Project no.	609551
Project acronym:	SyncFree
Project title:	*Large-scale computation without synchronisation*

European Seventh Framework Programme
ICT call 10

Deliverable reference number and title:	D.1.1 First report on Work Package 1
Due date of deliverable:	11th December 2014
Actual submission date:	11th December 2014
Start date of project:	15th October 2013
Duration:	36 months
Organisation name of lead contractor for this deliverable:	Trifork
Revision:	0.00
Dissemination level:	CO

Contents

1 Milestones in the Deliverable 1

2 Project objectives for the period 1
2.1 Overview 1
2.1.1 D1.1 [from M01 to M06] 1
2.1.2 D1.2 [from M06 to M18] 1

3 Contractors contributing to the Deliverable 1
3.1 Trifork Leeds 1
3.2 Koç 2
3.3 UNL 2
3.4 INRIA 2
3.5 KL 2

4 Work progress and achievements during the period 3
4.1 Progress overview 3
4.2 Progress overview and contribution to the research field 4
4.2.1 D1.1 - Natural language requirements 4
4.2.2 D1.2 - Formal-language requirements 4
4.2.3 Adaptive Replication 5
4.2.4 Internet of Things 9
4.2.5 Presentations in M12 10
4.2.6 Documents 10
4.2.7 Future work 10
4.2.8 Other work carried out 11

Glossary 14

A Optimistic Control for a Critical System 14

B Location of Full Documents 14

1 Milestones in the Deliverable

WP1.1 has reached the following milestone:

Mil. no	**Mil. name**	**Date due**	**Actual date**
MS1	CRDT consolidation in a static environment	M12	M12

The corresponding tasks are:

Task no	**Task name**	**Date due**	**Actual date**	**Lead contractor**
D.1.1	Natural–language requirements	M01	M12	Trifork

2 Project objectives for the period

2.1 Overview

The following sections provide an overview of the project objectives for the reporting period in question.

2.1.1 D1.1 [from M01 to M06]

Natural-language requirements: Initially, the use cases selected by the project partners will be described formally in natural language. Such a description consists of a outline of the targeted supporting architecture and a list of the informal constraints the system must ensure.

2.1.2 D1.2 [from M06 to M18]

Formal-language requirements: In this task, we will describe the selected use cases using a mathematical notation suitable for reasoning, verification and computation, as required by the later Work Packages. Constraints concerning scalability, quality of service, fault tolerance, and other global guarantees will be specified qualitatively and quantitatively. This formalisation will build upon recent work on eventual consistency semantics. The behaviour and composition of data types will be defined in terms of programming models, using the first results of Work Package 4 (WP 4).

3 Contractors contributing to the Deliverable

3.1 Trifork Leeds

Amadeo Ascó, Tom Benedictus, Rune Skou Larsen and Kresten Krab Thorup.

3.2 Koç

Burcu Kulahcioglu Ozkan and Serdar Tasiran.

3.3 UNL

Carla Ferreia.

3.4 INRIA

Jordi Martori Adrian, Marc Shapiro and Pascal Urso.

3.5 KL

Peter Zeller.

4 Work progress and achievements during the period

4.1 Progress overview

D1.1: The "Natural-language requirements" document was created and made available in SyncFree website, where six different use cases with their own characteristics and constraints are described using natural-language. The use cases were compiled from a group of use cases provided by the industrial partners and are described in some details in this document, also providing an outline of the informal constraints the system must ensure and the targeted supporting architecture.

D1.2: "Formal-language requirements" is not a deliverable until March 2015 (M18) but major progress has been archived. Currently there are three documents and the adaptive replication has been incorporated into this Work Package and the overall project.

All the use cases presented in D1.1 have been already modelled mathematically together with their constraints to help understanding the problem better, using a standard mathematical representation. Using this initial mathematical representation three of the six use cases have already been represented in TLA+. The TLA+ suit is well suited for concurrent and distributed systems, and it is an open-source project, which provides a language with a parser and syntax checker, model checker and simulator that will allow completing all the tasks associated with Work Package 1 (WP 1). The already represented use cases has been syntactically checked and many simulations conducted, focusing in the verification and assurance of application-level properties for applications built using Conflict-free Replicated Data Types (CRDTs), given that the key research question being explored by the SyncFree project is to determine the extent to which CRDTs can enable the building of practical systems.

In a WP 1 meeting conducted in Paris from 9^{th} to 11^{th} September 2014 it was agreed to be of academic and industrial interest to embrace more areas within the scope of SynchFree, particularly the study and implementation of adaptive replication.

An Adaptive Geo–replication algorithm has been proposed, described in the document "Adaptive Geo–Replication Based on Ant Colony Algorithm", and presented in M12. The algorithm is based in the well known Ant Colony algorithm commonly used in optimisation problems. An initial implementation has been provided and incorporated to a GUI application for demonstration proposes that could be further extended to allow simple tests to be carried out in a visible and controlled manner.

Five presentations were provided in M12 regarding the different areas covered up to now by WP 1.

4.2 Progress overview and contribution to the research field

The following sections present more information about the two deliverables covered in the pass twelve months. It must be noted that the second deliverable (D1.2) is not due in this period but main advances have been achieved which are also presented here.

4.2.1 D1.1 - Natural language requirements

The only deliverable for this period for WP 1 corresponds to the D1.1 document with title "Natural language requirements" which provides a general informal description in natural language of some use cases selected by the project partners, [5]. These use cases were selected from those originally proposed by some of the industrial partners.

Six use cases have been described by the WP 1 in the SyncFree project. All use cases make use of distributed databases having one or more Data Centres (DCs) acting as central authority. In some use cases, the distributed database also spans to mobile devices being sometimes connected - both with each other and the central authority.

A simple overview of the selected use cases is presented in Table 1.

Another document was created with an extended discussion of the FMK use case with the title "Optimistic Control for a Critical System" with extra details on its current implementation and requirements, which proposes some improvements such as divergence visibility, divergence guarantees, pseudo-transactions (CRDTs) and acting on conflicts, [6].

4.2.2 D1.2 - Formal-language requirements

This document provides a brief presentation of the natural language requirements, which shows the special nature of the paradigm shift forcing large-scale distributed applications away from Relational Database Management System (RDBMS), [4].

The document provides two mathematical representations of the use cases, and it is available in SyncFree GitHub. All use cases have been represented initially mathematically (common representation) and several of the use cases have already been represented in TLA+, syntax checked with SANY, model checked and emulated with TLC and verified. Work in progress as not all use cases has been fully represented in TLA+ and others have incomplete representations. This work is not due util M18.

The design of a dynamic behaviour exploration tool is being undertaken for applications built on top of the execution platforms developed in SyncFree, using Riak, work part of WP 4. In the process of building a tool that orchestrates multiple Riak instances running on a single machine in order to systematically explore the states that different replicas can get into, and how this affects application-level properties.

No.	Application	Provider	Brief Description
1	Ad Counter	Rovio	Web pages and apps are presenting commercial ads that must be displayed e.g. 100,000 times while not repeated too often on one device.
2	Leader Board	Rovio	For each game, records must be available about the leading player. Some players use several devices.
3	Wallet	Rovio	Debiting and keeping track of balance is is critical for a stable in-game economy.
4	Shared Medical Record (FMK)	Trifork	Sharing information on a patients current medication between healthcare professionals and the patient himself.
5	Business to Business (B2B)	Trifork	The ability to present new products, place orders, and keep track of delivery using tablet computers is the basis for this B2B application that connects thousands of stores with the manufacturer.
6	Festival	Trifork	Tens of thousands of participants in one of the worlds largest two week music festivals are concurrently rating their experience and the momentary results are made generally available.

Table 1: Overview of selected use cases.

4.2.3 Adaptive Replication

It has been proposed and accepted to extend the coverage to consider Adaptive Replication. A common technique to reduce latency is the replication of data between different DCs in a system with multiple DCs distributed around the world, as shown in Figure 1.

But keeping multiple replicas is an expensive commodity given the increase in storage requirements and bandwidth as a write operation needs to be propagated to all the other DCs with a replica. The data could be placed in only one DC, in which case there is not any replication, or a copy could exist in all the available DCs, which it is known as full replication. Alternatively the data may be located in some but not necessarily all the DCs such that the number, location and data is determined at run time, which it is known as adaptive replication.

The replications may be grouped, base on its existence, into two types; **static replication** where a replica persist until it is deleted by a user or its duration expires, and **dynamic replication** where the creation and deletion of a replica are managed automatically and normally directed by the access pattern of the data used by the users [8]. In static replication the major drawback is their inability to

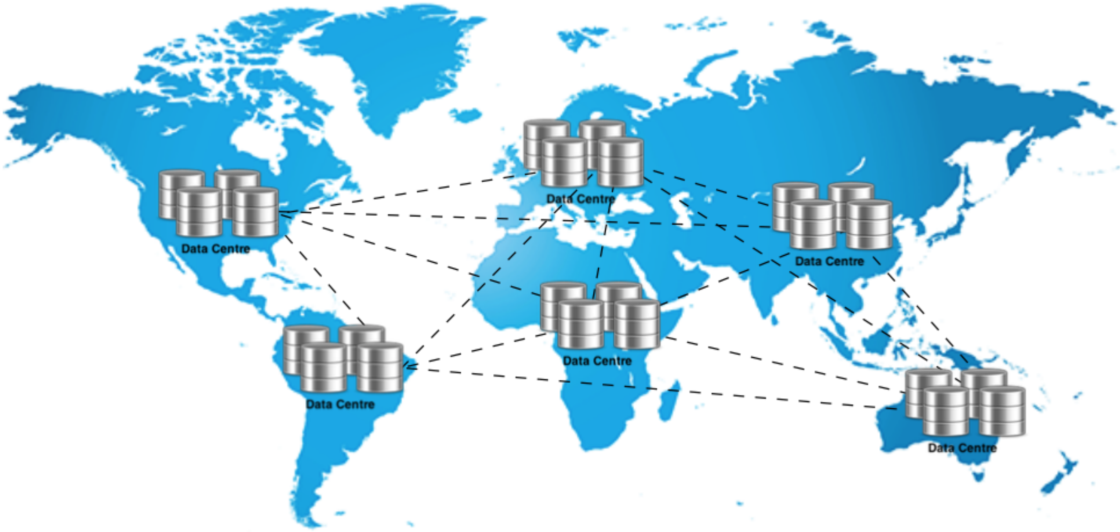


Figure 1: Multiple DCs distributed around the world.

adapt to changes in the access pattern of the data used by the users.

Also there are two types of replication based on their effect on the data; **partial replication** is concerned with the number of parts the full data is composed of, all of which may be located in different parts of the overall system, i.e. DCs, within a DC in different nodes or at the client-side [12, 7, 13], whereas **adaptive geo–replication** is concerned in what data and where the data or part of the data is located within the overall system of DCs and how many replicas exist simultaneously, [9, 10, 14, 1, 2, 11].

Partial Replication: It is concerned with the different parts of the data, [7, 13], and finding data types that allow breaking the data into smaller parts, [12]. It avoids replicating large data structures so helping to reduce the bandwidth and latency. The main principal is that not all the full data is always required, as shown in Figure 2. This introduces the need to find data structures which allow breaking the data into parts without loosing information, and maintaining the data integrity and required invariants.

This requires to define new CRDTs which allows splitting the data into different parts that may be placed in multiple DCs, [12].

Adaptive Geo–replication: This is also know as "Adaptive Location of Replicas". In the example, shown in Figure 3, data reads/writes to DCs 1 and 2 making the data replica to move from DC 3 to DCs 1 and 2 ensuring that the data is closer to where the reads and writes are requested based on the specified objectives and constraints.

Adaptive Geo–replication is concerned with:

1. Location: On which DCs to place the replicas so it is

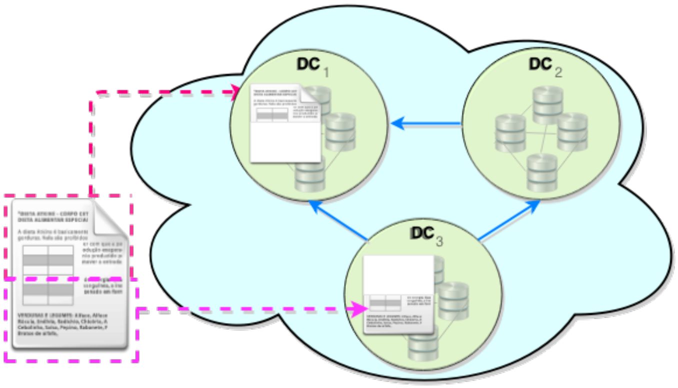


Figure 2: Example of Partial Replication of data between multiple DCs.

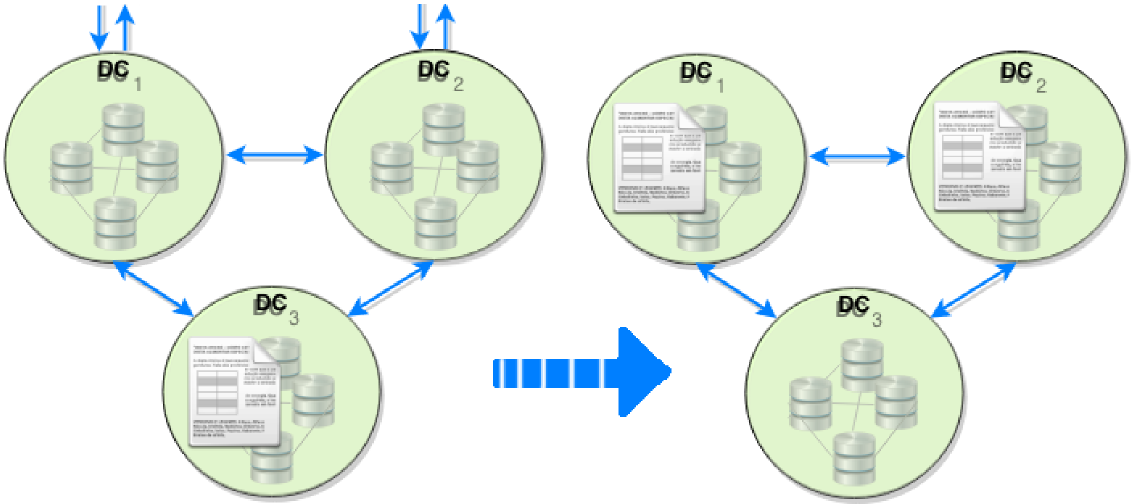


Figure 3: Example of adaptive geo–replication between DCs.

   (a) Improved the latency: reduce distance between user and replica,
   (b) Improved the data transmission quality.

2. <u>Selection</u>: Which data to replicate.

3. <u>Number</u>: How many replicas to have so it is
   (a) Reduced unnecessary replicas which
      i. Reduces storage consumption and
      ii. Reduces required network bandwidth.

An Initial proposal of an algorithm has been presented to the WP 1 for the "Adaptive Geo–replication". The algorithm decides without the need of human intervention where, when and how many times the data is replicated with the main objective of reducing the latency and network traffic (reduce usage of bandwidth).

In general terms any read operation in a data centre reinforces the need for a replica of the data in such DC, similarly but perhaps with a different degree it happens with the write operations. But write operations decrease the need for a replica of the data in the other DCs with replicas, so eventually these DCs will not have any replica of the data. Given that we do not want to keep replicas when there are not necessary replica strength will decay as time pass, but always making sure that the data is present (replicated) at least in a pre-set number of DCs. There is also a question about how the replication should decay as time pass, e.g. linear, exponential or arctangent.

This proposed algorithm uses simple and fast operations to adapt to the changing access pattern of the data. Furthermore, some or all of the parameters could be determined at run time by some other adaptive approach which has into account the cost function which is described in the algorithm paper.

This algorithm implies:

1. For reads on a DC, the DC does not require to communicate any information to any of the other DCs. A read only needs to use resources in the DC, where the read is initially requested from, so no extra network traffic is imposed on the systems.

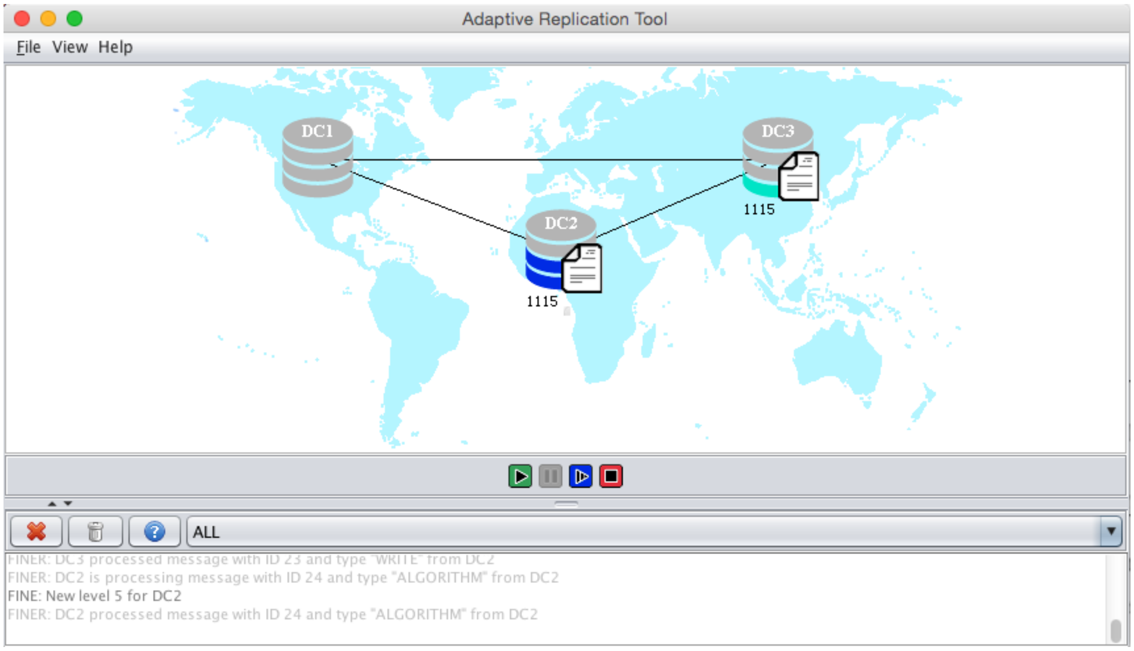
   In the particular case where a read is executed on a DC, without a replica of the data, storage and processing power in the DC will be required and the request will be forwarded to its closest DC with a replica. This operation incurs in extra network traffic but if it is repeated too often the extra network traffic will be removed by replicating the data in the requesting DC where the data was initially requested.

2. On a write the DC receiving the original request will (eventually) transmit it to the other DCs, which have a replica of the data, so no need to add extra network traffic as this is the normal approach. But extra data will be sent to the other DCs to notify them of the number of writes the changes refer to, which will depend of the type of CRDT approach used, i.e. op-based or state-based. In some cases not every write is transmitted to the other DCs, such is the case of the state-base approach, so it would be required to keep some track of the number of merged writes.

   Also the merging of the data should only be executed after the replication strength has been calculated and when it is still higher than zero, which will reduce unnecessary operations.

3. On data without any reads and writes on any DC, the number of replicas will be reduced as time passes by the temporal effect, until the data is only replicated in the minimum number of DCs.

The algorithms has already been implemented in Java as part of a GUI tool, as shown in Figure 4, that allows to easterly show its principals in a graphical manner, which was presented in the M12 meeting. The tool is available in SyncFree GitHub repository. This tool also allows to run scripts, represented in XML format,



```
FINER: DC3 processed message with ID 23 and type "WRITE" from DC2
FINER: DC2 is processing message with ID 24 and type "ALGORITHM" from DC2
FINE: New level 5 for DC2
FINER: DC2 processed message with ID 24 and type "ALGORITHM" from DC2
```

Figure 4: General view of the algorithm GUI tool for three DCs.

which execution can be controlled by starting, stepping, pausing and stopping them. The description of the system composed of DCs is also contained in the test script together with the operations to execute, i.e. read and write operations.

A document has been written which explains the proposed algorithm in more detail, [3], and contains some related literature, which is available in SyncFree GitHub. Also a simple comparison has been included in the document for when there is no replication, the data is only in one DC, when the data is fully replicated, a replica exists in all the DCs, and when the proposed algorithm is used.

4.2.4 Internet of Things

It was presented the need to extend the scope of the overall project to cover also Internet of Things (IoT) to maintain the industrial relevance of the project up to date with the trends, technologies and emerging needs.

The IoT is a world in which machines, cloud-based applications and networked devices connect with each other, to share and process data where big data analytics facilitate intelligent decision making. It covers all verticals such as retail point of sales, automotive, asset and fleet management, healthcare, manufacturing, security, etc. Also many of the world biggest industrial players are already actively taking part in this revolution.

It is proposed to incorporate in the project two main scenarios; first of all regarding the high burst of incoming data, known as time series, and secondly referring to the listening of events which then actions must be taken upon them, known as events. So new data types are required to efficiently fulfil the requirements of these scenarios.

The work covered in WP 1 regarding IoT refers to the presentation of this idea and providing the initial document with a description and some links to extra documentation and existing implementations which will be updated as further work is carried out. Also the introduction and representation of new data types are tasks for future work in WP 1.

4.2.5 Presentations in M12

List of presentations in M12 from WP 1.

1. WP 1 Progress Overview
2. Large Project Failure: Adaptive Replication
3. TLA+ description and model checking
4. Adaptive Location of Replicas: An algorithm based on the Ant Colony algorithm
5. Shared Medical Record (FMK)

4.2.6 Documents

List of current documents from WP 1.

1. D1.1 Natural language requirements
2. Optimistic Control for a Critical System
3. Adaptive Geo-Replication Based on Ant Colony Algorithm (work in progress)
4. Formal Mathematical Requirements (work in progress)

4.2.7 Future work

Some work to be carried out in WP 1:

1. Finalise the TLA+ representations: extend to the remaining use cases and investigate the formal refinement (correct implementation) relation between a high-level CRDTs specification and a lower-level implementation, e.g. static verification (rather than model checking) of application proper ties.
2. Run validation and simulations of the TLA+ representations.

3. Update the algorithm document.

4. The effort has concentrated on modelling applications that use transactions and CRDTs, but not more elaborate mechanisms for ensuring invariants such as reservations and escrow, and not more elaborate and efficient replication mechanisms such as partial or adaptive replication. Since the initial exploration of the verification and tooling issue has yielded encouraging results, it is envisage to investigate these more elaborate mechanisms, their correctness and how they enable programmers to guarantee application-wide invariants.

5. Incorporate the algorithm in the platforms developed in SyncFree (part of Work Package 2 (WP 2)) which includes the generalisation of the interface between the "Adaptive Replication Layer" and the "Across DCs Replication Layer".

6. Provide guidance on how to set the parameters in the algorithm, which should come from the test results and the study of the algorithm behaviour in different scenarios.

7. Create papers/article regarding the algorithm and how it compares to other approaches.

8. Study and propose data types for IoT.

Other future work directly emerging from the current work curried out in WP 1 are:

1. Implement the algorithm in Erlang (part of WP 2) to easterly interact with the platforms developed in SyncFree.

2. Execute comparative tests with other existing approaches present in the literature (also part of Work Package 5 (WP 5)).

Further work that may be of interest:

1. Extend the algorithm to use self tuning, if test results show this as appropriate, otherwise provide advise on setting the parameters based in some use cases.

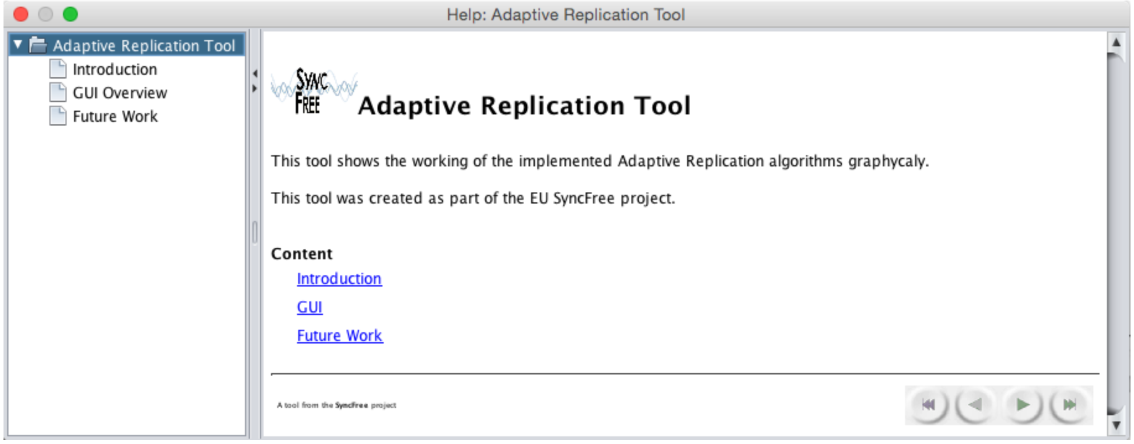
2. Extend the algorithm to consider other constraints such as the data must always be present (replicated) in a specific DC to fulfil legal requirements.

4.2.8 Other work carried out

Implementation of the proposed Adaptive Geo–replication Algorithm
To help in the understanding and dissemination of the Adaptive Geo–replication algorithm principals, and as a initial prove of concept an initial implementation of the algorithm has been coded in Java but given that the algorithm was envisaged to take part in a multi layer infrastructure where the management of the replicas

between DCs is the responsibility of the “Across DCs Replication Layer” then a simple implementation of the “Across DCs Replication Layer” was implemented in Java within the tool. The “Across DCs Replication Layer” has already been implemented in the platforms developed in SyncFree, using Erlang programming language and Riak, and future work will be carried out to incorporate the algorithm implementation within these layers infrastructure.

A Java GUI tool has been developed which implements the algorithm and allows to test it (work in progress). The tool provides help in form of HTML files displayed on GUI components in the tool, those components may be visualised from different parts of the tool, which work as links, shown in Figure 5. The bottom part of the tool displays the “log view” with the operations executed, which may be filtered to only display the messages of interest. In the middle of the display between the “graphical view” of the replica strength progress (at the top of the GUI) and the “log view” (at the bottom of the GUI) is the controls of the test, more information is available on the help. This tool and algorithm were coded under the scope of WP 2 and WP 5, and will be further improved and studied within the WP 1, WP 2 and WP 5.



Adaptive Replication Tool

This tool shows the working of the implemented Adaptive Replication algorithms graphycaly.

This tool was created as part of the EU SyncFree project.

Content

Introduction

GUI

Future Work

A tool from the SyncFree project

Figure 5: General view of the help for the Adaptive Replication Tool.

It is intended that once the tests have been conducted to asses the algorithm performance, when compared with other algorithms, then a paper(s)/article(s) will be created in part by extending the current document with the algorithm description and incorporating the results from these tests.

References

[1] Cristina L. Abad, Yi Lu, and Roy H. Campbell. Dare: Adaptive data replication for efficient cluster scheduling. In *Proceedings of the 2011 IEEE International Conference on Cluster Computing*, CLUSTER '11, pages 159–168, Washington, DC, USA, 2011. IEEE Computer Society.

[2] S. Abdul-Wahid, R. Andonie, J. Lemley, J. Schwing, and J. Widger. Adaptive distributed database replication through colonies of pogo ants. In *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International*, pages 1–8, March 2007.

[3] Amadeo Ascó. Adaptable location of replicas based on ant colony algorithm. Technical report, SyncFree, September 2014.

[4] Amadeo Ascó and Tom Benedictus. Formal mathematical requirements. Technical report, SyncFree, July 2014.

[5] Tom Benedictus. D1.1 natural language requirements. Technical report, SyncFree, May 2014.

[6] Tom Benedictus, Jordi Martori, Rune Skou Larsen, and Pascal Urso. Optimistic control for a critical system. Technical report, SyncFree, September 2014.

[7] Iwan Briquemont. Optimising client-side geo-replication with partially replicated data structures. Master's thesis, Louvain-la-Neuve, September 2014.

[8] Xiaohua Dong, Ji Li, Zhongfu Wu, Dacheng Zhang, and Jie Xu. On dynamic replication strategies in data service grids. In *Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on*, pages 155–161, May 2008.

[9] Junsang Kim; Won Joo Lee; Changho Jeon. A priority based adaptive data replication strategy for hierarchical cluster grids. *International Journal of Multimedia & Ubiquitous Engineering*, 9(6):127–140, 2014.

[10] R. Kingsy Grace and R. Manimegalai. Dynamic replica placement and selection strategies in data grids- a comprehensive survey. *J. Parallel Distrib. Comput.*, 74(2):2099–2108, February 2013.

[11] Thanasis Loukopoulos and Ishfaq Ahmad. Static and adaptive distributed data replication using genetic algorithms. *J. Parallel Distrib. Comput.*, 64(11):1270–1285, November 2004.

[12] PPoPP. *Conflict-free Partially Replicated Data Types*, 2015.

[13] D. Serrano, M. Patino-Martinez, R. Jimenez-Peris, and B. Kemme. Boosting database replication scalability through partial replication and 1-copy-snapshot-isolation. In *Dependable Computing, 2007. PRDC 2007. 13th Pacific Rim International Symposium on*, pages 290–297, Dec 2007.

[14] Zhe Wang, Tao Li, Naixue Xiong, and Yi Pan. A novel dynamic network data replication scheme based on historical access record and proactive deletion. *J. Supercomput.*, 62(1):227–250, October 2012.

Glossary

B2B Business to Business

CRDT Conflict-free Replicated Data Type

DC Data Centre

FMK Shared Medical Record

IoT Internet of Things

RDBMS Relational Database Management System

WP 1 Work Package 1

WP 2 Work Package 2

WP 4 Work Package 4

WP 5 Work Package 5

A Location of Full Documents

List of documents:

- D1.1
- Critical Optimistic Eventual Consistency
- Adaptive Ant Replication (in progress)
- D1.2 (in progress)