A denotes the set of activities in the workflow. The functionality of an activity is defined by means of the function $\lambda : A \rightarrow S$. This function binds an activity $a \in A$ to a required service $\lambda(a) \in S$, where S denotes the set of all services used by the workflow. S consists of 3 disjoint subsets S_{inf}, S_{human} and S_{mov} that represent different service classes. Each service $s \in S_{inf}$ is a computational task executed by piece of software, which is only available in the infrastructure, e.g., due to a large database which needs to be accessed. These kind of services are called *infrastructure* services. In contrast to this, each service $s \in S_{human}$ represents a task that is to be performed by a human user (e.g., repairing a car). We refer to these services as *human service*. A human service corresponds to a mobile device which is used by the human to retrieve and send the information relevant for his task. Each services $s \in S_{mov}$ can be either executed in the infrastructure or on the mobile device. An example of such a *movable service* is the diagnosis service from our application scenario in Section 2. The decision where to execute a movable service (on the mobile device or on the infrastructure) depends on the outcome of our workflow distribution algorithm.

The control flow of W is specified by means of the relation $F \subset A \times A$. The control flow defines the logical order of activities for the execution of the workflow. We refer to activities that model conditional or parallel behavior as *structural activities.* A conditional and parallel split is modeled as an activity with more than one outgoing control flow link. The set of outgoing control flow links of an activity $a \in A$ is denoted as F_a. For a given control flow link $f = (a_i, a_j)$, ρ_f is the probability that a_j will be executed after the completion of a_i. This value can be derived from execution traces of the respective workflow. For a conditional split, the workflow is executed following only a single alternative, i.e. the conditions $|F_a| > 1$ and $\sum_{f \in F_a} \rho(f) = 1.0$ hold. For a parallel split, all outgoing branches are executed in parallel, i.e. the conditions $|F_a| > 1$ and $\forall f \in F_a : \rho_f = 1$ hold. The latter also holds for all other links originating from a non-structural activity.

The data flow of W is described by the two relations $L_{\text{AA}} \subset A \times A$ and $L_{\text{AS}} \subset A \times S$. Each $(a_i, a_j) \in L_{\text{AA}}$ denotes a data link between two activities $a_i, a_j \in A$ and each $(a, s) \in L_{\text{AS}}$ denotes a data link between an activity a and its associated service $s = \lambda(a)$. Over each of the data links a certain amount of data is communicated. The amount of the data that needs to be transferred for a service call $(a, s) \in L_{\text{AS}}$ is denoted as $\theta_S(a, s)$. We assume that $\theta_S(a, s)$ covers the input as well as the output of the respective

service call. Similarly, $\theta_A(a_i, a_j)$ specifies the amount of data that has to be exchanged between two activities $(a_i, a_j) \in L_{\text{AA}}$. We assume that both θ_A and θ_S are available to our workflow distribution algorithm and can be either obtained by a workflow designer being an expert in the application domain or by analysis from histories of past execution traces of workflow instances.

4.3 Problem Description

Our goal is to find a distribution of a workflow that minimizes the energy consumption for the workflow execution on the mobile devices. More formally, a possible workflow distribution can be described by a function $\mu_1 : A \rightarrow H$ that maps each activity in the workflow to a host that shall execute it. At the same time, we have to decide for the host where to execute a movable service. Formally, this is expressed by a second mapping function $\mu_2 : S_{mov} \rightarrow H$. Among all possible mappings, we are interested in the most energy-efficient mappings μ_1^* and μ_2^*, i.e., the mappings that minimize the sum of the drained energy of all mobile devices $d \in D$.

The total energy cost for workflow execution under the given mappings μ_1 and μ_2 is denoted as $E_{total}(\mu_1, \mu_2)$. The cost is the sum of the energy consumed for executing movable services on the mobile devices as well as the energy spent for wireless communication. In the following, we refer to the energy required to transmit k bytes as $E_T(k)$. In our evaluation, we will use existing energy models to compute $E_T(k)$ for specific wireless communication technologies, e.g., GPRS. The energy required for executing a movable service s is given by $E_X(s)$. In the following, we describe the cost model to determine the total energy cost for a workflow execution.

First, let us consider the energy cost $E'_X(s)$ for executing a movable service $s \in S_{mov}$. $E'_X(s)$ is defined as:

$$E'_X(s) = \begin{cases} E_X(s), & \text{if } \mu_2(s) \in D \\ 0, & \text{otherwise} \end{cases}$$

Each movable service $s \in S_{mov}$ may be executed on a mobile device or in the infrastructure. If s resides on a mobile device, energy has to be spent for its execution. Otherwise, if s is executed in the infrastructure, no energy is consumed on the mobile devices.

Second, we have to consider the energy costs for remote service calls. For each data link $(a, s) \in L_{AS}$ the consumed energy $E'_T(a, s)$ can be calculated as follows:

$$E'_T(a,s) = \begin{cases} E_T(\theta_S(a,s)), & \text{if } (\mu_1(a) \in D \wedge (s \in S_{inf} \vee \\ & s \in S_{mov} \wedge \mu_2(s) = inf) \vee \\ & \mu_1(a) = inf \wedge (s \in S_{human} \vee \\ & s \in S_{mov} \wedge \mu_2(s) \in D)) \\ 0, & \text{otherwise} \end{cases}$$

We have to spend energy costs, whenever the activity a resides on a host which is different from the host where its required service $\lambda(a)$ is executed. As a consequence, data needs to be transferred over the wireless medium. This may be true for different cases. If a workflow activity is assigned to a mobile device and the service can be found in the infrastructure (because it is an infrastructure service or a movable service running in the infrastructure), network communication is required. In the other case, whenever the activity is assigned to the infrastructure and the service resides on the mobile device (because it is an human service or a movable service executed on the mobile device), the data needs to be transferred over the network. We have no transmission costs only if the activity a and the service s are either running both on the same mobile device or both in the infrastructure.

Third, we have to consider the communication costs for the transmission of data among worfklow activities. For each such data link $(a_i, a_j) \in L_{\text{AA}}$ the consumed energy $E'_T(a_i, a_j)$ is defined in the following manner:

$$E'_T(a_i, a_j) = \begin{cases} E_T(\theta_A(a_i, a_j)), & \text{if } \mu_1(a_i) \neq \mu_1(a_j) \\ 0, & \text{otherwise} \end{cases}$$

We only have to pay the energy costs in case the communicating activities do not reside on the same host. Then, the required data must be sent over the wireless medium from the preceding to the succeeding activity. In contrast to this, activities which are assigned to the same host do not produce any energy costs for communication.

The total energy required for the execution of a workflow under the given mappings μ_1 and μ_2 is then defined as:

$$E_{total}(\mu_1, \mu_2) = \sum_{\forall s \in S_{movable}} E'_X(s) + \sum_{\forall (a,s) \in L_{\text{AS}}} E'_T(a, s) + \sum_{\forall (a_i, a_j) \in L_{\text{AA}}} E'_T(a_i, a_j)$$

All sources of energy consumption are covered in this equation. In the following, we present our algorithm that minimizes this function by finding an optimal workflow distribution. In Section 5.3, we show that this algorithm minimizes both, the sum of energy consumption over all devices *and* the energy consumed individually on each device. Both optimization goals are equivalent in our case.

5. DISTRIBUTION ALGORITHM

Our approach for energy-efficient workflow distribution is divided into two steps. First, based on the network and workflow model we construct a *cost graph*, which models the energy costs for the workflow execution on the network hosts. The nodes of the cost graph are the activities in A and the hosts in H. Its edges are annotated with weights representing the energy costs resulting from data communication or from the execution of services on the mobile devices. Second, we use this cost graph as input to a minimum cut graph partitioning algorithm. The algorithm partitions the cost graph into $|H|$ subgraphs, where each subgraph contains exactly one host and zero or more activities. The set of activities contained in the subgraph represents the fragment of the workflow that is to be executed on the particular host contained in the subgraph. Since we apply a minimum cut approach to partition the workflow, we guarantee that the energy costs for the resulting placement are minimized. In the following, we describe each of the steps involved in more detail, and we prove the optimality of the approach.

5.1 Cost Graph Construction

The cost graph $G = (V, E)$ consists of a set of nodes V and a set of weighted edges E. The graph is constructed using Algorithm 1. Initially, we create a node in V for each host of the network and each activity of the workflow, i.e. $V = A \cup H$ (line 2). Then, the set of weighted edges E is determined, where an edge is created for each data link in the workflow (L_{AA} and L_{AS}). The weight w associated with an edge $(u, v, w) \in E$ represents the energy costs of a hypothetical placement, where the source node u and target node v of the edge are assigned to different partitions (which represent different hosts).

Weighted edges are created in the following manner. First, we handle the case of activity-to-activity communication and create an

Algorithm 1 Cost Graph Construction

1: // Let $G = (V, E)$ be the cost graph to be constructed
2: $V := A \cup H$
3: **for all** $(a_i, a_j) \in L_{\text{AA}}$ **do**
4: $\quad E := E \cup \{(a_i, a_j, E_T(\Theta_A(a_i, a_j)))\}$
5: **end for**
6: **for all** $(a, s) \in L_{\text{AS}}$ **do**
7: $\quad$ **if** $s \in S_{human}$ corresponding to device d_i **then**
8: $\quad\quad E := E \cup \{(d_i, a, E_T(\Theta_S(a, s)))\}$
9: $\quad$ **else if** $s \in S_{inf}$ **then**
10: $\quad\quad E := E \cup \{(a, inf, E_T(\Theta_S(a, s)))\}$
11: $\quad$ **else if** $E_T(\Theta_S(a, s)) > E_X(s)$ **then**
12: $\quad\quad$ //service communication costs dominate - call service locally
13: $\quad\quad E := E \cup \{(a, inf, E_X(s))\}$
14: $\quad$ **else**
15: $\quad\quad$ //service execution costs dominate - call service remotely
16: $\quad\quad E := E \cup \{(a, inf, E_T(\Theta_S(a, s)))\}$
17: $\quad$ **end if**
18: **end for**

edge between any two different activities $a_i, a_j \in A$ that share a data dependency (lines 3-5). The edge (a_i,a_j) is weighted by the amount of energy required to transmit the data $\Theta_A(a_i, a_j)$ between the activities. Then, we distinguish between several cases for the invocation of service s by activity a. If the service is provided by a human, we introduce an edge weighted by the costs of transmitting the data $\Theta_S(a, s)$ between the activity and the mobile device corresponding to the human service s (lines 7-8). If the service runs in the infrastructure and is not available on the mobile device, we add an edge weighted with the costs of transmitting the data $\Theta_S(a, s)$ between the activity and the service s (lines 9-10). Finally, in case of a movable service, we have to compare the energy required to transmit the input/output data of the service to the infrastructure with the energy required for local service execution ($E_T(\Theta_S(a, s)) > E_X(s(a))$). The rationale is that a necessary criterion for running the service on the mobile device is that the service execution costs are lower than the communication costs. In case the service communication costs dominate the execution costs, the service s should always be executed where the corresponding activity a is placed (either infrastructure *or* mobile device). Hence, we must only consider the service execution costs if a is placed on a mobile device. Thus, we create an edge between a and the infrastructure node inf which is weighted by the service execution cost $E_X(s)$ (line 13). However, in case the service communication costs dominate, we create an edge between the activity and the infrastructure weighted by the communication costs (line 16).

As an example, Figure 2 shows the cost-graph constructed for the workflow in Figure 1. The nodes of the graph are the set of the activities, the infrastructure (denoted as inf), and the single device used by the human service (called d_1). Data flow links are simply mapped to edges in the graph with corresponding weights (Algorithm 1, lines 3-5), in the same way as the data flow links between activities and the user (Algorithm 1, lines 9-10). Since the execution of the 'worker service' on a mobile device causes more energy costs (10J) than the costs for the required data transmission (2J), an edge between the 'determine worker' activity and the infrastructure is created (Algorithm 1, lines 14-16). As the execution costs for the diagnosis service are smaller than the cost of transmitting its input/output, both the diagnosis service and the 'diagnose picture' activity should be executed at the same host. If the 'diagnose picture' activity is executed on the mobile device, we have to consider the execution costs of 20J. Hence, we have to create an edge with weight 20J (Algorithm 1, lines 11-13).

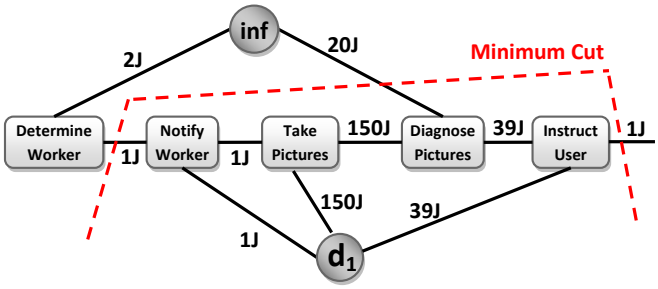

Figure 2: Cost graph created from the example workflow shown in Figure 1

5.2 Workflow Distribution

For the purpose of workflow distribution, we compute a partitioning of the cost graph that assigns each activity to a host d and, thus, represents the desired mapping μ_1. Based on the activity mapping, we can also derive the assignment of movable services to hosts for the mapping μ_2.

We use the *minimum cut algorithm* [13, 14] to determine the partitioning that minimizes the consumed energy. A minimum cut creates exactly two partitions C and $V \setminus C$ of the graph. Partition C contains the host d and zero or more activities from the set $C \cap A$ assigned to it, and partition $V \setminus C$ contains all remaining hosts $H \setminus \{d\}$ and activities $(V \setminus C) \cap A$. The sum of the weights of all edges between different partitions represents the energy consumed by the mapping. Consequently, our goal is to determine the partitioning that minimizes the sum of the weights of all edges which are cut through the partitioning.

The minimum cut is defined as follows: Given the cost graph $G = (V, E)$, the minimum $s - t$ cut of G is a partition $(C, V \setminus C)$ such that $s \in C, t \in V \setminus C$ and

$$\sum_{(u,v,w) \in E : u \in C \wedge v \in V \setminus C} w$$

is minimal among all possible partitions $C \subseteq V$. Several approaches have been proposed to find the minimum $s - t$ cut with polynomial time complexity [13, 14].

To solve our problem, we have to extend this algorithm to produce $|H|$ partitions instead of two. For this purpose, we propose Algorithm 2 that takes an iterative approach to compute the partition C with the activities for each mobile device d_i. In each iteration, we find what we call a minimum $s - T$ cut for mobile device d_i, i.e., a minimum cut that separates node d_i and all remaining hosts in the set $T = H \setminus \{d_i\}$. For this purpose, we modify the cost graph and merge all $t \in T$ into a new node t' such that all $(u, v, w) \in E$ with $v \in T$ are replaced by (u, t', w) (line 4). The idea is to create a new virtual node in the graph that represents all other hosts except for d_i. Thus, we can return to the two-partition cut problem and execute the minimum $s - t'$ cut to find the solution to our extended minimum $s - T$ cut problem (line 5). After this partitioning step, we place all activities which are part of d_i's partition on d_i (lines 6-9). We follow this approach for each mobile device. Afterwards, there may remain activities which have not been assigned to any mobile device. These activities are then assigned to

Algorithm 2 Workflow Distribution

1: // Let $G = (V, E)$ be the contructed cost graph
2: $V_{tmp} := V$
3: **for all** $d_i \in D$ **do**
4: $t' := merge(H \setminus \{d_i\})$
5: $(C, V \setminus C) = calculateMinimumCut(d_i, t')$
6: **for all** $a \in C \cap A$ **do**
7: $\mu_1(a) := d_i$
8: $V_{tmp} := V_{tmp} \setminus \{a\}$
9: **end for**
10: **end for**
11: // assign remaining activities to infrastructure
12: **for all** $a \in V_{tmp} \cap A$ **do**
13: $\mu_1(a) := inf$
14: **end for**
15: **for all** $(a, s) \in L_{AS}$ with $s \in S_{mov}$ **do**
16: **if** $\mu_1(a) \in D \wedge E_T(\Theta_S(a, s)) > E_X(s)$ **then**
17: $\mu_2(s) := \mu_1(a)$
18: **else**
19: $\mu_2(s) := inf$
20: **end if**
21: **end for**

the infrastructure (lines 12-14). Thus, we have created H partitions of the cost graph and we have found the desired mapping function μ_1 that creates an optimal mapping as we prove below. Based on this mapping, we can determine the placement of the movable services in S_{mov} on the hosts in H (lines 15-20). A movable service is placed on a mobile device only if its calling activity is also placed on the same device and the service execution costs are lower than the communication costs (lines 16-17). In all other cases, the service is executed in the infrastructure (line 19). Thus, we have found the required mapping for μ_2 and completed the workflow distribution.

For the cost graph in Figure 2, we execute the minimum-cut algorithm once, since only one mobile device is part of the scenario. The resulting cut is indicated as a dashed line. Three activities and the movable 'Diagnose Picture' service are executed on the mobile device d_1, resulting in $(1 + 20 + 1)J = 22J$ of consumed energy. Thus, we can achieve significant energy savings of $(1 + 39 + 150)J - 22J = 168J$ compared to the approach where the entire workflow is run in the infrastructure.

5.3 Optimality Discussion and Proof

In order to find the distribution of the workflow with minimal energy consumption, we calculate the minimum cut for each mobile device separately as explained in Section 5.2. Since each single cut is optimal, the overall system is also optimal if there are no conflicts, i.e. if there are *no overlapping cuts*. An overlap of two (or more) cuts (shaded area in Figure 3) means that at least one activity (residing in the overlapping region of both cuts) is claimed by two (d_i or d_j) or more devices to render the energy usage of each of those mobile devices minimal.

Of course, an optimal solution could still be found under these conditions simply by placing the activity that is requested by d_i and d_j on either one of the two such that it produces the lowest cost. However, if the overlap becomes complex, i.e. if more activities fall into overlap regions, this simple extension of the optimal cut algorithm would not suffice to create an optimal overall solution. Therefore, it suffices to show that our algorithm never produces overlapping cuts to prove its optimality.

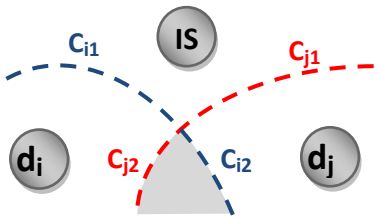

Figure 3: Two overlapping cuts

We show that there is no activity that is member of two cuts, i.e. $\forall d_i, d_j \in D, d_i \neq d_j : C(d_i) \cap C_(d_j) = \emptyset$, where $C(d_i) = \{a \in A | \mu_1(a) = d_i\}$ is the set of activities which are placed on device $d_i \in D$ based on our algorithm.

Assume there are two overlapping cuts $C(d_i), C(d_j)$ as shown in Figure 3 with $C(d_i) \cap C(d_j) \neq \emptyset$. Let c_{i2} and c_{j2} denote the overlapping part and c_{i1} and c_{j1} the non-overlapping part of cuts $C(d_i)$ and $C(d_j)$, respectively. Let $w(c)$ be the sum of weights (i.e. the cost) of a cut c. Then, $w(c_{i1}) + w(c_{i2})$ and $w(c_{j1}) + w(c_{j2})$ are the costs of the minimum cuts for d_i and d_j respectively. Note that there must be at least one activity in the overlapping area (shaded).

Since the cut $c_{i1}c_{i2}$ was chosen as minimum cut between d_i and d_j/IS, we must have $w(c_{i1}) + w(c_{i2}) \leq w(c_{i1}) + w(c_{j2})$, i.e. in particular $w(c_{i2}) \leq w(c_{j2})$. If $w(c_{i2}) = w(c_{j2})$, we can easily construct non-overlapping cuts for both d_i and d_j with minimum costs (actually $w(c_{i2}) = w(c_{j2})$ cannot happen if we assume non-zero execution costs for services, since the overlapping cut contains at least one activity). If $w(c_{i2}) < w(c_{j2})$, $c_{j1}c_{i2}$ would be a cut with less costs for d_j which contradicts the premise. Hence, there can be no overlapping cuts, which proves the optimality of our approach.

6. EVALUATION

In this section, we present our evaluation methodology and discuss the evaluation results. The goal of our evaluation is to give insight into the efficiency of our approach for a wide spectrum of different application scenarios. As there are no openly available data sets for real workflows, we rely on simulation and generate a large variety of different workflows for our evaluation.

The creation of the workflows is based on a grammar with rules to create sequential, conditional and parallel workflow structures. We apply the grammar to compose random workflow models from these partial structures, such that the number of activities per random workflow model ranges from 6 to 33. The varying size of workflows allows us to evaluate business processes of different complexity. The data flow defined by Θ_S is generated randomly according to a uniform distribution with a maximum of 5 MB per data link to allow for a variety of communication patterns. The values for Θ_A are implicitly defined by Θ_S to guarantee a consistent data flow in the workflow. That is, all data received by an activity is sent via its outgoing data flow links to other activities. Each activity is randomly assigned to either a human, a movable service or an infrastructure service. This assignment follows a uniform distribution. In order to assess the quality of our approach, we evaluate the energy savings per device and assume that a single human is interacting with the workflow using a single device.

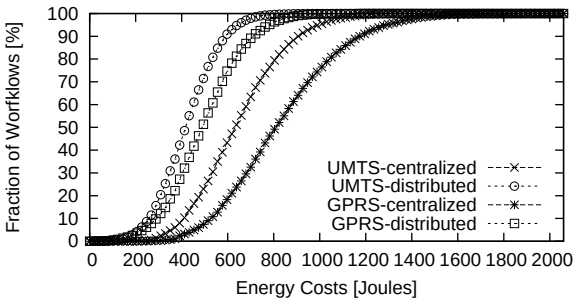

Figure 4: CDF for Absolute Energy Costs

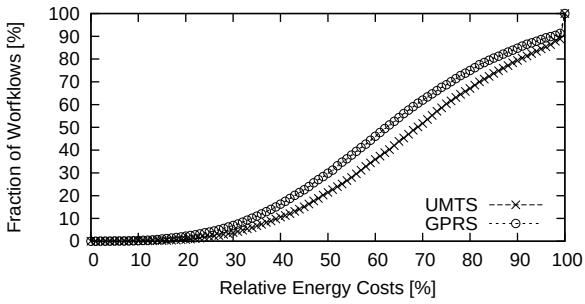

Figure 5: CDF for Relative Energy Costs

We leverage on existing energy models from the area of pervasive computing to determine the costs related to data communication and service execution. We assume that the communication is done via a wide area wireless network since such networks are globally available and provide a maximum degree of mobility for users. Therefore, we studied the energy costs for wireless communication over GPRS and UMTS based on the energy models proposed by Balasubramanian et al. [3]. We derive the drained energy based on the size of the data to be transmitted. We also assume a maximum tail time in between successive wireless connections due to the execution of time-consuming human services.

Cuervo et al. determined the energy cost for executing a given piece of code based on profiling [11]. We employ their model to derive the service execution costs. This allows a workflow designer to predict the typical energy usage of services running on a mobile device. In our experiments, we assume that the execution of a service on the mobile device consumes a random amount of energy that is drawn from a uniform distribution with a minimum of 20J and maximum of 150J.

In our evaluation, we compare the energy consumption of two different deployment scenarios. The *distributed* placement is determined according to our distribution algorithm presented in Section 5. In contrast, the *centralized* placement refers to the classical deployment scenario where the entire workflow is executed in the infrastructure. We ran 10000 different simulation experiments and measured in each experiment the required energy to execute the workflow with our distributed approach and with the centralized approach. Figure 4 shows the cumulative distribution function for the absolute energy costs, i.e., the fraction of all workflows which fall below a given energy budget. For communication over UMTS, 80% of all workflows consume less than 530J in case of our distributed approach. In contrast, only 29% of the workflows fall below this limit in case of a centralized deployment. The figures also demonstrates that 95% of all workflows do not exceed energy costs of 640J for our distributed approach, while the centralized approach requires energy up to 990J for the same fraction of workflows. We can also observe higher energy costs for both approaches when GPRS is used, since transmitting larger chunks of data consumes more energy due to the more limited bandwidth. For communication over GPRS, our approach guarantees that 80% of all workflows consume less or equal than 640J. However, only 23% of the workflows can meet the same energy constraints for the centralized execution.

Figure 5 shows the relative energy costs of our approach, measured as the fraction of energy consumed by the centralized approach. The figure depicts the cumulative distribution function, showing the fraction of workflows which remain below the given relative energy costs. While the performance for both GPRS and UMTS is similar, we can again observe slightly higher energy savings for GPRS-based scenarios due to the reasons explained above. In the best 10% of the cases, we have to spend less than 34% for GPRS and 40% for UMTS of the centralized energy consumption. Most of the workflows consume between 40% and 70% of the centralized energy consumption, depending on the degree of human interaction involved. On average, the workflow execution requires 63% of the centralized energy consumption for GPRS and 68% for UMTS. This represents a large improvement in energy usage, extending the lifetime of mobile devices significantly and reducing the costs of the underlying business processes.

7. CONCLUSIONS AND OUTLOOK

Mobility-enabled business processes become more and more widespread, and this trend will gain momentum as the device technology and the applications become more mature. This allows for porting complex workflow-based business applications to the mobile world. However, energy consumption will always be a major concern in such applications as battery capacity only grows slowly compared to computational power and network capacity.

We have proposed an approach for distributing workflows among a set of mobile and infrastructure-based hosts in order to minimize the energy drained on mobile devices. We have developed a cost graph that represents the energy consumed by the execution of workflows. Based on this cost graph, we have presented an optimal minimum-cut-based algorithm for the energy-efficient placement of workflow activities.

Our evaluation shows that our algorithm saves on average 37% of the energy for GPRS and 32% for UMTS over a purely infrastructure-based approach. In application domains that are heavily workflow-driven (like logistics and health care) this represents a significant energy saving that allows for longer operation between two recharge cycles and thus for less distractions in the daily routine of the involved personnel. In scenarios where the available battery capacities already ensure distraction-free processes, our approach allows for downgrading to cheaper technology with less capacity. Both routes lead to significant monetary savings in areas where mobile devices are indispensable tools.

Thus, our work represents an important step toward the cost-efficient and seamless integration of business processes and per-

vasive computing, enabling much more flexible workflow-driven applications that involve mobile users.

In our future work, we will investigate how the online prediction of data volumes for a concrete workflow instance can improve energy savings even further by providing more accurate knowledge.

8. REFERENCES

[1] F. Leymann and D. Roller, *Production Workflow: Concepts and Techniques*. Prentice Hall International, 1999.

[2] M. Weiser, "The computer for the 21st century," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 3, no. 3, pp. 3–11, 1999.

[3] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy consumption in mobile phones: a measurement study and implications for network applications," in *Proc. of the 9th ACM SIGCOMM conference on Internet measurement conference*, 2009.

[4] T. Bauer and P. Dadam, "Efficient distributed workflow management based on variable server assignments," *Lecture Notes in Computer Science*, vol. 1789/2000, pp. 94–109, 2000.

[5] J. H. Son, S. K. Oh, K. H. Choi, Y. J. Lee, and M. H. Kim, "GM-WTA: an efficient workflow task allocation method in a distributed execution environment," *Journal of Systems and Software*, vol. 67, no. 3, pp. 165–179, 2003.

[6] E. Bruns, B. Brombach, T. Zeidler, and O. Bimber, "Enabling mobile phones to support large-scale museum guidance," *IEEE Multimedia*, vol. 14, pp. 16–25, 2007.

[7] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin, "Diversity in smartphone usage," in *Proc. of the 8th international conference on Mobile systems, applications, and services (MobiSys)*. ACM, 2010, pp. 179–194.

[8] C. Hiesinger, D. Fischer, S. Föll, K. Herrmann, and K. Rothermel, "Minimizing human interaction time in workflows," in *Proc. od the 6th International Conference on Internet and Web Applications and Services (ICIW)*, 2011.

[9] L. Baresi, A. Maurino, and S. Modafferi, "Workflow partitioning in mobile information systems," in *Proc. of the IFIP TC8 working conference on Mobile Information Systems (MOBIS)*, 2004.

[10] IBM, *Web Services Business Process Execution Language Version 2.0*, Organization for the Advancement of Structured Information Standards (OASIS) Std.

[11] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: Making smartphones last longer with code offload," in *Proc. of the 8th International Conference on Mobile systems, Applications, and Services (MobiSys)*, 2010.

[12] A. Messer, I. Greenberg, P. Bernadat, D. Milojicic, D. Chen, T. J. Giuli, and X. Gu, "Towards a distributed platform for resource-constrained devices," in *Proc. of the 22nd International Conference on Distributed Computing Systems (ICDS)*, 2002.

[13] D. R. Karger and C. Stein, "A new approach to the minimum cut problem," *J. ACM*, vol. 43, pp. 601–640, July 1996.

[14] C. S. Chekuri, A. V. Goldberg, D. R. Karger, M. S. Levine, and C. Stein, "Experimental study of minimum cut algorithms," in *Proc. of the 8th annual ACM-SIAM symposium on Discrete algorithms*, ser. SODA '97. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1997, pp. 324–333.