The Scientific Report for Exchange Visit to the ASAP Research Group at INRIA, Rennes

Shen Lin
Lancaster University
s.lin@comp.lancs.ac.uk

1 Aim of the visit

This document is a scientific report about my visit to the As Scalable As Possible (ASAP) Research Group at INRIA, Rennes (June 9 - July 2, 2008). The visit was funded by the European Science Foundation (ESF) MiNEMA Scientific Programme, Grant No. 1954. The purpose of the visit was to exchange the knowledge between the ASAP research group and the middleware research group at Lancaster University, in order to identify potential mutual collaborations.

The remainder of this report is organised as follows. Section 2 will introduce gossip-based protocols and the requirements of middleware support of this protocol family. Section 3 will briefly summarise the related research activities that Lancaster University and INRIA have been carrying on, respectively. Section 4 will present the main activities carried out during my visit at INRIA, and finally, Section 5 will illustrate our plans for the future collaboration.

2 Background

A large number of gossip protocols have been developed in the last few years to address a wide range of functionalities. A gossip-based protocol prescribes individual nodes to periodically exchange data with some randomly selected neighbours, causing information to eventually spread through the system in a "rumour-like" fashion. To deliver scalable communication in large-scale systems, gossip-based algorithms typically limit the number of contacted neighbours to a certain number M (known as the fan-out factor) and bound the size of the data exchanged at each time interval T (termed gossip round, for periodic gossip protocols). In a network with full connectivity, past research has shown that gossip protocols allow information to spread through the entire system in $\log_M N * T$ time units for a network of size N. Furthermore, thanks to the randomised and periodic exchange of information, gossip-based algorithms offer self-healing capacities and robustness to transient node failures and network partitions. Because of these benefits, gossip-based protocols have been applied to a wide range of contexts such as peer sampling, ad-hoc routing, reliable multicast, database replication, failure detection, and data aggregation.

However, most existing gossip protocols have been developed independently, resulting in a low degree of reutilisation. and a general lack of reusable software functionalities that support the development of gossip-based applications (e.g. protocols and middleware).

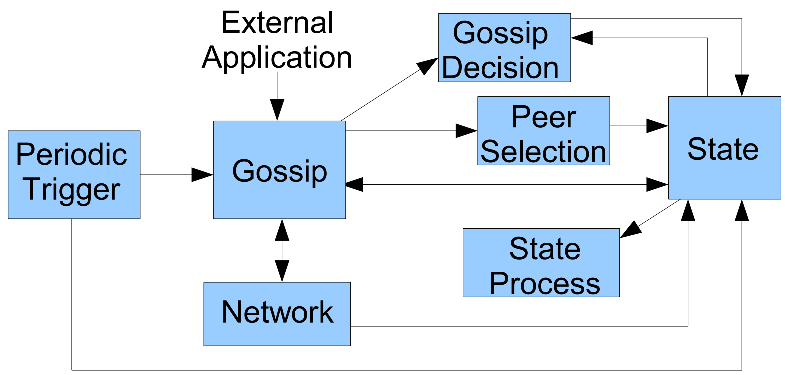

Figure 1: GossipKit's Common Interaction Model

3 Research works at Lancaster and INRIA

3.1 The GossipKit middleware framework at Lancaster University

My PhD work at Lancaster University focuses on delivering solutions to the problem that Section 2 describes. As the result of my research work, GossipKit [1, 2], a component-based middleware framework has been developed to support the operation of multiple gossip protocol instances and their interactions.

GossipKit's underlying design concept is based on our identification of a common interaction model (Fig. 1) that many gossip protocols follow. More precisely, we have observed that most gossip protocols can be decomposed into the components and interactions shown in Fig. 1 [2].

We have implemented a prototype of GossipKit on top of the Java version of OpenCom [3], a lightweight, efficient and reflective component framework that has been developed at Lancaster University. We have also used GossipKit to implement a range of gossip-based protocols such as the membership service protocols SCAMP and the peer sample service protocol, the ad-hoc routing protocols GOSSIP1 and GOSSIP2, the probabilistic broadcast protocol pbcast, the overlay topological management protocol T-MAN, and the garbage collection protocol GSGC. The source code of these protocols and of GossipKit is available on GossipKit's home page [1].

Based on the implementation of these protocols and the detailed case studies in our previous work [2], we have observed that GossipKit promotes code reuse, simplifies configuration for deploying gossip protocol middleware, supports concurrent execution of multiple protocol instances, and minimises the resource usage at runtime to a certain level.

Despite these successes, we have noticed two disadvantages of the framework's XML-based

[1] `www.lancs.ac.uk/postgrad/lins6/sub/GossipKitWeb/GossipKit.html`

configuration description. First, this description is difficult for users to understand since they need to analyse component compositions in the XML files to understand protocol behaviours. Second, for the same reason, it is more difficult and error-prone for users to write protocol descriptions in terms of component compositions. This shortage leads to the domain specific langauge (DSL) that I proposed to ESF MiNEMA for the exchange programme.

3.2 Gossip protocol development and gossip library at INRIA

The ASAP research group at INRIA, Rennes is particularly expertise in designing and evaluating gossip-based systems. The research group, which is under the direction of Dr. Anne-Marie Kermarrec, aims to provide scalable mechanisms such as gossip-based algorithms to deal with scalability and performance issues encountered in large-scaled networks. To name a few of their significant contributions:

In order to better understand the a various gossip protocols and draw a commonality from these protocols, one of the ASAP group member, Dr. Davide Frey, is responsible to systematically maintain the gossip library. This gossip library maintains individual gossip protocols as distinct components (i.e. coarser-granularity compared with GossipKit) with standard interfaces to operate on a gossip-based peer sampling service protocol.

4 The work carried out during the visit

4.1 Presentation and initial discussions

To identify potential research areas to collaborate, I gave a presentation on the first week of my visit at Rennes. On the presentation, I briefly introduced the research works at the middleware research group at Lancaster, my published work on the GossipKit middleware framework, my research interests, and my proposal on developing a domain specific language (DSL) for the abstraction and configuration of gossip protocols. Based on the presentation, I had a meeting with Dr. Kermarrec and Dr. Marin Bertier, as well as several informal discussions with many other members at ASAP group. These discussions resulted to the motivation and the design of the domain specific language below.

4.2 Design of the domain specific language for gossip programming

As discussed in Section 3.1, traditional component compositions for describing local node behaviours, which has been widely adopted for configuring component frameworks, can become more and more difficult for users to write and understand, as the number of the component (interface) types increases. Visual modeling tools such as EASYCOM and Genie can reduce the time to write component compositions, however, the complexity of the configuration (i.e. the effort for users to understand and specific component configurations) still remain.

Instead of modeling protocol abstractions locally, macro-programming such as Kairos has been developed to allow programmers to express, in a centralized fashion, the desired global behavior of a distributed computation on sensor networks. The underlying concept of such programming is related to shared-memory based parallel programming models implemented over message passing infrastructures using a small set of programming primitives. Therefore, it hides programmers from details of distributed-code generation and instantiation, remote data access and management, and inter-node program flow coordination.

Inspired by macro-programming in sensor networks, we consider to apply such programming to describe the abstraction and configuration of gossip-based protocols. There are two reasons for doing so: first, similar to programming in sensor networks, the abstractions of gossip-based protocols can be viewed as how data flows and how data are processed over the networks; and second, macro-programming helps to hide implementation details, so that gossip-based programmers can focuses on the protocol design.

To develop a macro-programming style DSL to express the abstractions of gossip-based protocols, we first designed the following list of programming primitives that expressive enough to describe the abstractions of various gossip-based algorithms.

- Type Node.
- Primitive types int, float, and boolean. Arithmetic and logic operation on these types.
- Generic type List<T>, and functions getNext() to iterate through the list.
- Built-in variables: ALL_NODES is used to express all the available nodes in a network, and SIZE indicates the SIZE of a list.
- State declaration: "State [data type list] [size] stateVar;". This defines that structure of the state that every node in the network should maintain.
- Built-in function getOneHopNeighbours() can be used to get physical neighbours for gossip-based algorithms that run on wireless networks.
- Call to GossipKit's components: "nodeVar.ComponentName(specify the target states)[component parameterisation];". Because the DSL is designed to configure GossipKit's component compositions. It is inevitable to explicitly refer to certain components in the component framework. We have minimised the number of component types that need to be explicitly referred in the programme to two: PeerSelection and StateProcess. Other component types can be implicitly implied.
- The if-else statements.
- The "atomic" block, which indicates that the distributed statements needs to be executed as an atomic operation. This is useful if some gossip-based protocols require each node to perform exactly one pairwise state exchange and process at a gossip round.

```
package example.orderedSlicing;
import example.rps.RPS;
OrderedSlicing (GetData[Get(Property, RandomNumber)],
                AddData[Add(Property, RandomNumber)]) {
  State[Property, RandomNumber][1] state;
  Node n, neighbour;
  for (n in ALL_NODES) {
    for (;;) {
      sleep(5000);
      neighbour = n.RPS.GetPeers.getNext();
      atomic {
        neighbour.SwapMisorderValuePairs(n.state, neighbour.state);
        n.SwapMisorderValuePairs(n.state, neighbour.state);
}}}}
```

Figure 2: Use Gossip DSL to describe a gossip-based ordered slicing protocol

To verify the expressiveness of this language, we have applied it to describe eight gossip protocols that differ by their communication styles, underlying network types, and state processes. Fig. 2 illustrates how the macro-programming Gossip DSL can be used to describe a gossip-based ordered slicing protocol [4].

The remainder of this subsection describes the programming process to convert the DSL to per-node runtime deployment.

The architecture of the program execution is depicted in Fig. 3. Using this DSL, an abstractions of an individual gossip protocol can be described as a macro-program, which can then be parsed by a parser that compiler generation tools such as Javacc can generate. When parsing the gossip protocol abstractions, the embedded interpreter converts the macro-programmed gossip protocol abstractions to local per-node component configuration files. The implementation of the interpreter is independent of the parser, so that various interpreters may be applied to convert the gossip protocol abstractions to configurations of different frameworks such as GossipKit and INRIA's gossip library. Finally, the configuration file can be distributed to nodes so that GossipKit can (re)configure on each node based on this configuration specification.

4.3 Pilot Implementation

To study the feasibility of developing such a programming language for configuring the GossipKit framework, I have carried out an early implementation on the parser of the language on the second and the third week during my stay at INRIA. The programming tool I adopted is Javacc because of my familiarity. The implementation of the parser has been completed when I was at INRIA. The resulted tool can successfully parse programs that describe the above mentioned eight case studies, and report syntactic errors on any malformed programs.

4.4 Second Discussion

On the third week of my visit, I had a meeting with Dr. Kermarrec and Dr. Marin Bertier to present my design and implementation progress on the domain specific language development. We have also decided on the evaluation approaches. The main evaluation criteria we considered were: simplicity, expressiveness, efficiency, and correctness of the

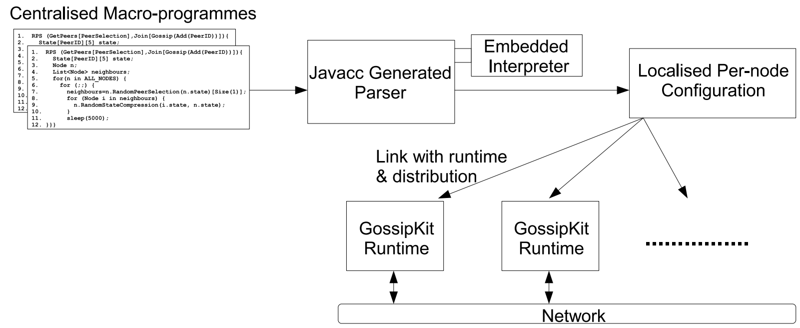

Figure 3: Gossip Domain Specific Programming Architecture

language. Simplicity will be analysed based on the comparison against existing configuration approaches; Expressiveness will be analysed using representative case studies; and finally efficiency and correctness will be ensured by comparing the performance of gossip protocols using the gossip DSL programming against the performance of gossip protocols developed from scratch.

4.5 Others

During my stay at INRIA, I had opportunity to exchange ideas with other researchers working at the ASAP group in different sub-areas of gossip programming. In particular, One of Dr. Kermarrec's PhD student and I have identified a common research interest in exploiting the synergies between coexisting gossip protocols. We plan to look into this research area in the future, in order to identify a way to systematically identify potential synergies and exploit them dynamically.

5 Future collaboration

Our future collaboration will be based on the two research topics (see Section 4) that we have identified during my visit at INRIA. As the next step, I will continue developing the language interpreter, which aims to convert the DSL to the localised component configuration for GossipKit. Together with this domain specific language, GossipKit will then be provided to members at ASAP group to use, and their feedbacks will be collected as an important evaluation criteria on GossipKit's DSL. Meanwhile, to demonstrate the generality of the DSL, Dr. Davide Frey will work on the implementation of a separate language interpreter, which can convert the same DSL to the configuration of INRIA's gossip library. We have planed to submit a conference paper to ICDCS'09 based on this work. We will also look for possible collaborations on exploiting synergies between coexisting gossip protocols.

Travel costs

The following table summarises the travel costs that are related to the exchange programme. The images of the receipts, tickets, and boarding passes are attached at the end of this report.

Item	Date	Cost
Return Air Tickets, Manchester - Rennes	09.06.2008 02.07.2008	169.78 pounds
Taxi, Lancaster - Manchester Airport (No train connection in the early mornings)	09.06.2008	70.5 pounds
Train Ticket, Manchester Airport - Lancaster	02.07.2008	10.25 pounds
Bus ticket, Rennes Airport - INRIA	09.06.2008	1.2 Euro
Bus ticket, INRIA - Rennes Airport	02.07.2008	1.2 Euro
Bus ticket, Lancs Rail Station - Lancaster Univ.	02.07.2008	1.3 pounds
Total Cost	-	251.83 pounds and 2.4 Euro

Acknowledgments

I would like to thank Dr. Anne-Marie Kermarrec and all the members in the ASAP research group at INRIA for their hospitalities and the interesting discussions. I would also like to thank the MiNEMA project of the European Science Foundation for providing me the opportunity to take this very fruitful exchange visit.

References

[1] Shen Lin, Francois Taiani, Gordon S. Blair. *GossipKit: A Framework of Gossip Protocol Family.* In Proc. of the 5th Middleware for Network Eccentric and Mobile Applications workshop, 11-12 September 2007, Magdburg, Germany.

[2] Shen Lin, Francois Taiani, Gordon S. Blair. *Facilitating Gossip Programming with the GossipKit Framework.* To appear in Proc. of the 8th IFIP International Conference on Distributed Applications and Interoperable Systems, 4-6 June 2008, Oslo, Norway.

[3] M. Clarke, G. Blair, G. Coulson, et al. *An efficient component model for the construction of adaptive middleware.* In Proc. of IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing, 2001.

[4] Mark Jelasity and Anne-Marie Kermarrec. *Ordered Slicing of Very Large-Scale Overlay Networks.* In Proc. of Peer-to-Peer Computing, 2006.

Current Folder: **INBOX** **Sign Out**

Compose Addresses Folders Options Search Help Lancaster University

[Previous | Next] [Delete & Prev | Delete & Next] [Message List]

Reply Reply All Forward ☐ As Attachment Delete ☐ Bypass Trash Move to: INBOX Move

Subject: FW: Expedia travel confirmation - Rennes, France - 9 Jun 2008 - (Itin# 11723159844)
From: "Redburn, Liz" <l.redburn@lancaster.ac.uk>
Date: Wed, 7 May, 2008 12:18 pm
To: s.lin6@lancaster.ac.uk
Options: View Full Header | View Printable Version | View as plain text | Download this as a file | View Unsafe Images| Add to Addressbook

Insurance form attached

Liz Redburn

Travel Officer

Computing Department

'phone - 01524 510315

fax - 01524 510488

From: travel@support.expedia.co.uk [/src/compose.php?send_to=travel@support.expedia.co.uk]
Sent: 07 May 2008 13:17
To: Redburn, Liz
Subject: Expedia travel confirmation - Rennes, France - 9 Jun 2008 - (Itin# 11723159844)

Travel Confirmation

Thank you for booking your trip with Expedia.co.uk. View this itinerary online for the most up-to-date information.

This email is your ATOL confirmation invoice and should be retained until your travel has been completed.
It is your evidence of your contract with us and you will not receive a paper copy.
Expedia Inc, 7 Soho Square, London W1D 3QB

Need a hotel or a car or an activity or service in Rennes? Here are some options we've found for you.

Appart'City Rennes Ouest **£43.98** per night
Citea Rennes **£43.98** per night
Campanile Rennes Cleunay **£47.77** per night
Car Rental - Economy Midsize Full Size
Attractions & Services - Sightseeing Dining options Ground transportation Attraction passes

Search for more hotels

For fully inclusive car hire, click here.

Search for more Attractions & Services

Receive Expedia.co.uk **newsletters**

Buy Travel Insurance with Mondial

printed receipt satisfies this requirement.

Expedia.co.uk itinerary number: **11723159844**

Main contact: Shen Lin
E-mail: l.redburn@lancaster.ac.uk

Home phone: (1524) 510315

Traveler and cost summary

Shen Lin	Adult	**Add Frequent Flyer number(s)**	£85.58
		Taxes & Fees	£84.20
	Print a receipt	Total amount charged to credit card (Visa/Delta)	**£169.78**

Flight summary

Mon 9-Jun-08 **- Expedia Special Fare**

Manchester (MAN) Depart 07:00 Terminal 3	to	**Southampton** (SOU) Arrive 08:05	172 mi (277 km) Duration: 1hr 5mn	BE **Flybe** Flight: **7360**

3**Economy/Coach Class**, DE HAVILLAND DHC-8 DASH 8-400 DASH 8Q

Southampton (SOU) Depart 11:00	to	**Rennes** (RNS) Arrive 13:05	200 mi (322 km) Duration: 1hr 5mn	BE **Flybe** Flight: **1025**

3**Economy/Coach Class**, DE HAVILLAND DHC-8 DASH 8-400 DASH 8Q

Total distance: 372 mi (599 km) Total duration: 2hr 10mn (5hr 5mn with connections)

Wed 2-Jul-08 **- Expedia Special Fare**

Rennes (RNS) Depart 13:30	to	**Southampton** (SOU) Arrive 13:30	200 mi (322 km) Duration: 1hr 0mn	BE **Flybe** Flight: **1026**

3**Economy/Coach Class**, DE HAVILLAND DHC-8 DASH 8-400 DASH 8Q

Southampton (SOU) Depart 15:35	to	**Manchester** (MAN) Arrive 16:40 Terminal 3	172 mi (277 km) Duration: 1hr 5mn	BE **Flybe** Flight: **7369**

3**Economy/Coach Class,** DE HAVILLAND DHC-8 DASH 8-400 DASH 8Q

Total distance: 372 mi (599 km)

Total duration: 2hr 5mn (4hr 10mn with connections)

Airline rules & regulations

General Rules

Image removed This is a restricted fare ticket.

Image removed Tickets are nontransferable.

Image removed This fare is non-changeable and non-refundable.

Image removed This fare is not guaranteed until purchase.

Image removed Carriers reserve the right to change the flight schedule at any time prior to departure.

Note that this trip is priced using a fare which is limited in availability. The itinerary you have selected currently meets the criteria for this fare.However, please note that **any** changes to this itinerary may cause this fare to be invalidated.

Refunds and Changes

This fare is nonrefundable.

Lost Tickets

Lost tickets are only replaced if the airlines' lost ticket procedures are followed. Lost ticket applications are charged a fee at the airlines' prevailing rates.

Image removed Please read important information regarding airline liability limitations.

Figure 1: Boarding passes (Machester - Rennes)

Figure 2: Train ticket (Manchester Airport - Lancaster Rail Station)

Figure 3: Bus tickets

Acceptance Check In 1st Ltd

Client ID	LAN010	Private Hire ID	4789
Client	Ms Redburn	Movement ID	9724
Company	Lancaster University	Status	Firm
Client Ref 1	07904683507	Passengers	1
Client Ref 2		Distance	

First Pick-up	Shen Lin	Destination	Manchester Airport Terminal 3
Pick-up Date	Mon 09/06/2008 Time 03:30	Arrival Date	Mon 09/06/2008 Time 05:00
Single Journey	Yes	Leave Date	Mon 09/06/2008 Time
Vehicle To Stay	No	Back Date	N/A Time

First Pick-up Instructions

House 3
Graduate College
Lancaster University

Tel: 07904683507

Destination Instructions

Seats	Vehicle Description	Vehicle No	Price	VAT %	VAT	Total
1	Toyota Prius	1	£60.00	17.5	£10.50	£70.50
Movement Totals			£60.00		£10.50	£70.50

Driver Description	Vehicle No	Driver Description	Vehicle No
Driver	1		

References

Purchase Order Number: 118104684
Flight Number: BE 7360

Passenger Destination: Southampton
Passenger Names: Shen Lin

The above client agrees to pre hire the above vehicle for a period as and from the date of this agreement until termination of this agreement which shall occur on the last day of December 2008. It is agreed by all parties that the hire of the above vehicle will have the services of a chauffeur included for the period of hire. The charges are as per our agreed rates. There will be no penalties or charges for non-usage. The customer agrees that he/she has read the Terms and Conditions.

Signature		Print Name		Date	