Towards Opportunistic Service Composition in Dynamic Ad Hoc Environments

Christin Groba*
Supervised by: Siobhán Clarke

Lero Graduate School of Software Engineering
Distributed Systems Group
School of Computer Science and Statistics
Trinity College Dublin, Ireland
{grobac,Siobhan.Clarke}@scss.tcd.ie

Abstract. Service composition in pervasive computing envisions the collaboration of services provided by mobile and embedded devices to achieve complex application requirements. Although some research has investigated the composition of services in dynamic ad hoc settings, it is not clear how service composites manage the conflict between application layer complexities and network unreliability that may cause them to fail repeatedly. This paper outlines investigations on opportunistic service composition as a solution for managing complex service requests in mobile infrastructure-less environments. Opportunistic service composition dynamically transfers control among service providers to bind and execute services in a decentralised way. Unlike existing fragmentation and broker-based designs, it interleaves provider allocation with service execution to counteract mobility in a flexible manner. Early simulation results suggest that this approach reduces the failure probability and communication overhead of complex service requests. This illustrates that opportunistic service composition may be feasible to manage collaboration in dynamic ad hoc environments.

1 Introduction

Advances in mobile and embedded device technology have led to the vision of pervasive computing environments in which complex applications can be achieved by composing the services and capabilities of different devices [2]. Service composition is the process of discovering, selecting, and invoking suitable service providers based on a description that defines the order and nature of required services. A service composite represents such an arrangement of constituent services. The notion of combining services that are offered by different devices is only just emerging in environments that lack a managed infrastructure and a stable network topology [6]. The following scenario motivates an example for service

* Partially supported by Science Foundation Ireland grant 03/CE2/I303_1 to Lero - the Irish Software Engineering Research Centre (http://www.lero.ie).

G. Pallis et al. (Eds.): ICSOC 2011, LNCS 7221, pp. 189–194, 2012.

composition in ad hoc settings, where dynamics mainly stem from node mobility. Microphone sensors on mobile phones may soon allow for personal health monitoring such as tracking daily exposure to noise by periodically recording and analysing audio snippets. Receiving or making a call, however, blocks noise recording and results in a non-equidistant data series which could hamper further analysis. Leveraging nearby mobile phones that temporarily share their audio capabilities may be one solution to this problem. If aggregated context information rather than raw data values are required, audio tracking is complex and entails multiple sub-tasks (e.g., sampling, filtering, classifying, and geo-tagging). A single surrogate device may not have the resources or suitable software to complete the tasks alone and thus requires the collective effort of multiple peers. However, this collaboration may fail if peers are unable to communicate due to a) their mobility and thus dynamic availability or b) repeated path or message loss in the network.

2 Problem Statement

Coordinating multiple parties to accomplish service composites in dynamic ad hoc environments is challenging due to the conflict between application complexities and network limitations. Ad hoc or hastily formed networks lack a central entity that has a global system view and the ability to undertake service discovery and execution alone. Service composition has to be decentralised due to the nature of the environment which in turn requires additional coordination and synchronisation among the participators. For composites with forking and merging tasks, for example, multiple service providers have to synchronise and agree on a common merge provider. In addition, system knowledge about available services has first to be established. Routes between service providers initially do not exist because service providers are not aware of their neighbours until they announce themselves or respond to a request. Generally, decentralisation, the need for synchronisation, and acquisition of system knowledge requires service providers to communicate. Communication in ad hoc networks, however, is wireless and inherently unreliable causing message loss and failure. Mobile service providers increase the probability of failure because they frequently change the network topology creating disconnections and stale routes. Mechanisms for repair and recovery produce extra network traffic and strain the limited wireless bandwidth even more.

For these reasons, this work investigates the ***research question*** of what strategies allow for service composition in dynamic ad hoc environments and at the same time keep communication over a resource-constrained, unreliable network low. ***Key challenges*** of this research are: First, communication has to be reduced because the more the composite is exposed to an unreliable network, the more likely it is to fail. Second, service discovery in initially unknown settings nonetheless requires the exchange of information. Third, service flows, i.e., composites with splitting and merging tasks, need synchronisation among potentially unknown parties, which again requires communication.

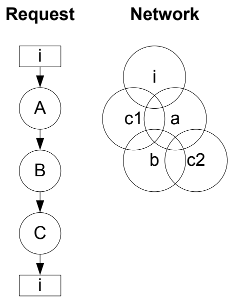
3 Proposed Solution

This work proposes an opportunistic composition protocol to satisfy complex service requests in mobile ad hoc networks. It is designed to reduce the failure probability and communication overhead of service composites. A service provider temporarily acquires the role of the coordinator who executes its local service and finds a provider for the next required service. Control over provider allocation and service execution travels from one provider to the next achieving full decentralisation. The protocol interleaves the allocation process and executes a provider immediately after it has been bound. Binding a provider only prior to its execution gives it less time to move disadvantageously and thus decreases the probability of stale routes and the need for recovery.

Late binding implies that providers are searched and selected at runtime. The proposed solution employs a directory-less architecture in which nodes are only aware of their own services. Service providers advertise themselves if they overhear a composite request and know they offer the next required service. If they provide a service at a later stage, they observe the composition and hold their advertisement back until their service is required next. This situational service advertisement avoids constant network traffic created by otherwise periodically beaconing service providers. Similarly, it does not risk traffic peaks as beaconing would when many providers simultaneously arrive in a new location. For selection, the providers' connectivity status serves as a decision criterion, if multiple providers offer the same service.

Complex application requirements provide an additional opportunity to reduce network traffic because handling them creates interaction among network nodes and thus system knowledge. Little or no extra routing messages are needed to keep an up to date view of the network topology. Routing information can be deduced from the underlying composition protocol and by observing the message exchange among service providers.

These strategies and architecture decisions address the first two research challenges. First, decentralised interleaved service allocation and execution reduces the composite's exposure to the unreliable network. Second, situational service advertisement and observation of the composition process allow for efficient service discovery in initially unknown settings. The example in Figure 1 illustrates these strategies. A client or composition initiator i issues a request with three sequentially-connected services A, B, C, and specifies that it wants to receive the overall composition result. The network consists of nodes whose lower-case node identifiers indicate that they host the corresponding upper-case service (e.g., node a hosts service A). The protocol consists of three message types: *Search* contains the remaining part of the request that has not been executed yet. *Ad* advertises that a node provides a required service and the way it is connected. *Token* transfers the coordinator role to the contained service and its provider. In addition to the protocol comments, note that a node knows about its connections only if it received a message from a neighbour. Therefore, when node a advertises itself, it is not aware of other neighbours except from the initiator i. The example shows how the request travels from one node to the next and how



Protocol

Sender	Message	Content	Comment
i	search	A-B-C-i	// i issues complex service request
a	ad	A/a, con(i)	// a provides A and connects i
i	token	A/a	// i binds a and transfers control to a
			// **a executes A locally**
a	search	B-C-i	// no provider for B advertised itself
b	ad	B/b, con(a)	// b provides B and connects to a
a	token	B/b	// a binds b and transfers control to b
			// **b executes B locally**
c1	ad	C/c1, con(i,a,b)	// c1 advertises due to search from i, a
c2	ad	C/c2, con(a,b)	// c2 advertises due to search from a
b	token	C/c1	// b binds c1 due to better connectivity
			// **c1 executes C locally**
c1	token	i	// c1 returns composite result to i

Fig. 1. Example protocol for composing sequentially-connected services

service execution interleaves the allocation process. The behaviour of node c1 illustrates situational service advertising as it does not instantly respond to i's search. Binding service C demonstrates connectivity-based provider selection.

The third research challenge addresses service flows. Following the late binding approach means that the merge provider is bound only prior to its execution. This, however, is challenging because parallel tasks imply that the coordinator role is assigned to multiple providers at the same time. These concurrent controllers may be mutually unknown and nonetheless have to identify and agree on the same merge provider. Current work explores how tracking the composition progress and connectivity status can help to find a common merge provider efficiently. In addition to that, this work investigates how service flows can be dynamically adapted to match the current provider availability.

Earlier work [3] studied how interleaving service allocation and invocation compares to an approach that first allocates all service providers and only starts executing after the last required service has been bound. ***Simulation results*** in Figure 2 show for sequentially-connected sub-services that the proposed protocol generally improves the composition success ratio, response time, and communication effort in comparison to the baseline approach.

4 Progress beyond the State-of-the-Art

Key approaches for achieving complex service requests can be classified into centralised, hierarchical, and decentralised composition management.

Centralised solutions employ a single coordinator that may be fixed or dynamically elected to allocate and invoke required services. A fixed coordinator [4] can cause long expensive routes, if it is not in vicinity of the mobile service providers. A dynamically appointed coordinator or broker [2] is more flexible because it is assigned afresh for each new request and selected based on how well it connects to required services. However, it still requires all messages to be passed through it preventing direct interaction among providers.

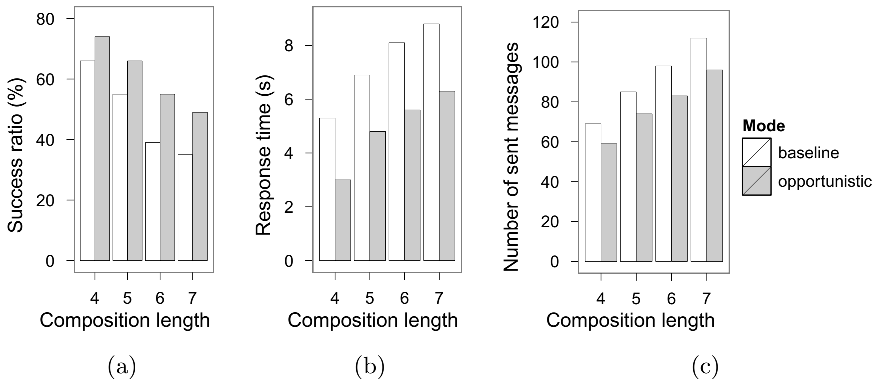


Fig. 2. Simulation results for sequentially-connected services moving at 1-13 m/s

Hierarchical architectures [1] relax this requirement and enable direct message exchange among service providers. The coordinator solely allocates services and monitors their execution. However, the coordinator requires all service providers to stay connected and to achieve this, it controls their movement. Unstructured settings in which service providers act independently may not allow for such an authority.

Decentralised approaches may contain a central component but its responsibility is limited to service discovery [5] or provider selection [6] leaving service execution and movement decisions entirely to the service providers. Other composition solutions, like [8] or the one proposed in this paper, discover, select, and execute service providers fully decentralised. Provider allocation further distinguishes all these decentralised approaches. Fragmentation-based designs [6] bind all services prior to execution. This raises two issues: First, they bind providers that may not get executed due to conditional tasks or premature termination allocating scarce resources in vain. Second, composites are less sensitive to providers that dynamically arrive and depart during execution. Designs supporting decentralised provider selection are more flexible. They bind providers during execution and only treat service flows in a special way. Merge tasks are bound either before composite execution starts [5] or early during execution together with the fork task [8]. However, the merge provider is assumed to be reachable from all parallel branches and to be available until it gets executed. This assumption may not always hold in mobile ad hoc networks. A solution for binding the merge node only right before it should execute [7] incurs a high communication overhead that seems infeasible for bandwidth-constrained networks: For each service in each branch the binding decision needs to be exchanged with all other branches. In addition, each branch selects its next provider independent from all other branches and thus risks long routes between the final decision partners.

Fully decentralised service composition with support for late binding for all sub-services has not yet been investigated in dynamic, ad hoc, and bandwidth-constraint networks. The proposed work targets this gap and studies what effect opportunistic service composition has in such settings. Its aim is to contribute to a better understanding of what service composition approaches are available and how they may be applied in mobile infrastructure-less environments.

5 Research Plan

Current work explores ways of extending the proposed protocol to handle service flows. Enabling service providers to observe the composition looks promising for binding a merge provider late and efficiently. Suitable recovery strategies will be required for compositions that started but cannot finish because a provider for a required service does not exist. The verification of protocol properties requires a formal notation of the protocol. UML state charts and Markov chains are at present analysed to assess their suitability for modelling the dynamic controller role and the gradually advancing topology knowledge. Further evidence on the feasibility of the opportunistic approach in dynamic ad hoc environments will be collected by conducting experiments. The simulation of mobile infrastructure-less networks creates a controlled environment that allows for comparing runtime characteristics of the proposed approach against related solutions.

References

1. Catarci, T., de Leoni, M., Marrella, A., Mecella, M., Salvatore, B., Vetere, G., Dustdar, S., Juszczyk, L., Manzoor, A., Truong, H.L.: Pervasive software environments for supporting disaster responses. IEEE Internet Computing 12(1), 26–37 (2008)
2. Chakraborty, D., Joshi, A., Finin, T., Yesha, Y.: Service composition for mobile environments. Mobile Networks and Applications 10, 435–451 (2005)
3. Groba, C., Clarke, S.: Opportunistic composition of sequentially-connected services in mobile computing environments. In: International Conference on Web Services (ICWS), pp. 17–24. IEEE (2011)
4. Philips, E., Van Der Straeten, R., Jonckers, V.: NOW: A Workflow Language for Orchestration in Nomadic Networks. In: Clarke, D., Agha, G. (eds.) COORDINATION 2010. LNCS, vol. 6116, pp. 31–45. Springer, Heidelberg (2010)
5. Schuler, C., Weber, R., Schuldt, H., Schek, H.J.: Scalable peer-to-peer process management - the osiris approach. In: International Conference on Web Services (ICWS), pp. 26–34. IEEE (2004)
6. Sen, R., Roman, G.-C., Gill, C.D.: CiAN: A Workflow Engine for MANETs. In: Wang, A.H., Zavattaro, G. (eds.) COORDINATION 2008. LNCS, vol. 5052, pp. 280–295. Springer, Heidelberg (2008)
7. Yildiz, U., Godart, C.: Synchronization solutions for decentralized service orchestrations. In: International Conference on Internet and Web Applications and Services (ICIW), p. 39. IEEE (2007)
8. Yu, W.: Decentralized Orchestration of BPEL Processes with Execution Consistency. In: Li, Q., Feng, L., Pei, J., Wang, S.X., Zhou, X., Zhu, Q.-M. (eds.) APWeb/WAIM 2009. LNCS, vol. 5446, pp. 665–670. Springer, Heidelberg (2009)