

Leinonen, Harry; Lumiala, Veli-Matti; Sarlin, Riku

**Working Paper**

## Settlement in modern network-based payment infrastructures: Description and prototype of the E-Settlement model

Bank of Finland Discussion Papers, No. 23/2002

**Provided in Cooperation with:**

Bank of Finland, Helsinki

*Suggested Citation:* Leinonen, Harry; Lumiala, Veli-Matti; Sarlin, Riku (2002) : Settlement in modern network-based payment infrastructures: Description and prototype of the E-Settlement model, Bank of Finland Discussion Papers, No. 23/2002, ISBN 952-462-009-X, Bank of Finland, Helsinki, <https://nbn-resolving.de/urn:nbn:fi:bof-20140807585>

This Version is available at:

<https://hdl.handle.net/10419/211927>

**Standard-Nutzungsbedingungen:**

Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden.

Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen.

Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte.

**Terms of use:**

*Documents in EconStor may be saved and copied for your personal and scholarly purposes.*

*You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public.*

*If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.*



---

# BANK OF FINLAND DISCUSSION PAPERS

---

23 • 2002

Harry Leinonen – Veli-Matti Lumiala –  
Riku Sarlin  
Financial Markets Department  
19.9.2002

## Settlement in modern network-based payment infrastructures

– description and prototype of  
the E-Settlement model

Suomen Pankin keskustelualoitteita  
Finlands Banks diskussionsunderlag

---

**Suomen Pankki  
Bank of Finland  
P.O.Box 160  
FIN-00101 HELSINKI  
Finland  
☎ + 358 9 1831**

**<http://www.bof.fi>**

---

# BANK OF FINLAND DISCUSSION PAPERS

---

23 • 2002

Harry Leinonen – Veli-Matti Lumiala – Riku Sarlin  
Financial Markets Department  
19.9.2002

## Settlement in modern network-based payment infrastructures

– description and prototype of the E-Settlement model

The views expressed are those of the authors and do not necessarily reflect the views of the Bank of Finland.

<http://www.bof.fi>

ISBN 952-462-009-X  
ISSN 0785-3572  
(print)

ISBN 952-462-010-3  
ISSN 1456-6184  
(online)

Suomen Pankin monistuskeskus  
Helsinki 2002

# Settlement in modern network-based payment infrastructures

## – description and prototype of the E-Settlement model

Bank of Finland Discussion Papers 23/2002

Harry Leinonen – Veli-Matti Lumiala – Riku Sarlin  
Financial Markets Department

### Abstract

Payment systems are undergoing rapid and fundamental changes stimulated largely by technological progress especially distributed network technology and real-time processing. Internet and e-commerce will have a major impact on payment systems in the future. User demands and competition will speed up developments. Payment systems will move from conventions that were originally paper-based to truly network-based solutions.

This paper presents a solution – E-Settlement – for improving interbank settlement systems. It is based on a decentralised approach to be fully integrated with the banks' payment systems. The basic idea is that central bank money, the settlement cover, is transferred as an encrypted digital stamp as part of the interbank payment message. The future payment systems would in this model operate close to the Internet/e-mail concept by sending payment messages directly from the sending bank's account/payment server to the system of the receiving bank with immediate final interbank settlement without intervening centralised processing. Payment systems would become more efficient and faster and the overall structure would be come straightforward. The E-Settlement and network-based system concept could be applied with major benefits for correspondent banking, ACH and RTGS processing environments.

In order to assess this novel idea the Bank of Finland built a prototype of the E-Settlement model. It consist of a group of emulated banks sending payments to each other via a TCP/IP network under the control of a central bank as the liquidity provider and an administration site monitoring the system security.

This paper contains an introduction to network-based payment systems and E-Settlement, the specifications of the E-Settlement model and the description, results and experiences of the actual E-Settlement prototype.

Key words: network-based payment systems, settlement systems, interbank settlement, payment system integration

# Katteensierrot verkkopohjaisissa maksujärjestelmissä – E-Settlement-mallin kuvaus ja prototyyppi

## Suomen Pankin keskustelualoitteita 23/2002

Harry Leinonen – Veli-Matti Lumiala – Riku Sarlin  
Rahoitusmarkkinaosasto

### Tiivistelmä

Maksujärjestelmät ovat merkittävien ja nopeiden muutoksien edessä, mikä johtuu tekniikan kehityksestä erityisesti hajautetun verkkotekniikan ja reaaliaikaprosessin alueilla. Internetillä ja elektronisella kaupankäynnillä tulee olemaan merkittäviä vaikutuksia maksujärjestelmiin. Käyttäjävaatimukset ja kilpailu nopeuttavat kehitystä. Maksujärjestelmät siirtyvät alun perin paperipohjaisista ratkaisuista todellisiin verkkopohjaisiin menettelyihin.

Tämä keskustelualoite esittelee ratkaisun pankkien välisen katteensierrojen tehostamiseksi: E-Settlement. Se perustuu katteensiertojärjestelmän hajauttamiseen ja täydelliseen integrointiin pankkijärjestelmien kanssa. Perusideana on, että keskuspankkiraha eli maksujen kate siirtyy salakirjoitettuna digitaalisena leimana osana pankkien välisiä maksusanomia. Tulevaisuuden maksujärjestelmät toimisivat tässä mallissa hyvin samanlaisella periaatteella kuin Internetin sähköposti. Sanoma välitetään suoraan lähettävän pankin tili- ja maksuserveriltä vastaanottavan pankin järjestelmiin ilman keskitettyjä käsittelyvaiheita. Maksujärjestelmät tehostuvat ja nopeutuvat sekä niiden kokonaisstruktuuri yksinkertaistuu. E-Settlementiä ja verkkopohjaista maksujärjestelmästruktuuria voidaan soveltaa kirjeenvaihtajapohjaisiin, clearingkeskusta käyttäviin ja RTGS-järjestelmiin.

Uuden idean arvioimiseksi Suomen Pankki rakensi prototyypin E-Settlement-mallista. Se koostuu ryhmästä emuloituja pankkeja, jotka lähettävät toisilleen maksuja TCP/IP-verkon kautta. Järjestelmää valvovat likviditeettiä toimittava keskuspankki ja erityinen hallintayksikkö turvallisuuden ja tekniikan osalta.

Tämä keskustelualoite sisältää johdannon verkkopohjaiseen maksuinfrastruktuuriin ja E-Settlementratkaisuun määrittelyineen sekä kuvauksen, tulokset ja kokemukset varsinaisesta E-Settlement-prototyypistä.

Asiasanat: verkkopohjainen maksujärjestelmä, katteensiertojärjestelmät, pankkien väliset katteensierrot, maksujärjestelmien integrointi

# Foreword

E-Settlement is a very new and different concept compared to the existing settlement conventions. It is based on efficient usage of network-based solutions and modern security technology. The Bank of Finland decided to build a prototype in order to assess the new idea and to be able to concretely present the system in operation.

The idea was first presented in the Bank of Finland discussion proposal “Re-engineering payment system for the e-world”<sup>1</sup>. A follow-up project seemed to be needed in order to make the model more concrete and illustrative. A project was launched with the following objectives

- specify the model in detail
- find out the stability and reliability of the model
- check attainable efficiency and cost levels
- assess the attainable security level
- evaluate decentralised administration issues
- build and test a prototype with the basic core functionalities.

The project started in November 2001 with a tender procedure. The specifications were ready by end of February 2002 and the prototype was built from March through to May. During summer 2002 the different parts of the prototype were integrated and tested. It was ready for complete demonstrations in August 2002.

The prototype includes the basic core elements of a network-based payment infrastructure eg TCP/IP network, direct addressing and transfers, multilevel PKI encryption, decentralised account management, liquidity control and continuous automated reconciliation etc. The prototype shows that the E-Settlement concept is technically possible and reliable. The cost-benefit analyses show that clear benefits can be found in all kinds of payment system environments (correspondent banking, ACH and RTGS) by utilising a network-based approach for payments and settlements.

This discussion paper contains the main reports from the E-Settlement prototype project divided into separate sections

- Section 1: Introduction to E-Settlement
- Section 2: E-Settlement system architecture specification
- Section 3: E-Settlement system security specification

---

<sup>1</sup> Harry Leinonen ”Re-engineering payment systems for the e-world” Discussion Proposal 17/2000, Bank of Finland.



Section 4: E-Settlement cost/benefit analysis

Section 5: E-Settlement prototype description and preliminary experiences

This paper can be found in electronic format on the E-Settlement Internet webpage<sup>2</sup> of the Bank of Finland together with slide-presentations and additional documents eg bank and central bank interfaces as well as technical architecture specifications.

The authors want to thank everybody participating in the project. The international payment systems experts that have contribute by commenting on the E-Settlement idea during the project. We want also to thank the Finnish banking industry for the comments received throughout the project from the payment systems cooperation bodies. Most of the specification and prototype building work was out-sourced and therefore we especially want to thank the two IT-companies, one that has written most of the specifications and built the prototype e-settlement module and the other that has constructed the hardware secure module and the public key security solution for the prototype, for forming a very good and successful team work. We hope that the prototype and this discussion paper will stimulate further studies in network-based solutions aimed at implementing efficient new infrastructures.

---

<sup>2</sup> The link is [www.bof.fi/sc/e-settlement](http://www.bof.fi/sc/e-settlement)

# Contents

Abstract .....	3
Tiivistelmä .....	4
Foreword .....	5
<b>Section 1: Introduction to E-Settlement.....</b>	<b>9</b>
<b>Section 2: E-Settlement system architecture specification .....</b>	<b>15</b>
<b>Section 3: E-Settlement system security specification .....</b>	<b>67</b>
<b>Section 4: E-Settlement cost/benefit analysis.....</b>	<b>115</b>
<b>Section 5: E-Settlement prototype description and preliminary experiences .....</b>	<b>127</b>
<b>References .....</b>	<b>167</b>



# Section 1

## INTRODUCTION TO E-SETTLEMENT

The views expressed here are those of the project. They are at the moment preliminary and open for all comments. The views do not necessarily reflect the views of the Bank of Finland.

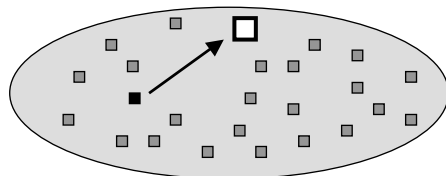
# 1 INTRODUCTION TO E-SETTLEMENT

E-Settlement is a **settlement method for the next generation of payment systems**, which can nonetheless be employed already in current payment systems. The next generation of payment systems will be network- and real-time-based. In order to fulfil customer needs, payment systems will be standardised and globalised, providing access and services to everyone, free of national technical barriers. These systems will have to be open and supportive of interoperability (see figures 1a and 1b). In this environment, the settlement method will need to entail immediate finality between all the different participating service-providing institutions (mainly banks).

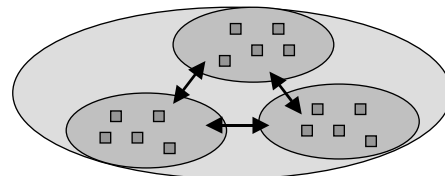
The settlement process itself will have to be a well-integrated part of the payment process, with end-to-end control from sending to receiving bank. In order to support rapid payment transfers, the settlement method must also support real-time processing. An efficient settlement system also supports continuous reconciliation – for immediate error detection. It is also important that most of the security and control features be built into the system, to enable immediate reaction.

In an open global payment system, the sending customer should need to know only the receiver's account number in order to be able to send a payment. The **international account number** (IBAN or its equivalent) would be the only information needed for routing the message properly through the system, just as email addresses or international gsm-telephone numbers route messages (emails or sms messages) internationally. Interoperable bridges are needed to route payments between the accounts of different service providers. As regards payments, these bridges must transfer payment messages as well as interbank settlements. This is the main difference as compared to other messaging systems and this is where E-Settlement introduces new electronic possibilities.

**Figure 1a: Payments can be made by transferring funds by addressing directly the receiving account in a common account number space**

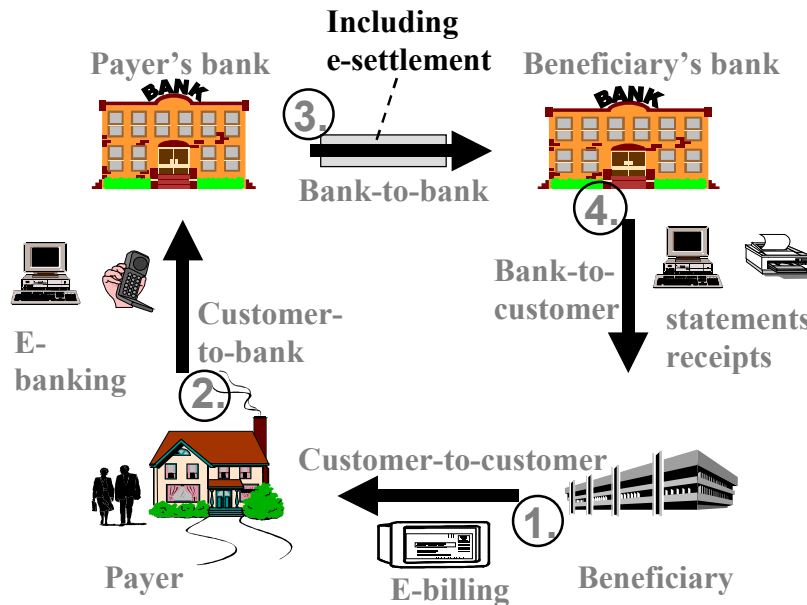


**Figure 1b: The common account number space is divided into sub-spaces belonging to service providers which are connected via interoperable system bridges.**



The E-Settlement process is designed to support and accomplish final interbank settlement in central bank money in real-time for end-to-end payments in straight-through-processing (STP) mode. It should be seen as one essential part of the whole **payment (credit transfer) circle** as described in figure 2.

**Figure 2: E-settlement is part of the credit transfer circle, which provides efficient electronic communications between participants in a payment.**



(1) The payer receives a bill or other instruction from the beneficiary concerning a payment to be made. (2) The payer then sends the instruction to his bank for processing and routing to the beneficiary's bank. (3) This interbank leg includes E-Settlement, so that the beneficiary's bank receives both the payment message and the final settlement. (4) The beneficiary's bank can then inform the beneficiary as to the incoming/final payment. This credit-push/credit-transfer type of payment is the most convenient and efficient in the network real-time world. It has fewer processing and transportation legs than electronic credit/debit card payments, direct debits, or electronic cheques. In credit transfers, the payer's bank identifies its customer, checks the payment instruction, and debits the payer's account; the beneficiary's bank checks the settlement and credits the beneficiary's account. In the future real-time world, payments will be processed within seconds in the same way as email and sms messages are now processed.

The E-Settlement solution should be seen as part of the **future payment infrastructure** that will support increased

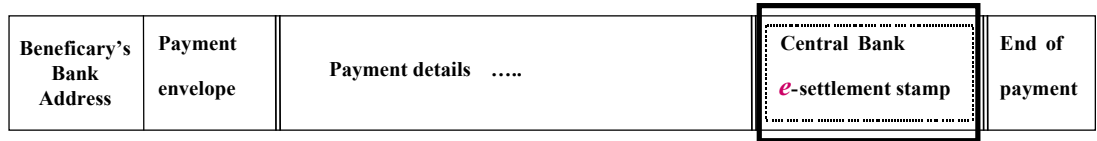
- e-commerce via Internet
- real-time security and money market deals and transfers
- mobile payments (currently GSM-based but soon UMTS-based)
- cross-border volumes.

The payment world (for all kind of payments) will be changing as will all other messaging systems, from slower-paced batch processing to immediate real-time service, integrated directly with user systems.

E-Settlement provides a solution that can be integrated into current systems, using a part of the existing infrastructure, and hence it facilitates a gradual change from current structures to new e-based structures.

The fundamental idea of E-Settlement is attachment of a digital E-Settlement stamp to the current payment messages, as shown in figure 3.

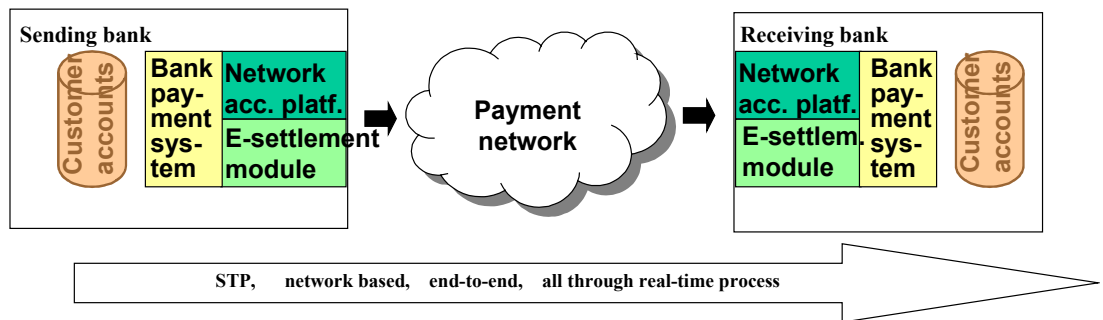
**Figure 3: The digital e-settlement stamp is part of the payment message**



The E-Settlement stamp is added to the payment message and serves to transfer central bank money from payer's bank to beneficiary's bank. Final settlement is part of the payment message, in the form of an electronic central bank draft or central bank e-cash for interbank settlement purposes.

The stamp is protected by very strong and modern cryptographic technology. These stamps are produced and decoded by E-Settlement modules situated close to banks' payment systems, as shown in figure 4.

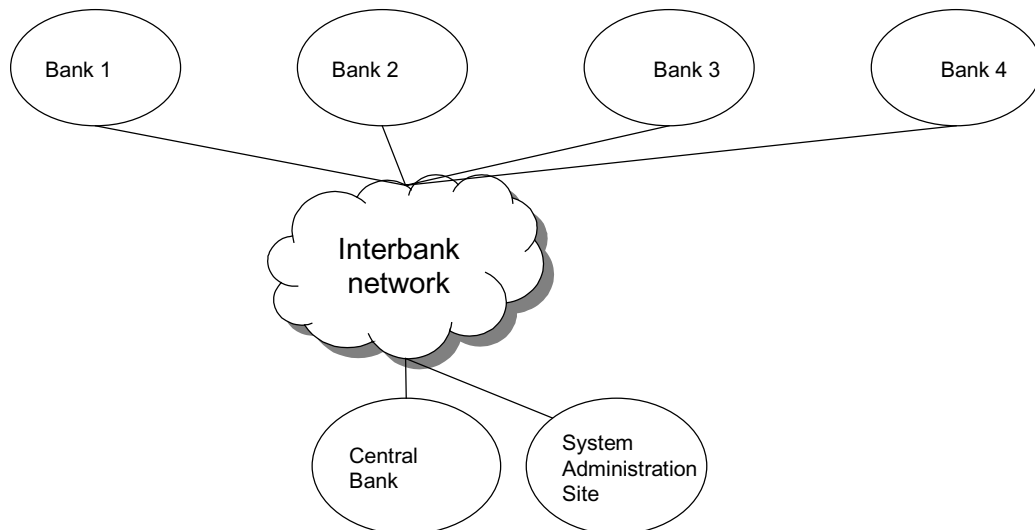
**Figure 4: E-settlement stamps are produced by e-settlement modules, which are closely integrated with banks' payment systems.**



The E-Settlement modules are tamper-resistant devices provided by central banks to each bank. These are closely integrated with banks' payment systems, eg directly integrated with the SWIFT-net access platform (CBT). This makes settlement transfers a highly automated part of payment processing. Integrating the new settlement process will be quite straightforward, given that it will be done on the access platform (eg SWIFT CBT) level. In traditional RTGS systems, banks' settlement accounts are located in the centralised RTGS system. In the E-Settlement system, each bank's settlement account is distributed to the bank's own processing site. Each bank has access to its own account, as before, but is much more closely integrated in a more automated and efficient way. The distribution of central bank money in electronic format to banks' payment platforms is the essential feature of the E-Settlement approach. The distributing E-Settlement modules need to be highly secure and to meet at least the same security standards as do traditional RTGS systems. The system should also be generally open and independent, to support the various payment networks used by banks.

A dedicated **interbank network** (figure 5) is needed to link together all participating banks and the central bank, for the purpose of processing payments.

**Figure 5: A dedicated interbank network connects all banks and the central bank with each other for payment processing.**



The interbank payment network is the essential part in a network-based payment system. All banks can address each other directly and send payments to each other without a centralised processing and routing site. This is the essential new paradigm introduced by Internet communications (TCP/IP-networks). All participants can operate independently; they need only enough networking capacity to meet their own needs. System administration is needed only for administration purposes, ie not for payment processing. The new SWIFTnet network, introduced by SWIFT, is one that can support direct communications between all participants. There are also national dedicated payment networks with the same capability, eg the interbank network Pankkiverkko in Finland.

For settlement purposes, banks need **liquidity**. Liquidity is transferred by the central bank to the system (ie E-Settlement modules) at the start of the day. It can be increased during the day by the central bank via liquidity transfers or payments to the banks. At the end of the day, liquidity is transferred back to the central bank. The liquidity in the settlement modules is thus composed of positive balances of central bank money, originally in the traditional form of reserve deposits, intraday credits etc, but transferred from the centralised system in the morning to distributed E-Settlement modules to be employed during the day in the E-Settlement system. In the evening the liquidity will be transferred back to the centralised accounts for overnight bookings.

In a true real-time environment, there is generally little scope for the various types of advanced liquidity saving features, based on delaying/queuing of payments. Customers are waiting for direct confirmation of their payments. A bank that is often obliged to inform its customers that payments are queued – waiting for liquidity – will lose customers. In the real-time environment, customers expect direct delivery.

Still, the E-Settlement module could contain basic queuing facilities for situations in which the available liquidity is not sufficient or customers are willing to accept delays. These would be decentralised queues, designed for different levels of complexity. Bilateral netting could be accomplished in the distributed E-Settlement system through bilateral netting requests to check whether there are transactions also queued at the other end. Multilateral netting requires a centralised netting agent. Different types of netting and advanced liquidity saving features would complicate the system. It is advisable to keep the basic system very simple; possible add-on services should be provided separately.

The system's **security features** must be carefully designed. The settlement balance and all security keys need to be in tamper-resistant environments and all the encryption algorithms must be highly reliable. There should be no possibility of system intrusion, and any type of 'hacking' should be



immediately detectable. The system will be closed, with settlement money circulating among a limited number of trustworthy users. The system will include automated reconciliation at end-of-day, from time to time during the day and in connection with each transaction. In a network-based environment, all parts – centralised and distributed – must be well secured. A digital/electronic version of the four-eyes principle has to be implemented.

**High availability** must also be ensured in the distributed system. In a distributed system, a malfunction will generally affect only one participant at a time and only those payments to and from that particular participant. In order for the participant to re-establish normal operations quickly, there should be back-ups and mirrored devices for all critical components.

**The main benefit** of E-Settlement is that it enables redesign of the whole payment system process in an efficient way, using new network possibilities. It thereby creates the next generation of payment systems infrastructures and makes the settlement process more efficient. Payment systems will change considerably in the near future due to modern technology and it would be an advantage to modernise the settlement conventions at the same time.

The cost-advantage of the E-Settlement system is in the low processing costs of adding the E-Settlement stamp that enables instant final settlement in central bank money. The extra processing cost of adding the E-Settlement stamp will be practically nil. It will be an integral part of the payment process itself. Banks need only invest in low cost equipment. The very low transaction costs for of E-Settlement will enable banks also to transfer payment flows from centralised processing centre systems to more efficient decentralised network-based communications. The bottlenecks created by centralised resources will disappear and even the dependence on critical centralised resources will be dramatically reduced. E-Settlement could offer a solution for integrating the euro-area payment systems, and a multi-currency version could serve an even larger area. In order to achieve large-scale benefits via the E-Settlement model, the number of participating banks and the payment flows must be sufficiently large.

## **Section 2**

### **E-SETTLEMENT**

# **SYSTEM ARCHITECTURE SPECIFICATION**

The views expressed here are those of the project. They are at the moment preliminary and open for all comments. The views do not necessarily reflect the views of the Bank of Finland.

# TABLE OF CONTENTS

<b>TABLE OF CONTENTS .....</b>	<b>16</b>
<b>ABBREVIATIONS.....</b>	<b>17</b>
<b>1 INTRODUCTION.....</b>	<b>18</b>
1.1 E-SETTLEMENT SYSTEM.....	18
1.2 ENSURING INFORMATION CONSISTENCY IN E-SETTLEMENT SYSTEM.....	19
1.3 ENSURING CONSISTENCY OF AUDIT TRAILS .....	20
1.4 TO BE SPECIFIED LATER.....	20
1.5 DOCUMENT SCOPE .....	20
1.6 DOCUMENT STRUCTURE.....	21
<b>2 DESIGN PHILOSOPHY.....</b>	<b>22</b>
2.1 DESIGN RATIONALE .....	22
2.2 LAYERED DESIGN .....	23
<b>3 STRUCTURE OF E-SETTLEMENT SYSTEM .....</b>	<b>26</b>
3.1 BANK PAYMENT SYSTEM BAPS .....	27
3.2 CENTRAL BANK PAYMENT SYSTEM CBPS .....	28
3.3 BANK E-SETTLEMENT MODULE BEM.....	29
3.4 CENTRAL BANK E-SETTLEMENT MODULE CEM.....	31
3.5 SYSTEM ADMINISTRATION SITE SAS.....	32
3.6 E-SETTLEMENT SYSTEM INTERFACES.....	34
<b>4 CORE PROCESSES.....</b>	<b>36</b>
4.1 PAYMENT PROCESSES.....	37
4.2 LIQUIDITY MANAGEMENT PROCESSES .....	46
4.3 RECONCILIATION PROCESSES .....	48
4.4 CONTROL PROCESSES .....	55
<b>5 EXCEPTION HANDLING.....</b>	<b>58</b>
5.1 EXCEPTIONAL SITUATIONS.....	58
5.2 DISASTER RECOVERY .....	59
<b>6 MANAGEMENT ARCHITECTURE.....</b>	<b>60</b>
<b>7 SECURITY ARCHITECTURE .....</b>	<b>61</b>
7.1 LAYERED APPROACH TO SECURITY .....	61
7.2 FACETS OF E-SETTLEMENT SECURITY .....	62
7.3 KEY SECURITY MECHANISMS USED IN E-SETTLEMENT.....	64

## ABBREVIATIONS

Abbreviation	Meaning
BAIF	Bank Interface – E-Settlement system external interface to banks
BAPS	Bank Payment System –bank system linked with E-Settlement system
BBIF	BEM-BEM interface – E-Settlement system internal interface
BCIF	BEM-CEM interface – E-Settlement system internal interface
BEM	Bank E-Settlement Module – E-Settlement system element
BIC	Bank Identifier Code - technical code that uniquely identifies a financial institution.
BNET	Interbank network
BOD	Beginning-of-day
BSIF	BEM-SAS interface – E-Settlement system internal interface
CA	Certification authority
CB	Central Bank
CBIF	Central Bank Interface – E-Settlement system external interface to CBs
CBPS	Central Bank Payment System – CB system linked with E-Settlement system
CEM	Central Bank E-Settlement Module – E-Settlement system element
CSIF	CEM-SAS interface – E-Settlement system internal interface
ECB	European Central Bank
EOD	End-of-day
ESN	E-Settlement Network – E-Settlement system element
FCAPS	Faults, Configuration, Accounting, Performance, Security – areas of management
HSM	Hardware security module
IBAN	International Bank Account Number
PKI	Public Key Infrastructure
RTGS	Real-Time Gross Settlement
SAS	System Administration Site – E-Settlement system element
SWIFTNet	IP-based secure interbank network serviced by SWIFT
TMN	Telecommunications Management Network, a model for managing networks

# 1 INTRODUCTION

## 1.1 E-Settlement system

Practically all RTGS systems currently in use are centralized systems, where liquidity management, payment processing and data storage are handled by the hosting central bank. See Figure 1 below for a schematic picture. The client banks interact with the central system using applications that participate in the workload of the RTGS system only partly or not at all. Depending on the RTGS system, the client applications are offered by the RTGS provider, or the banks can access RTGS directly by writing their own client applications. In some systems client access can even be terminal based. Security control (hardware and/or software based) is required to ensure access to authorized clients only using secured communications. In a centralized RTGS settlement is done using the liquidity available in each bank's centralized RTGS account.

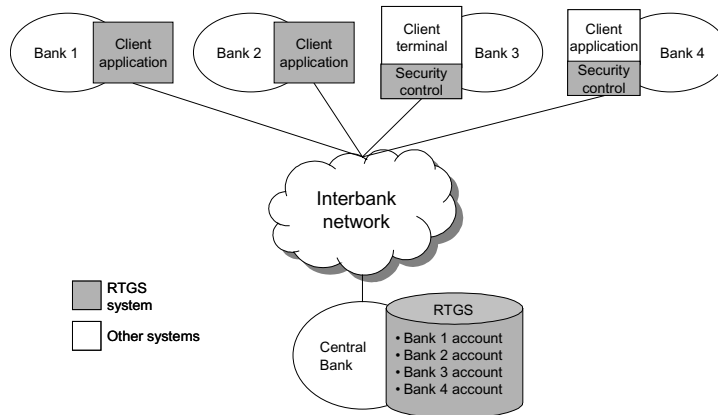


Figure 1. Centralized RTGS system

A distributed RTGS system is still centrally managed and controlled, but intraday liquidity management, payment processing and data storage are more scattered. One way to do this is to distribute the centralized database so that each bank has a "small RTGS database" available at their site with only bank's own account. System control and management is still centralized. Once liquidity has been transferred to banks' accounts, the client applications can perform settlement directly and finally between each other in real time using the account's liquidity. E-Settlement is a concept for such a distributed RTGS system [PAYMENTSYS]. The basic idea of E-Settlement is to distribute processing into standardized tamper-resistant elements called E-Settlement modules – the "small RTGS database" of the system. These modules use encrypted E-Settlement "stamps" to negotiate settlement directly with each other, without intervention by centralized system elements. Due to the high security requirements, the E-Settlement modules enforce a set of tight controls. Figure 2 below shows an E-Settlement system in a national context.

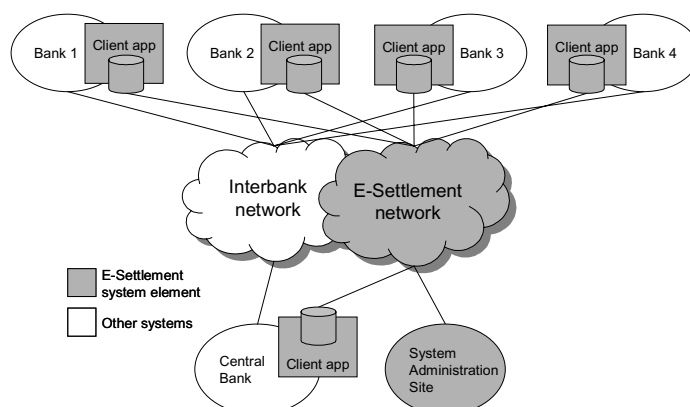


Figure 2. E-Settlement in a national context

The system consists of a number of elements interlinked with an E-Settlement network:

- **E-Settlement network** connects the E-Settlement system elements. It is used for system control and management traffic. The payment messages are transferred through the existing interbank networks. Physically these networks can be the same.
- **Bank E-Settlement module** is linked with bank's liquidity management and payment systems. It is able to process liquidity transfers to/from Central Bank E-Settlement module and perform payment settlement with other banks using encrypted E-Settlement stamps.
- **Central Bank E-Settlement module** is linked with CB accounting, liquidity issuance and payment systems. It can process liquidity transfers to/from banks and collect statistics from the banks.
- **System Administration Site** is in charge of all the centralized system features such as system-wide reconciliation, multilateral netting, directory management as well as overall system management and control.

E-Settlement can also be used in currency area context where multiple CBs co-operate to perform settlement in a single currency, such as in Euro area. Figure 3 below shows E-Settlement in currency area context. Banks can send payments directly to each other just as in a national E-Settlement system, also to banks controlled by other CBs. Each CB issues liquidity to banks under its control, performs CB level reconciliation and collects payment statistics of these banks. In this context, SAS has the additional task of collecting system level statistics and performing system-wide reconciliation. In this document we present the E-Settlement system in currency area context.

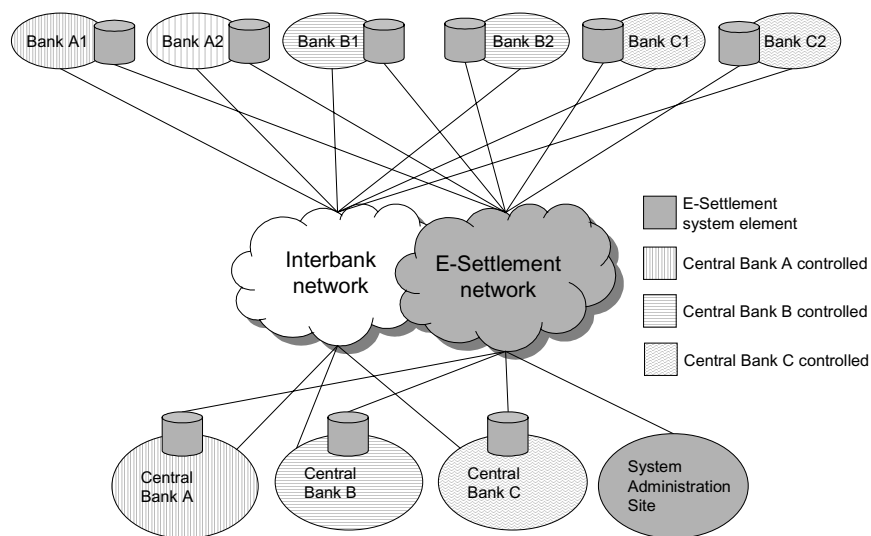


Figure 3. E-Settlement in currency area context

## 1.2 Ensuring information consistency in E-Settlement system

In designing distributed systems, a lot of attention has to be paid to ensuring that information remains consistent in the distributed elements. This is even more so in E-Settlement system, since the consequences of any inconsistencies can be severe. Liquidity information must be consistent in the system at all times. At any point, the total amount of liquidity in the E-Settlement modules must be equal to the amount of liquidity issued by the CBs. There are four levels of consistency checking.

### 1. Payment level

Consistency of information on payment level is accomplished through the use of E-Settlement stamps. The messaging mechanism between two E-Settlement modules provides all key facets

of secure communications: confidentiality, authenticity, integrity and non repudiability. Integrity of E-settlement stamps is accomplished through the use of a strong hash algorithm.

#### 2. Payment sequence level

A settlement stamp from E-Settlement module A to E-Settlement module B contains verification information: the order number of the message (from A to B), the amount of liquidity successfully transferred from A to B during the day and during the reconciliation period. In addition, the stamp contains similar information from the last few settlement transactions between the banks. Upon reception of a payment, E-Settlement module B compares the information sent by module A against the information it has in store.

#### 3. Reconciliation period level

During reconciliation breaks or when requested by CB, consistency of settlement traffic is checked between E-Settlement modules. The transactions and the amounts of liquidity sent between two E-Settlement modules have to match.

#### 4. Overall liquidity control level

At reconciliation breaks, and also otherwise when needed, consistency is checked at system level by requesting liquidity balance from banks' E-Settlement modules. This information is then compared to the amount of liquidity issued to the system by the CBs.

### 1.3 Ensuring consistency of audit trails

To ensure complete audit trails, the bank payment system must give its unique payment identifier and authenticated user credentials together with payment information to Bank E-Settlement module. The module stores this information and returns a unique E-Settlement payment identifier and an E-Settlement stamp. The bank payment system keeps a table of cross references between these.

### 1.4 To be specified later

At the time of this writing, E-Settlement is at a concept level with a prototype system being under specification. While a lot of effort has been put into ensuring that the concept is consistent and functionally complete, a number of issues can be considered in detail only after experiences from the prototype work have been gathered and analyzed. These issues include:

- Performance: performance of the system and any performance bottlenecks to be solved
- Availability: the detailed mechanisms needed to ensure production level availability
- System management: the practices needed in running and maintaining the system
- Exceptions: while many exceptional situations have been considered already at this stage, the full range of possible exceptions will become evident during the prototype work

### 1.5 Document scope

This document describes the E-Settlement system on a general level. Main topics covered are system structure and the core processes of the system. Related documents include [BAIF\_SPEC] describing banks' interface to the E-Settlement system in detail, [CBIF\_SPEC] describing the central banks' interface to the E-Settlement system in detail, and [TECH\_ARCH] defining technical aspects of E-Settlement system such as addressing, data stores and internal system interfaces.

## **1.6 Document structure**

First the design rationale behind the architecture specification work is presented. Then the system structure is presented, main processes are charted and exception situations presented. Issues related to system management and security are addressed.



## 2 DESIGN PHILOSOPHY

This chapter describes some of the principles adopted for system design and decisions made regarding the architecture.

### 2.1 Design rationale

**Open technologies will be utilized throughout the system** whenever possible. Proprietary (such as a service utilizing a proprietary interface provided by an operating system vendor) and off-the-shelf solutions used will be accessed through an interface hiding the proprietary and vendor-specific features. This will make it easier to replace these later on if the need arises.

**External interfaces will be specified in detail.** The responsibilities of the external systems are clearly defined. This will give the banks and the CBs the possibility to integrate with E-Settlement system as they see fit.

**Economics** will be a consideration in system design. The whole idea of the decentralized E-Settlement system relies on the relatively cheap price of “normal” server hardware as compared to the prices of hardware required in massive centralized processors. In addition, E-Settlement utilizes banks’ existing payment systems and interbank network connections.

**Security issues will be paid utmost attention.** It is seen that E-Settlement system concept will not be successful without due attention to security issues. Security must be “built in” the system from the very start of the development process. The approach chosen here is “security through openness” – system security controls must be strong enough to resist attempts at compromising the system by a party knowledgeable about the system’s security solution. The security management framework of E-Settlement system has been defined according to an ISO standard. See chapter 7 for more information.

The efficiency of the system relies on **decentralization and networking**. The basic operations are decentralized, and centralized control process should only require a limited part of the total processing. The 20/80 rule should be in force: at most 20 percent of the payment transactions, and at most 80 percent of the total value of payments, will be handled by centralized processing elements.

**E-Settlement is neutral towards any interbank payment network.** It is up to a bank’s payment system to decide whether to use E-Settlement to perform settlement of the payment, and in case it is used, what interbank payment network to use in transmitting the payment and E-Settlement information to the recipient’s bank. Naturally, the banks must mutually agree to settle payments using E-Settlement system beforehand, and the system must be aware of the agreement.

**An internal representation of payment information will be defined.** Settlement of payments is based on this information. Multilateral nettings also uses the internal representation of payments. To ease auditing and accounting, bank and E-Settlement payment representations are interlinked – a bank needs to give its unique payment ID to the E-Settlement module together with the payment information. Correspondingly, the E-Settlement system provides an unique E-Settlement payment ID to the bank systems.

**Liquidity operations are final in nature.** That is, liquidity is removed from the bank’s E-Settlement account immediately when an E-Settlement stamp is formed. If a transaction is rejected by the receiving bank, liquidity is then added back to the account. Payment cancellation at a later stage requires a transaction in another direction to transfer the liquidity back.

**E-Settlement is designed for real-time operation.** If the payment systems of both the sending and the receiving banks operate in real time, E-Settlement can provide final settlement of payments in real time. E-Settlement works without modification in non real-time environment as well.

## 2.2 Layered design

E-Settlement system is designed in a layered manner. The whole system relies on validity of a basic set of functionality, the E-Settlement core. After the core functionality has been proven to work, add-on services for both banks and central banks will be defined. See Figure 4 below for a schematic presentation of the approach.

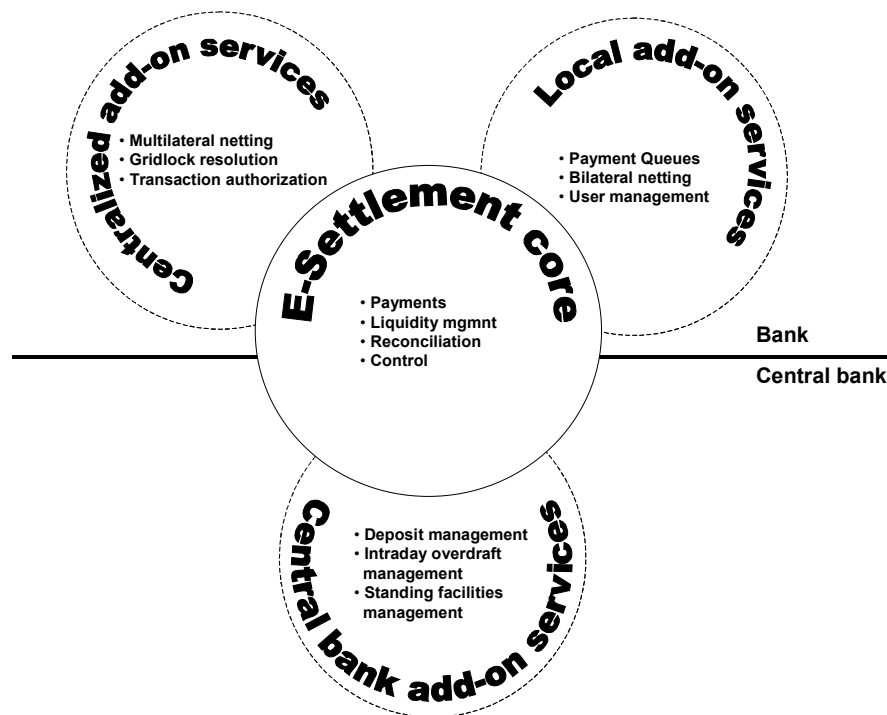


Figure 4. Layered structure of E-Settlement system

### 2.2.1 Core

Core functionality is needed for the E-Settlement system to work as a reliable settlement provider. Core processes, described in chapter 4, define the functionality. This functionality is provided solely by the E-Settlement system. This document concentrates on the core functionality.

Core functionality includes the following:

- Payment processing: Payments can be securely and reliably settled between the E-Settlement modules.
- Liquidity management
  - Liquidity is securely transferred in the system
  - Banks can manage their liquidity situation
  - Liquidity situation in the system is known in real time
- Reconciliation: Consistency of payment information is validated continuously based on E-Settlement stamps, and periodically on bank-to-bank, CB and system levels based on payment history.
- Control

- Security controls: security controls, as described in E-Settlement security management framework, are applied
- Beginning-of-day and end-of-day activities: E-Settlement modules can be opened and closed in a controlled and secure way
- Payment traffic control: deviations and problems in payment traffic are detected and can be reacted to
- System management: System problems are detected and can be reacted to

### 2.2.2 Local add-on services for banks

Local add-on services increase the service level and payment processing intelligence of the bank E-Settlement module, but do not directly contribute to payment settlement per se. These are needed by most banks to perform real-life payment settlement and will in the future be distributed as a part of the E-Settlement system.

- Payment queues: Managing payments for which at the moment there is not enough liquidity. Payment queue operates in prioritized “first in, first out” principle. The bank can reorder the queue and remove payments from it.
- Bilateral netting: bank-to-bank netting to save liquidity. The idea is that two banks that have frequent payment traffic with each other can have pending payments to each other at the same time. If this is the case, the payments can possibly be directly netted against each other, with only a small amount of liquidity (by the bank with smaller amount of payments to be netted) needed.
- User management: A service for managing bank’s E-Settlement users and their access rights to the E-Settlement system. Banks handle their user management towards the E-Settlement system using this service.

### 2.2.3 Centralized add-on services for banks

These services increase the service level of the E-Settlement system. From banks’ perspective, Bank E-Settlement module acts as an “access point” to these services.

- Multilateral netting: based on payments and liquidity sent to a centralized netting component. Netting is performed according to a schedule, i.e. netting is performed at specific times of the day. Banks can be given an opportunity to send more liquidity in case the original liquidity is not sufficient for the netting.
- Gridlock resolution: also based on payments and liquidity sent to a centralized netting component. In theory, every time additional liquidity or a new payment is sent to the service, or bank’s netting queue is modified, a new netting could be attempted. Out of efficiency concerns it is probably good to limit the resolution interval e.g. to every 15 minutes and reserve the service for large value payments only.
- Transaction authorization: centralized authorization service for very large value payments. All payments beyond a certain value, and possibly a small randomly selected portion of all payments, are checked and authorized by the centralized service. In addition, authorization procedures could be invoked when the payment traffic turnover reaches given predefined reporting points.

#### 2.2.4 Central bank add-on services

The basic central bank services needed in the E-Settlement system are liquidity transfers between E-Settlement modules. In the basic situation all balances in the Bank E-Settlement modules are positive and Central Bank E-Settlement modules have the opposite negative balances, reflecting the amount of liquidity issued to the banks. Any payment transaction from the central bank to the bank and any additional liquidity issue during the day will increase the liquidity balance in the Bank E-Settlement modules and at the same time increase the negative balance in the Central Bank E-Settlement modules.

However, central banks will need some additional features for liquidity management. These can be provided as add-on services. Generally the liquidity issued to the banks for payment system purposes consists of reserve or other deposits and of intraday overdraft facilities. If a bank at the end of the day has an exceptional and unplanned liquidity situation, the bank may need to revert to standing facilities. Hence the central bank add-on services could consist of

- reserved deposit management systems
- other deposit managements systems
- intraday overdraft management systems
- standing facilities management systems

The add-on services would typically provide an automated interface between these liquidity providing systems and even the accounting and management system for these deposits and credits.

### 3 STRUCTURE OF E-SETTLEMENT SYSTEM

This chapter maps the structure of the E-Settlement system. First, the elements in the system and the external systems are defined. The responsibilities of each of the elements in the E-Settlement core processes (payment, liquidity management, reconciliation and control) are then charted. E-Settlement system must provide the means for the elements to meet these responsibilities. The responsibilities act as high-level functional requirements for the E-Settlement system. Lastly, the interfaces between the system elements are presented shortly.

E-Settlement system consists elements interlinked with a network, shown in gray in Figure 5 below. Two kinds of external systems are linked with the E-Settlement system, and the external systems are linked with another network. These are jointly called “external elements” in this specification. The figure also shows the abbreviations for the elements and the active system elements.

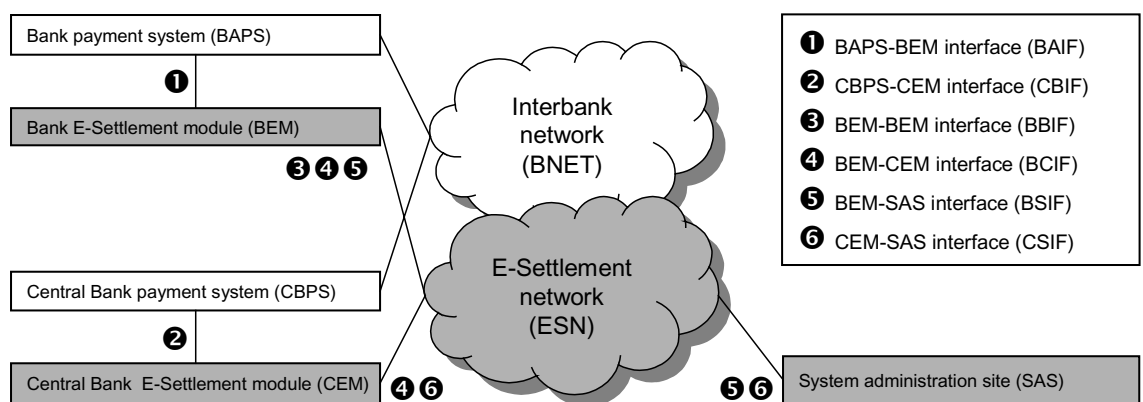


Figure 5. E-Settlement system elements and interfaces

E-Settlement system elements are defined as follows:

- Bank E-Settlement module (BEM): A secure processing element responsible for performing settlement of payments in real time
- Central Bank E-Settlement module (CEM): A secure processing element responsible for controlling a number of BEMs.
- System administration site (SAS): Centralized element responsible for overall system control
- E-Settlement network (ESN): The secure, dedicated network connecting the E-Settlement system elements

The external elements are defined as follows:

- Bank payment system (BAPS): The existing payment system of the bank enhanced with an interface to the E-Settlement system
- Central Bank payment system (CBPS): The existing payment system of the CB enhanced with an interface to the E-Settlement system
- Interbank network (BNET): The existing network(s) connecting the payment systems of banks and CBs, such as SWIFT and SWIFTNet in an international setting, or Pankkiverkko 2 in Finland.

E-Settlement system has two external interfaces, one towards the banks and one towards central banks. Banks and CBs access the services provided by the E-Settlement system through these interfaces. In addition, there are four internal interfaces between the system elements.

## 3.1 Bank payment system BAPS

From E-Settlement system point of view, a bank is an entity that sends and receives payments, the settlement of which is handled via E-Settlement stamping. In addition to the responsibilities listed here, bank payment system has the security responsibilities set forth in [SECURITY\_POLICY].

### 3.1.1 Payment

In this process the bank payment system has the following responsibilities in the E-Settlement system:

- Payment message transfer via interbank networks
- Maintaining payment queues (basic payment queues will be provided as an add-on service, too)
- Outgoing credit transfer payments:
  - Requesting E-Settlement stamps from BEM. This entails giving the core payment information, and the bank's unique payment identifier to BEM and receiving the E-Settlement payment ID and E-Settlement stamp in return.
  - Storing a table of cross references between bank's unique payment identifiers and E-Settlement payment IDs
  - Attaching E-Settlement stamps received from BEM to interbank payments and sending the payments to beneficiary's bank as soon as possible
  - Confirming the success or failure of each E-Settlement stamped payment to BEM based on the acknowledgement sent by the beneficiary's bank as soon as possible when such an acknowledgement has been received
  - Listening to and reacting to notifications from BEM regarding payments for which BEM has not yet received a (positive or negative) confirmation
- Incoming credit transfer payments:
  - De-loading the E-Settlement stamps from the interbank payments
  - Sending all received stamps to BEM for validation even if the payment was rejected by BAPS, and receiving the confirmation stamps from BEM
  - Sending an E-Settlement stamped confirmation message to the sending bank for each received E-Settlement stamped interbank payment

### 3.1.2 Liquidity management

Bank payment system is responsible for following up and ensuring availability of sufficient liquidity for smooth processing of payments, more specifically:

- Following up BEM liquidity situation, responding to low-on-liquidity notifications sent by BEM
- Requesting for more liquidity from CB if required
- De-loading unnecessary liquidity to CB when seen feasible

### 3.1.3 Reconciliation

In this process BAPS is responsible of the following:

- Reacting to reconciliation related notifications sent by BEM
- Ensuring that there are no inconsistencies between BEM transaction history and the payment history stored by the bank
- Reading and storing BEM statements of account

### 3.1.4 Control

- Storing BEM transaction logs for legal and accounting purposes

## 3.2 Central Bank payment system CBPS

Central Bank payment system issues liquidity to the to banks and controls banks' payment traffic. In addition to the responsibilities listed here, Central Bank payment system has the security responsibilities set forth in [SECURITY\_POLICY]. Note that from technical E-Settlement system point of view, ECB payment system site is identical with other CB sites.

### 3.2.1 Payment

Central Bank payment system is responsible for sending liquidity to banks as normal interbank payments, the recipient of which is the E-Settlement module of the bank in question. Thus, its responsibilities in this respect are similar to that of the Bank payment system. The main differences stem from the fact that CB payment has limitless liquidity. In payment processing its responsibilities include:

- Payment message transfer via interbank networks
- Outgoing credit transfer payments:
  - Requesting E-Settlement stamps from CEM. This entails giving the core payment information and the CB's unique payment identifier to CEM and receiving the E-Settlement payment ID and E-Settlement stamp in return.
  - Storing a table of cross references between bank's unique payment identifiers and E-Settlement payment IDs
  - Attaching E-Settlement stamps to interbank payments
  - Attaching E-Settlement stamps received from CEM to interbank payments and sending the payments to beneficiary's bank as soon as possible when an E-Settlement stamp has been received from CEM
  - Confirming the success or failure of each E-Settlement stamped payment to CEM based on the acknowledgement sent by the beneficiary's bank as soon as possible when such an acknowledgement has been received
  - Listening to and reacting to notifications from CEM regarding payments for which CEM has not yet received a (positive or negative) confirmation
- Incoming credit transfer payments:
  - De-loading the E-Settlement stamps from the interbank payments

- Sending all received stamps to CEM for validation even if the payment was rejected by CBPS, and receiving the confirmation stamps from CEM
- Sending an E-Settlement stamped confirmation message to the sending bank for each received E-Settlement stamped interbank payment

### 3.2.2 Liquidity management

This functionality has been planned as an add-on service to the central banks.

Central Bank payment system has the following liquidity management responsibilities:

- Keeping up to date on the amount of liquidity issued to each bank
- Issuing more liquidity to banks upon request after validating the request against the funds available to the bank
- Receiving unused liquidity from banks during the day and at the end of the day
- Ordering forced liquidity de-issuing as required
- Replying to liquidity situation requests from the banks

### 3.2.3 Reconciliation

In this process CBPS is responsible for the following:

- Reacting to reconciliation related notifications sent by CEM
- Ensuring that the transaction log of Central Bank E-Settlement module is consistent with CBPS liquidity accounting information
- Reading and storing CEM statements of account

### 3.2.4 Control

In this process CBPS is responsible for the following:

- Storing full CEM transaction logs
- Reacting to bank profile violations

## 3.3 Bank E-Settlement module BEM

BEM is a secure, decentralized processing element capable of forming and validating E-Settlement stamps, controlling the consistency of payment traffic, storing and providing access to payment transaction log, performing continuous and bank-to-bank reconciliation of payment traffic, calculating payment statistics and supporting the liquidity management needs of banks.

### 3.3.1 Payment

In this process BEM is responsible for the following:

- maintaining BEM liquidity balance information



- forming and validating E-Settlement stamps
- alerting SAS if E-Settlement stamp is invalid or there are any other stamp problems
- Ensuring that the payment profiles (limits on the number and amount of payments that can be processed per unit of time) of the banks are enforced

### **3.3.2 Liquidity management**

In this process BEM is responsible for the following:

- Reporting liquidity situation to BAPS, CEM and SAS upon request
- Reporting funds available to the BAPS upon request
- Sending a notification to BAPS if available liquidity goes below a pre-configured limit

### **3.3.3 Reconciliation**

In this process BEM is responsible for the following:

- Maintaining bilateral (bank-to-bank) transaction history for audit trail checking and reconciliation purposes
- Performing continuous reconciliation
- Performing bank-to-bank reconciliation upon request and automatically at the end of reconciliation periods

### **3.3.4 Control**

In this process BEM is responsible for the following:

- Allowing start-up of BEM only after strong and secure user authentication
- Storing a local copy of the system directory data as indicated by SAS and the CEM controlling the BEM
- Restoring system state from persistent storage after system failure
- Informing BAPS, CEM and SAS about changes in system state
- Giving periodical activity signals to controlling CEM and to SAS
- Stop (and resume) sending payments to a particular BEM if ordered so by CEM or SAS
- Stop (and resume) sending payments altogether if ordered so by CEM or SAS
- Checking that all payments have been either rejected or confirmed by BAPS. Alert BAPS and SAS on expiration of payment timers.
- Calculating payment, IT resource usage, availability, payment performance and activity statistics of the bank
- Providing access to BEM transaction log

- Monitoring possible deviations from normal payment flow (too many unconfirmed payments, prolonged unavailability of liquidity) and alerting CEM/SAS

### **3.4 Central Bank E-Settlement module CEM**

CEM is a secure processing element capable of forming and validating E-Settlement stamps, controlling the consistency of payment traffic, and for controlling the liquidity situation in the banks under its control. The Central Bank E-Settlement module also stores the full CB payment logs, calculates payment statistics for the CB area, and takes some system management responsibility for the banks under its control.

#### **3.4.1 Payment**

In payment processing CEM is responsible for the following:

- maintaining CEM liquidity balance information
- forming and validating E-Settlement stamps
- alerting SAS on invalid E-Settlement stamps or any other stamp problems

#### **3.4.2 Liquidity management**

In this process CEM is responsible for the following:

- Following up the amount of liquidity issued to each BEM
- Informing SAS on any changes of the liquidity issued to, sent to or received from the BEMs
- Forwarding BEM liquidity requests to CBPS and forwarding CBPS replies to BEM

#### **3.4.3 Reconciliation**

In this process CEM is responsible for the following:

- Maintaining bilateral (bank-to-bank) transaction history for audit trail checking and reconciliation purposes
- Performing continuous reconciliation
- Performing bank-to-bank reconciliation upon request and automatically at the end of reconciliation periods

#### **3.4.4 Control**

In this process CEM is responsible for the following:

- Allowing start-up of CEM only after strong and secure user authentication
- Storing a local copy of the system directory data as indicated by SAS
- Restoring system state from persistent storage after system failure
- Informing CBPS and SAS about changes in system state

- Giving periodical activity signals to SAS
- Stop (and resume) sending payments to a particular BEM or CEM if ordered so by SAS
- Stop (and resume) sending payments altogether if ordered so by SAS
- Checking that all payments have been either rejected or confirmed by CBPS. Alert CBPS and SAS on expiration of payment timers.
- Calculating payment, IT resource usage, availability, payment performance and activity statistics of CEM
- Collecting BEM payment statistics from BEMs under the CEMs control after reconciliation breaks and at the end of the day.
- Cross-tabulating the payment statistics for the CB after reconciliation periods and at the end of the day
- Providing CB statistics to SAS upon request
- Providing access to CEM transaction log
- Granting E-Settlement smart cards (used for storing critical bank-specific information) to banks, reacting to any alarms sent by BEMs
- Maintaining BEM and BAPS information (identifications, contact information, PKI, bank profiles) for the BEMs and BAPSEs under CEM's control
- Starting and stopping payment processing of BEMs under the CEMs control during beginning-of-day and end-of-day activities, respectively, and also during the day if required

## **3.5 System administration site SAS**

System administration site is the centralized administration and management entity in the E-Settlement system. ECB could be a natural environment for SAS.

### **3.5.1 Payment**

In this process SAS has the following responsibilities:

- Ordering BEMs to stop (and resume) sending payments to specified BEMs if there are problems in payment traffic between BEMs
- Ordering BEMs to stop and start processing payments in case of serious payment problems

### **3.5.2 Liquidity management**

In this process SAS has the following responsibilities:

- Keeping up to date on the amount of liquidity currently being issued in the E-Settlement system

### **3.5.3 Reconciliation**

In this process SAS has the following responsibilities:

- Correcting bank-to-bank reconciliation problems
- Performing system level reconciliation, i.e. ensuring that the amount of liquidity in circulation, as reported by CEMs, corresponds to the amount of liquidity reported by the BEMs. This is to be done according to a predefined schedule, and at will during the day.
- Defining the reconciliation schedule for the day and writing this information to BEMs during beginning-of-day activities

### 3.5.4 Control

System administration site is responsible for general system management tasks, such as:

- Managing system security: keys, certificates and security policy
- Maintaining CEM and CBPS information (identifications, contact information, PKI) for the CEMs and CBPSs
- Maintaining system directory information
  - private CEM/BEM information
    - bank profile
    - bank-specific timeouts
    - controlling element (CEM or SAS)
  - public CEM/BEM information
    - CB code of the bank
    - Bank name
    - BIC(s)
    - IP address
    - public key
    - the state of the bank (whether payments should be sent to the bank)
  - software version information
  - warning and alarm limits
- Responding to any alarms sent by the system elements
- Providing BEMs with system data during beginning-of-day activities
- Collecting and storing IT resource usage, activity, availability and payment performance statistics from BEMs
- Managing software versions in CEM and BEM
- Collecting statistical information from E-Settlement modules and calculating system level statistics based on these.

## 3.6 E-Settlement system interfaces

What follows is a very brief description of the system interfaces.

### 3.6.1 Bank interface BAIF

Bank interface offers the E-Settlement functionality to banks. It needs to be flexible enough for many kinds of banks to use, and thus needs to be carefully designed. The interface is specified in [BAIF\_SPEC] and offers the following functionality:

- E-Settlement stamp request and validation
- Liquidity management
- Reconciliation between BAPS and BEM
- User management
- Access to system directory data
- Notifications mechanism

### 3.6.2 Central Bank interface CBIF

Central bank interface offers the E-Settlement functionality to CBs. The interface is an extension of the Bank interface. It has been augmented with commands related to liquidity issuing/de-issuing and bank control functions. Flexibility is also crucial for CBs. The interface is specified in [CBIF\_SPEC].

### 3.6.3 BEM-BEM interface BBIF

BBIF contains bank-to-bank reconciliation functionality and other functions needed in controlling settlement traffic between two banks. The interface is specified in [TECH\_ARCH].

### 3.6.4 BEM-CEM interface BCIF

BCIF provides functionality for sending alerts between BEM and CEM, transferring transaction log, statistics and BEM account data between BEM and CEM, and for controlling BEM. [TECH\_ARCH]

### 3.6.5 BEM-SAS interface BSIF

BSIF provides functionality for authentication, sending alerts between BEM and SAS, controlling BEM, and transferring following data between BEM and SAS:

- transaction logs on request
- system directory data
- reconciliation data
- liquidity balance information
- activity information
- availability information

- payment performance information
- IT resource usage information
- software version control information

The interface is specified in [TECH\_ARCH].

### **3.6.6 CEM-SAS interface CSIF**

CSIF provides functionality for controlling CEM, sending alerts between CEM and SAS, and for transferring data between CEM and SAS similar to that transferred between BEM and SAS. In addition, issued liquidity information is transferred over this interface. The interface is specified [TECH\_ARCH].

## 4 CORE PROCESSES

This chapter presents an overview of the E-Settlement core processes. Figure 6 describes how the core processes are interlinked. It presents a typical (unproblematic) “day in life” of a Bank E-Settlement module. The day of BEM starts with BEM start-up process, after which beginning-of-day activities (reception of system data from SAS) are performed. Initial liquidity loading follows. After this, liquidity management in the form for liquidity issuing and de-issuing processes goes on through the day. An *E-Settlement day* is divided into a number (minimum one) of *reconciliation periods*, during which BEM processes payments normally and performs continuous reconciliation. After the period has ended, bank-to-bank reconciliation is performed to ensure that payment information is consistent throughout the distributed system. After the last reconciliation period of the day has ended, liquidity is de-loaded to CEM and end-of-day activities are performed. EOD activities include de-issuing remaining liquidity, and calculation and collection of system statistics.

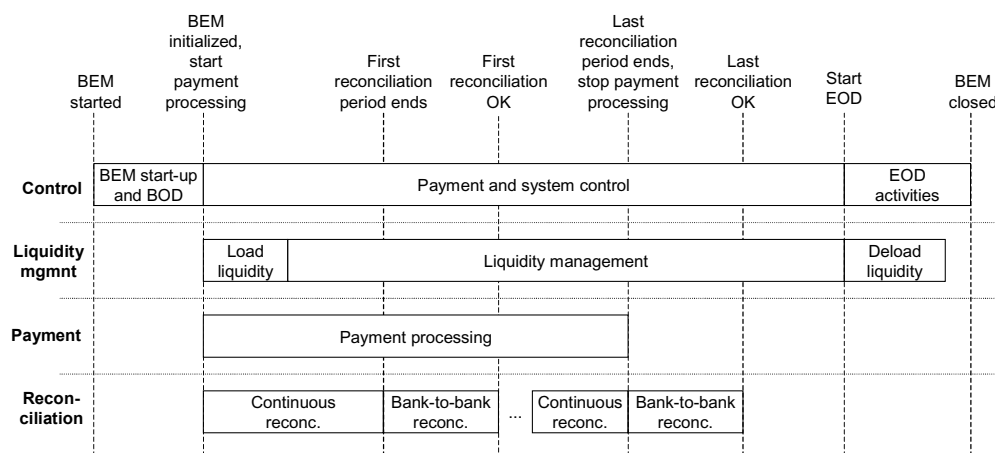


Figure 6. "Day in life" of BEM

The processes in this chapter are presented using a sequence diagram notation presented in Figure 7 below. In the notation, each system element is described as a gray box at the top of the picture, with each element having a descriptive name and type separated with a colon, such as “Sending:BEM”. Messages between the elements are represented using numbered and labeled arrows, with conditional messages shown as dotted lines. An arrow starting and ending in the same element is used to represent processing or a decision by the element. The flow of time in the pictures is top-down.

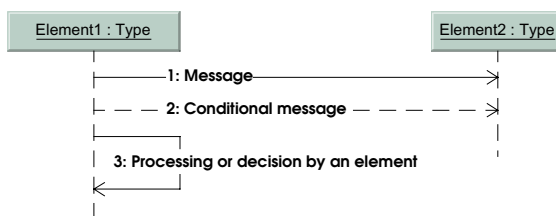


Figure 7. Process notation

In order to make it easier to see the responsibilities of each system element in the processes, all process descriptions have the same system elements in the same order: sending BEM, sending bank’s payment system, receiving bank’s payment system, receiving BEM, (sending bank’s) CEM, (sending bank’s) Central Bank payment system and finally SAS. This is done even if a system element does not take part in the messaging of a particular process – for example, in liquidity issuing the receiving bank (in the payment process) does not have any role. It is still in process description to emphasize that it does not have any role, or responsibility, in the process. Banks are always referred to as “Sending” and “Receiving” even if no message transfer takes place between the banks. “Sending” bank is the active

one starting the transactions. Core processes have been separated into four groups as described earlier: Payment, Liquidity management, Reconciliation and Control processes.

## 4.1 Payment processes

In the payment process two banks exchange payment and settlement information to process an inter-bank payment. This chapter presents the different payment processes, including rejections at various stages and payment time-outs. See also [BAIF\_SPEC] for more information.

In sending BEM, a payment can be in one of three states. It is important to note that all changes in payment state are final in nature, i.e. that liquidity is finally increased or decreased at each change of state – except when a payment is positively acknowledged, in which case no change in liquidity takes place. The three states are:

1. Final unconfirmed credit transfer (upon stamp creation)
2. Final confirmed credit transfer (upon reception of positive confirmation)
3. Final rejected credit transfer (upon reception of negative confirmation)

In receiving BEM, a payment can be in exactly two states. Liquidity is increased if an E-Settlement stamp can be successfully validated.

1. Final confirmed credit transfer (upon validation of E-Settlement stamp)
2. Final rejected credit transfer (if payment was rejected by the receiving BAPS or receiving BEM)

As an implication of this finality of operations, canceling a payment for which an E-Settlement stamp has been created requires processing the payment normally followed by a payment in the other direction at the same amount – see payment cancellation process.

At each stage of payment processing after the E-Settlement stamp has been created, the stamp can be given to next participant in the payment process either as accepted or rejected. In all cases, the stamp goes through the whole payment process – even if the payment was rejected at some earlier stage::

1. Sending BAPS normally sends a payment and the stamp to Receiving BAPS as accepted but can also send it as rejected if the payment has been immediately canceled. Note that even in this case the payment and the stamp are sent to the Receiving BAPS.
2. If Receiving BAPS can validate the payment, it forwards the stamp to Receiving BEM as accepted, and if it cannot validate the payment (e.g. beneficiary not found) it forwards the stamp to the receiving BEM as rejected.
3. If Receiving BEM can validate the stamp and can perform continuous reconciliation successfully, it accepts the payment and performs final settlement of the payment. In the opposite case the stamp is rejected. A new E-Settlement stamp (“return stamp”) containing the acceptance or rejection is returned to Receiving BAPS. From E-Settlement system point of view, this is the “final judgement” for the payment, since the return stamp cannot be modified by BAPSes. Thus, it will be transferred back to Sending BEM without modification.
4. If Receiving BAPS can complete the payment transaction, it sends the payment and the return stamp to Sending BAPS as accepted, and if it cannot complete the transaction, they are returned as rejected. In the latter case the Receiving BAPS has to resort to payment cancellation process, since liquidity has already been finally transferred.



5. Sending BAPS sends the return stamp to Sending BEM as a confirmation of the settlement transaction. If Sending BAPS can complete the transaction, the payment transaction has been completed successfully. If it cannot do so, it has to resort to payment cancellation process, since liquidity has already been finally transferred.

#### 4.1.1 Normal payment flow

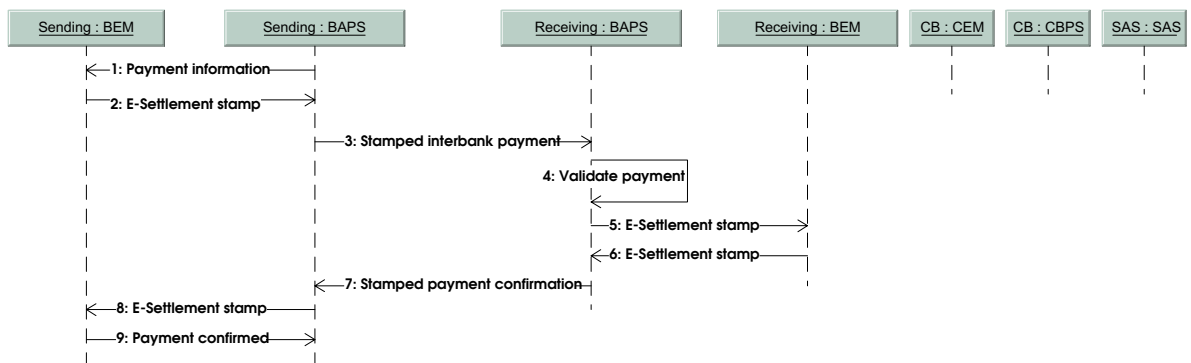


Figure 8. Normal payment flow

- 1: Sending BAPS presents payment information to its BEM.
- 2: Sending BEM validates the payment message, reduces liquidity from the bank's liquidity balance and generates an E-Settlement stamp, which it returns to BAPS.
- 3: Sending BAPS attaches the E-Settlement stamp to the payment message, and sends it immediately to the Receiving BAPS via BNET.
- 4: Receiving BAPS gets the message and validates the payment message content.
- 5: Receiving BAPS checks that the payment can be completed in its payment system and presents the E-Settlement stamp to its BEM.
- 6: The Receiving BEM validates the E-Settlement stamp, adds the payment amount to the bank liquidity and returns a positive acknowledgement to Receiving BAPS, together with a new E-Settlement stamp containing the Receiving BEM confirmation. Receiving BAPS can commit the payment at this stage.
- 7: Receiving BAPS sends Sending BAPS an E-Settlement stamped positive acknowledgement message to inform the payment transaction has been completed successfully.
- 8: Sending BAPS receives the message, detaches the E-Settlement stamp in it and presents the stamp to its BEM informing that the payment message was processed successfully.
- 9: Sending BEM acknowledges that the operation was concluded successfully. Sending BAPS can now consider the payment and settlement complete.

### 4.1.2 Payment cancellation

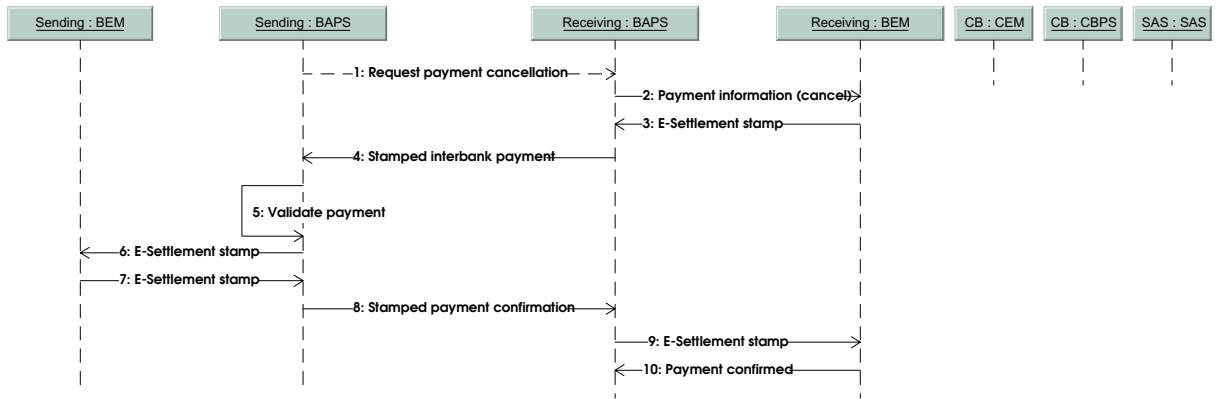


Figure 9. Payment cancellation

From E-Settlement system point of view, payment cancellation is a process for transferring liquidity from Receiving BEM to Sending BEM. Only a confirmed payment can be canceled.

1: The cancellation process can either be started by the Sending BAPS or the Receiving BAPS, depending on the situation as described in the beginning of chapter 4.1. In both cases, a cross-reference to the cancelled payment is sent in the request.

2: Receiving BAPS presents the cancelled payment information to its BEM.

3: Receiving BEM forms a new E-Settlement stamp containing a cross-reference to the canceled E-Settlement transaction.

4-10: The payment is processed normally as a payment from Receiving bank to Sending bank. As end result, the liquidity has been transferred back from the Receiving bank to Sending bank.

### 4.1.3 Payment resending

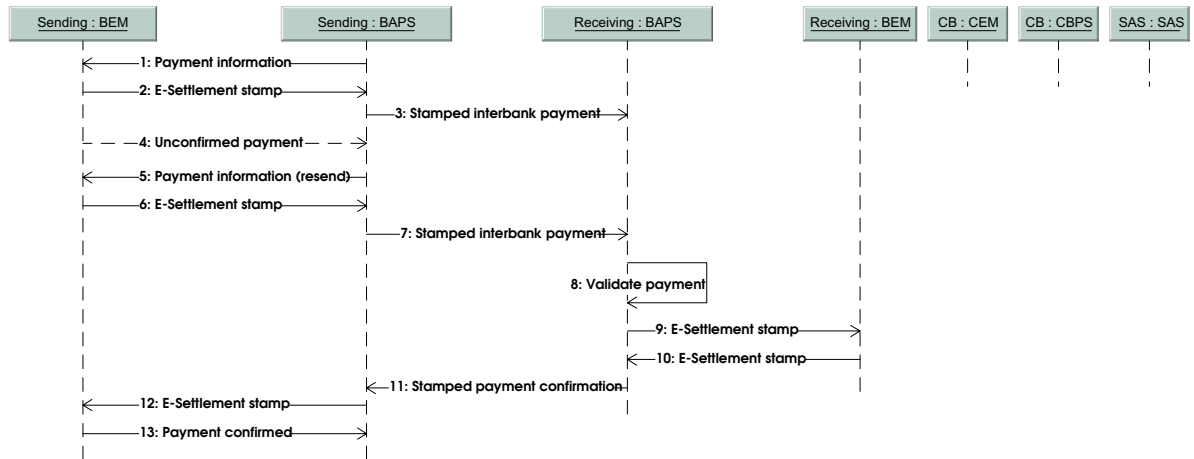


Figure 10. Payment resending

Payment resending is process for trying to receive a confirmation for hitherto unconfirmed payments.

1-3: Payment transaction is begun normally.

4: A notification from Sending BEM can start payment resending. Independently of this, Sending BAPS can also keep timers on payments and try to resend payments in order to receive a confirmation for them.

5: Sending BAPS requests resending of a payment by presenting payment resending information to its BEM. This resets the payment timer in Sending BEM. A new stamp is created but no additional liquidity is used for the stamp.

6-13: Normal payment process with the resent payment information is conducted. All elements must be prepared to receive duplicate payment information and take care that the payment will not be processed twice.

#### 4.1.4 Bank profile violation

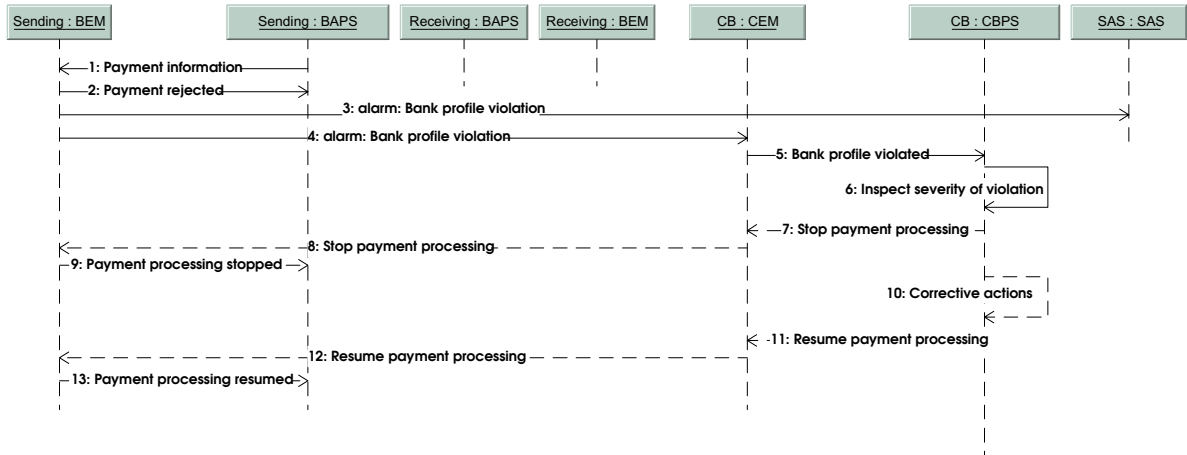


Figure 11. Bank profile violation

1: Sending BAPS gives payment information to Sending BEM.

2: Sending BEM concludes that bank profile would be violated (e.g. too large-value payment) if the payment was processed. The payment is rejected and the cause is returned to Sending BAPS.

3-4: Sending BEM sends an alert to both CEM and SAS about the situation.

5-6: CEM informs the violation to CBPS. The decisions and corrective actions in this respect are taken by CBPS.

7-9: In this case, CEM suspects misconduct by Sending BAPS and orders Sending BEM to stop processing payments. Sending BEM informs its BAPS of this.

10: CBPS performs corrective actions, such as modifies the bank profile. In this case, system directory would be updated on BEM's part and BEM would be informed of the change.

11-12: Problem was solved. CBPS tells CEM to order Sending BEM to resume payment processing, which BEM informs to BAPS.

### 4.1.5 Receiving BAPS rejection

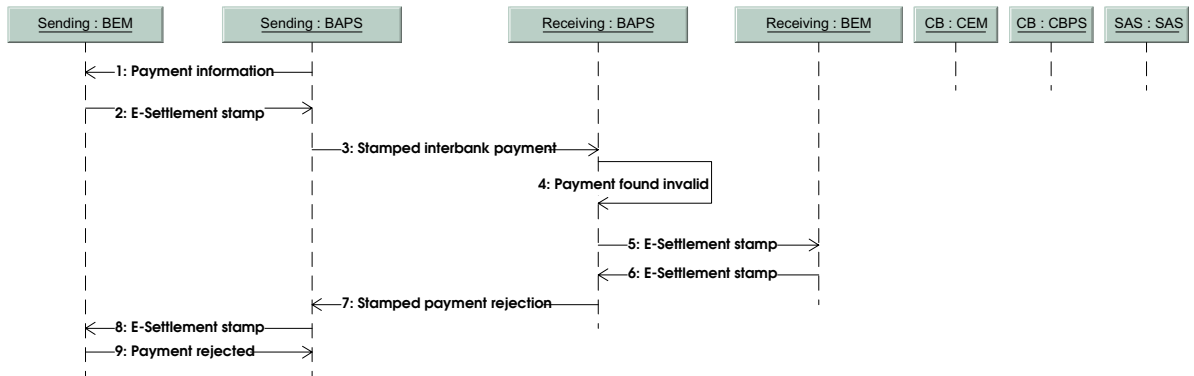


Figure 12. Receiving BAPS rejects the payment.

1–3: Normal payment process.

4: Receiving BAPS rejects the payment. There can be many reasons for this, for example unknown beneficiary IBAN number, or the payment has been sent to wrong BAPS or BEM.

5: This is a different call in the interface, telling BEM that the payment corresponding to the stamp was rejected by the bank. In this way Receiving BEM can update its payment info and keep the E-Settlement audit trail intact.

6: Receiving BEM returns an E-Settlement stamp indicating rejection by Receiving BAPS.

7: Receiving BAPS sends Sending BAPS an E-Settlement stamped negative acknowledgement message to inform the payment was rejected.

8: Sending BAPS receives the message, detaches the E-Settlement stamp in it and presents the stamp to BEM informing that the payment message was rejected.

9: Sending BEM acknowledges that the payment was rejected. Sending BAPS can now proceed with the rejection in its internal statistics.

#### 4.1.6 Receiving BEM rejection

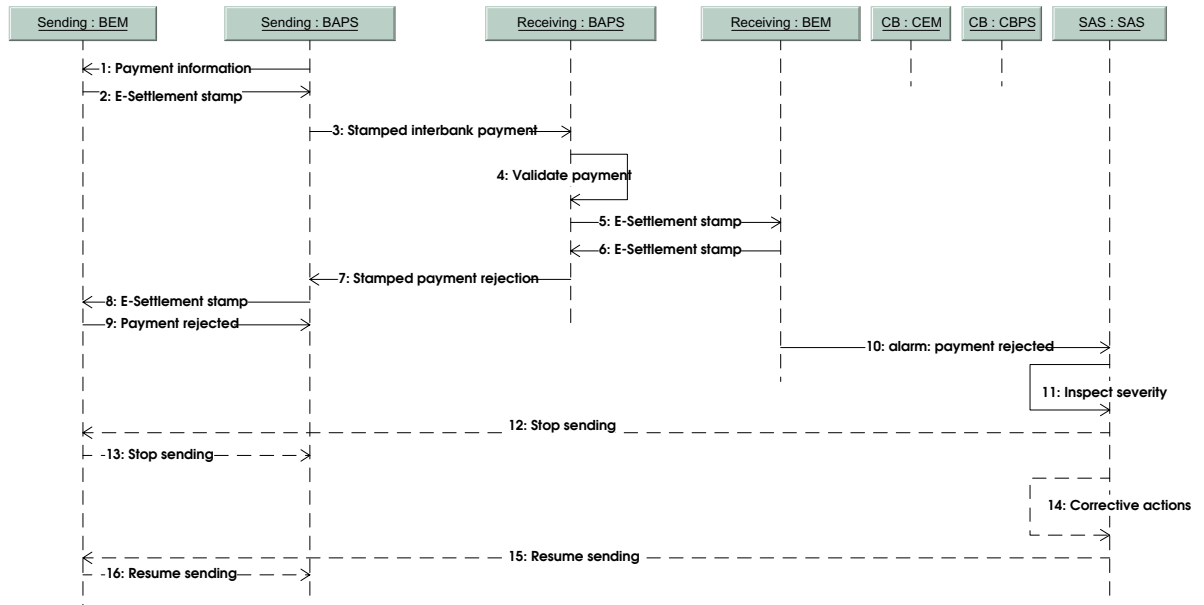


Figure 13. Receiving BEM rejects payment.

1–4: Normal payment process.

5–9: For some reason (e.g. invalid payment stamp) BEM rejects the E-Settlement stamp. The payment is handled as a rejected payment back to Sending BEM. After stage 5, the liquidity balance of the Receiving BEM is *not* changed. Upon reception of a confirmation stamp implying rejection of settlement by the Receiving BEM, the liquidity balance of the Sending BEM is increased for the amount of the transaction to correct the balance.

10: Sending BEM informs SAS of the cause of the rejection.

11: SAS studies the cause and decides on action to take, if any.

12–16: SAS orders the Sending BEM not send any payments to the Receiving BEM. It inspects the situation more carefully and tries to correct it, and then orders Sending BEM to resume sending to Receiving BEM.

The corrective actions that SAS takes depend on the cause of the rejection. These will become more clear during later phases of E-Settlement work.

### 4.1.7 Time-out during payment processing

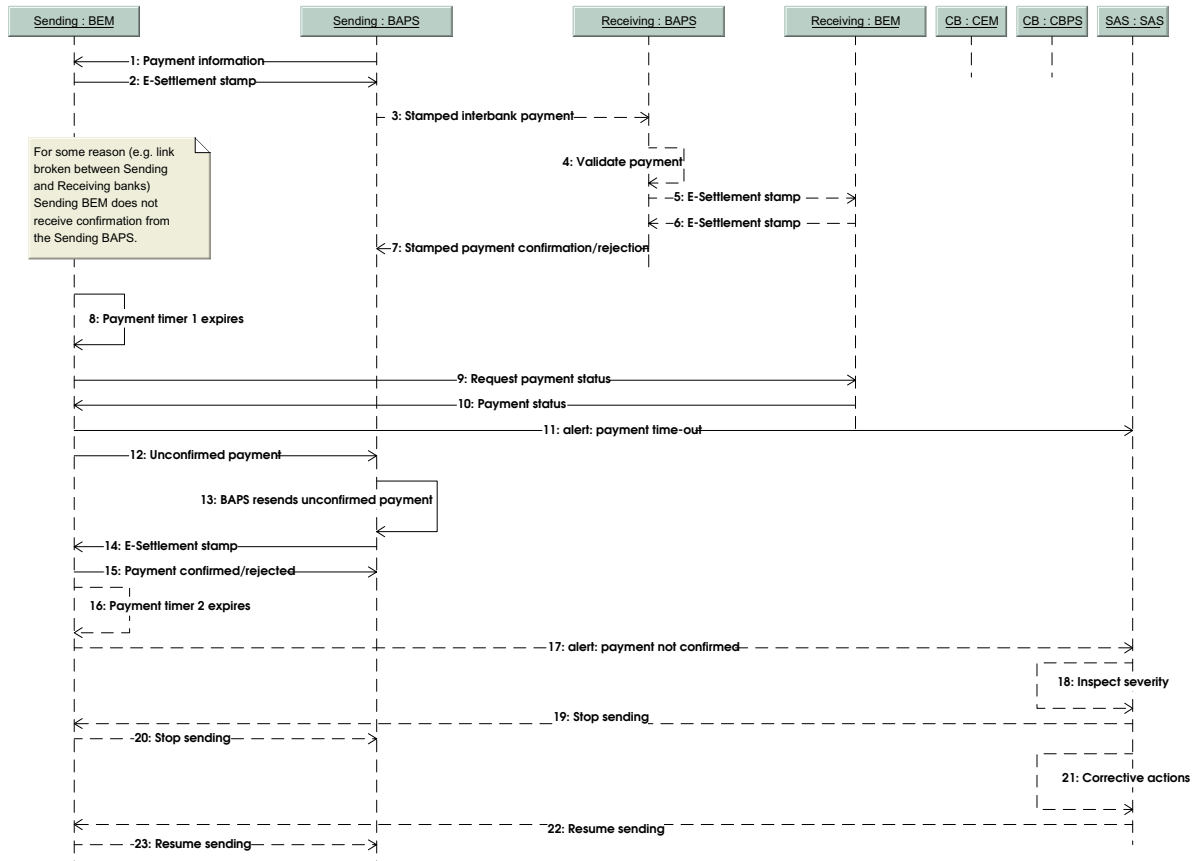


Figure 14. Payment process time-out.

A two stage "no confirmation" process is presented here. First sending BAPS has some time to get the missing confirmations. The problem is escalated to SAS if it persists.

1–2: Normal payment process is started and E-Settlement stamp is created.

3–7: Sending BEM does not receive the confirmation to the payment message. The possible reasons for this include the following:

- There is failure in Sending BAPS, resulting in the payment and stamp never being sent to the Receiving BAPS, or the return stamp is erroneously never presented to the Sending BEM
- Communication failure between Sending and Receiving BAPS either at stage 3 or stage 7
- There is failure in Receiving BAPS, resulting in the E-Settlement stamp not being sent to Receiving BEM for validation, or the return stamp is not sent back to the Sending BEM

8: Sending BEM does not get a confirmation. Payment timer 1 expires.

9–12: Sending BEM requests the status of the pending payment from the Receiving BEM. Receiving BEM returns its status (not received, confirmed or rejected). In this case, the Receiving BEM has not received the payment, and Sending BEM alerts SAS and its BAPS of this.

13–15: Sending BAPS resends the payment (see 4.1.3). To its BEM the BAPS can only present the return stamp formed by the Receiving BEM.

16–17: Confirmation was not received. Payment timer 2 expires, the problem is escalated to SAS.

18-20: SAS inspects the situation. It orders Sending BEM to stop sending to Receiving BEM.

21: SAS performs corrective actions depending on the situation. These can include manual actions.

22-23: SAS orders Sending BEM to resume sending to Receiving BEM.

#### 4.1.8 Payment missing

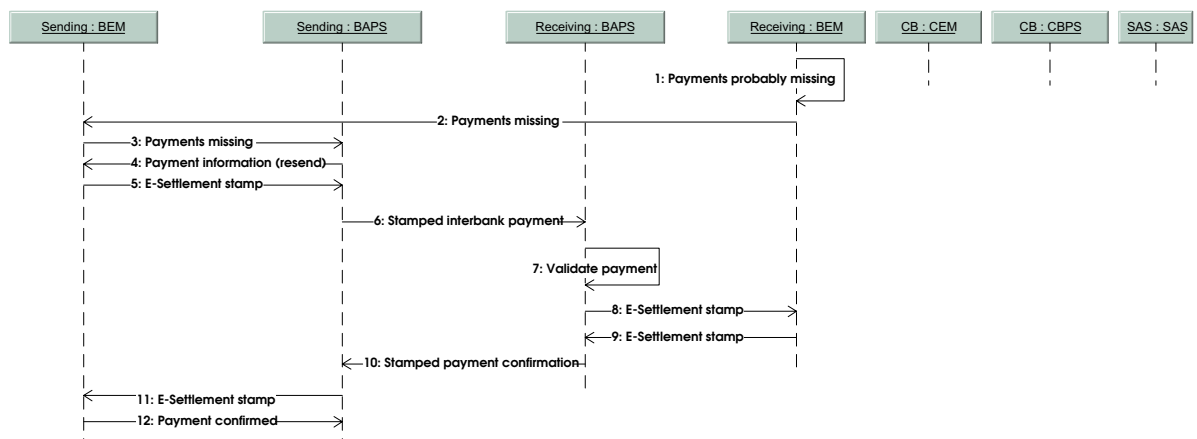


Figure 15. Payment missing

1: Receiving BEM deduces, based on the continuous bank-to-bank audit trail numbering of settlement stamps, that it is probably missing some payments. The deduction is based on a simple configurable timer after which the Receiving BEM should receive all previous settlement stamps.

2: Receiving BEM informs Sending BEM that it has no knowledge of the payment with the bank-to-bank audit trail numbers specified in the message.

3: Sending BEM forwards the information to Sending BAPS.

4: Sending BAPS requests resending of a payment by presenting payment resending information to its BEM. A new stamp is created but no additional liquidity is used for the stamp.

5–12: Normal payment process with the resent payment information is conducted. Since a payment could very well be meanwhile received and processed, all elements must be prepared to receive duplicate payment information and take care that the payment will not be processed twice.



## 4.2 Liquidity management processes

Liquidity processes issue and de-issue liquidity to/from BEMs. Both CEMs and BEMs have an IBAN account number, called “CEM account” and “BEM account” in this document, respectively. In this way, liquidity transfers can be handled as normal payment transactions. All liquidity operations are reported to SAS, so that SAS is always aware of the amount of liquidity currently in circulation. See also [CBIF\_SPEC] for more information.

Liquidity is issued to banks based on the collateral they have stored in the central bank. Each bank can receive E-Settlement liquidity up to the value of bank’s collateral. CEM issues liquidity to BEMs based on liquidity requests sent by BEMs. CEM forwards the requests to CBPS, which needs to validate each liquidity request against the knowledge it has of the amount of liquidity issued to the bank during the day and the value of the collateral of the bank.

At the beginning of the day, both CEMs and BEMs have a zero balance. During beginning-of-day activities, BEMs are issued liquidity up to the value of collateral they have stored in their central bank. Thus, at the beginning of day, CEM liquidity balance is negative exactly to the amount of liquidity issued to BEMs under CEM’s control. During the day, banks can request for more liquidity (if their collateral permits) or de-issue surplus liquidity. At the end of the day, after day’s final reconciliation, remaining liquidity is then de-issued to CEM by all BEMs. At this stage, all BEMs have again zero balance, and CEMs have a small negative or positive balance, depending on the intra-CEM flow of payments during the day. Each CEM can calculate its per-CB positions based on this information, and CB settlement can be conducted using these positions.

### 4.2.1 Liquidity issuing / liquidity request

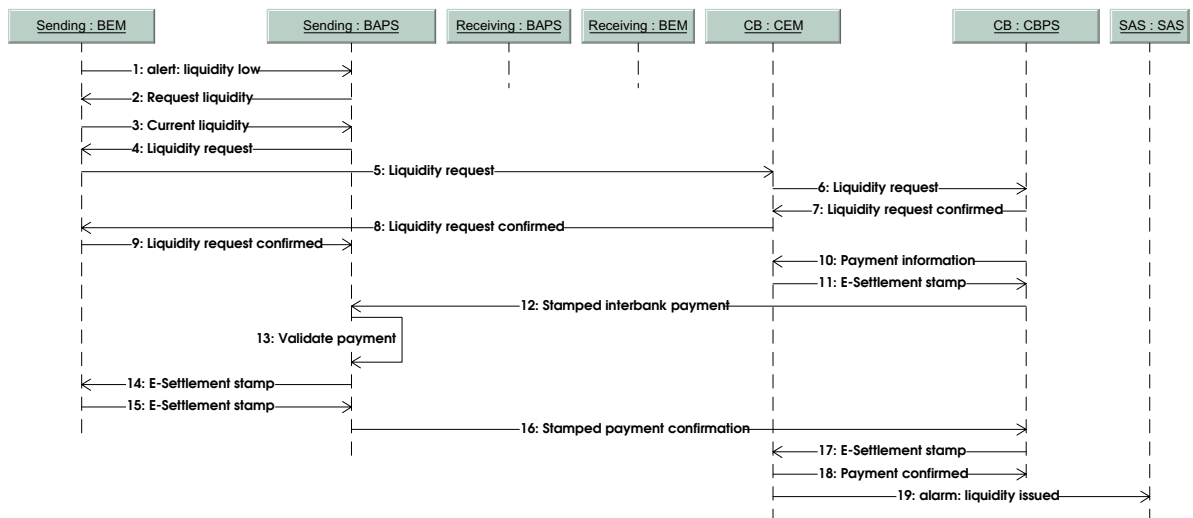


Figure 16. Liquidity request

This is a normal payment process from CEM account to BEM account activated Sending BAPS (after beginning-of-day activities this could be automatic).

1: BEM alerts BAPS that liquidity just got low, i.e. the “liquidity low” threshold is crossed.

2–3. BAPS requests current balance and amount of funds available from BEM.

4–6: BAPS sends a liquidity request to BEM, which in turn informs CEM of the request, which again notifies CBPS that BEM has requested a specified amount of liquidity.

7–9: CBPS checks that the request is valid and returns a confirmation to CEM. The confirmation is then relayed to BEM and BAPS. The confirmation includes current balance information.

10–18: Normal payment process from CEM account to BEM account.

19: CEM informs SAS that more liquidity has been issued to the system.

### 4.2.2 Liquidity de-issuing

During the day a bank might want to reduce the amount of liquidity in its disposal. This is achieved by transferring the excess liquidity from the BEM account to the CEM account. This happens also automatically during end-of-day activities.

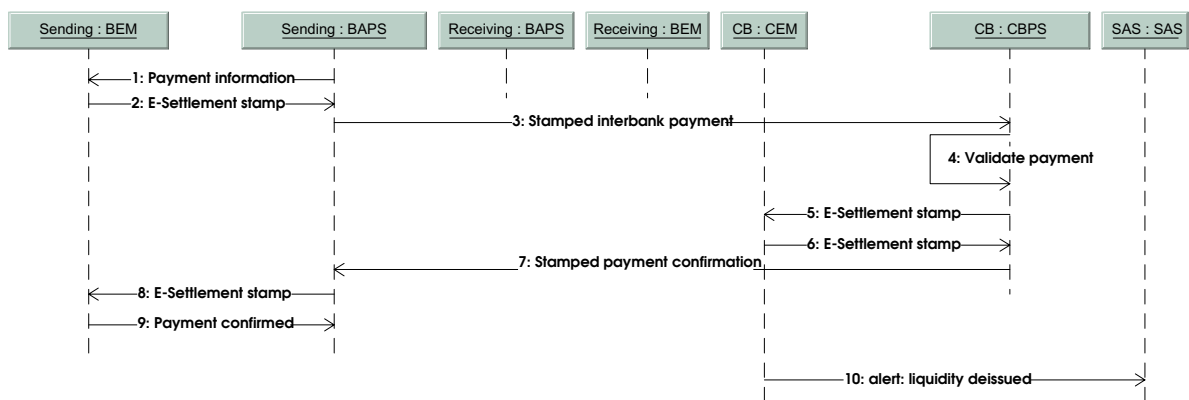


Figure 17. Liquidity de-issuing

This is a normal payment process from BEM account to CEM account, the only additional message being 10) when SAS is informed about de-issued liquidity.

### 4.2.3 Forced liquidity de-issuing

Sometimes it turns out that bank has “too much” liquidity. This can happen, for example, if the value of bank’s deposits have decreased in value drastically during the day. In these cases a central bank may decide that that a bank has more liquidity than the value of its deposits, a situation which poses a risk to the central bank. Although the situation could be corrected by the bank by de-issuing some liquidity, a mechanism for forcing de-issuing of bank’s liquidity is also presented. This case is quite special since liquidity is transferred via E-Settlement network, not via an Interbank network.

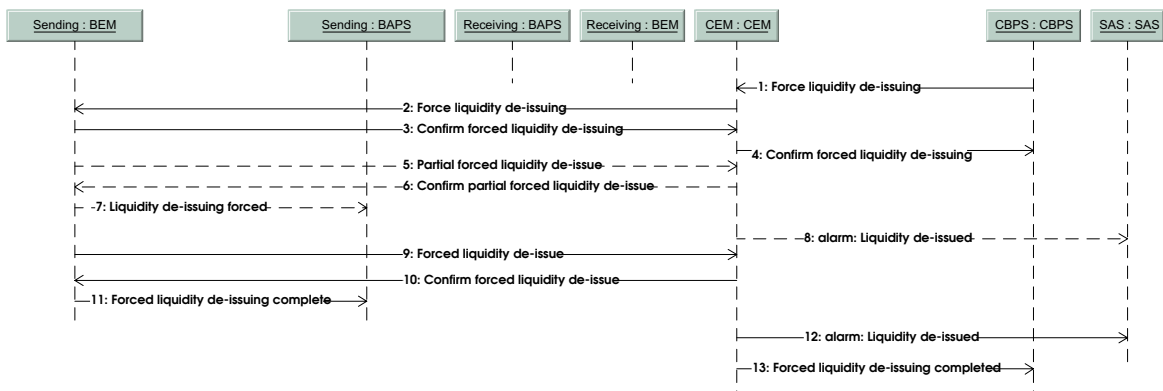


Figure 18. Forced liquidity de-issuing

1–2: CBPS requests that a certain amount of a bank's liquidity to be withheld. CEM forwards this requests to the BEM in question.

3–4: BEM confirms the forced liquidity de-issuing to CEM, which forwards the confirmation to CBPS.

5–8: If BEM does not have enough liquidity for the request at the moment, it sends a partial liquidity de-issue to CEM, which confirms the operation. BAPS and SAS are informed of the change in liquidity.

9–12: If BEM does have enough liquidity in the first place, or as enough liquidity to finish the forced liquidity de-issue becomes available, BEM does the operation at once. CEM confirms the operation, and BAPS and SAS are informed of the change in liquidity.

13: As the operation is finished, CBPS is notified.

### 4.3 Reconciliation processes

There are three levels of reconciliation in the E-Settlement system, performed at different times and different frequencies during the day:

1. Reconciliation between BAPS and BEM. This is based on comparison of BAPS/CBPS and BEM settlement statistics and transaction logs.
2. Continuous reconciliation. This is performed by the Receiving BEM every time an E-Settlement stamp is being validated, based on the bilateral payment history information contained in the stamp.
3. Bank-to-bank reconciliation. This is performed by the Sending BEM after the end of reconciliation periods, and also at other times if requested by the Sending BAPS. This is based on the bilateral payment transaction history stored by both Sending and Receiving BEMs. In Bank-to-bank reconciliation, the payment histories are compared to ensure that there are no discrepancies in between Sending and Receiving BEMs.
4. System reconciliation. This is performed by SAS at any time during the day. In system reconciliation, SAS compares the amount of liquidity issued by CEMs to the amount of liquidity reported by BEMs.

Reconciliation problems are resolved on bank-to-bank level. Some problems need to be escalated to SAS for resolving.

Note that although reconciliation processes are presented here from BAPS/BEM perspective, the same processes apply to CBPS and CEM.

#### 4.3.1 Reconciliation between BAPS and BEM

The basic mechanism in reconciling between BAPS and BEM information is to follow up BEM cumulative payment statistics, and if there discrepancies between these, finding in which individual transaction(s) the discrepancies lie. Correcting the discrepancies may require canceling payments using interbank messaging or even manual operations.

BAPS/CBPS can divide the day into a number of *bank reconciliation periods*. These are independent from the reconciliation periods of the E-Settlement system. BEM calculates cumulative total and bank-to-bank settlement indicators also for the bank reconciliation periods. BEM moves into next bank reconciliation period when ordered by BAPS.

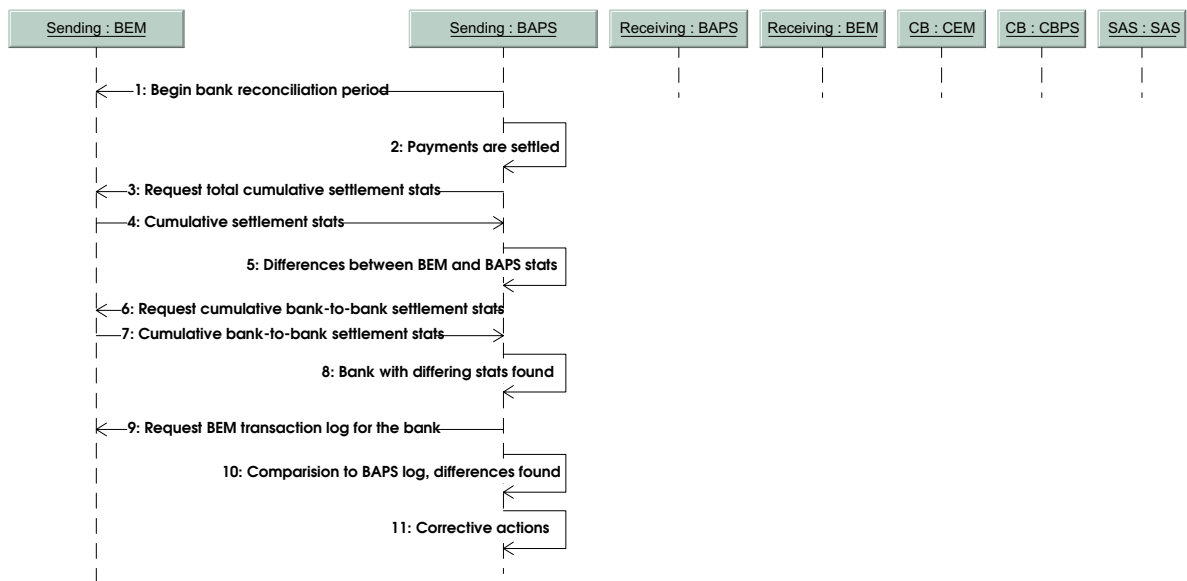


Figure 19. BAPS-BEM reconciliation

1: BAPS begins a new bank reconciliation period. This initializes the total and bank-to-bank cumulative BAPS settlement statistics counters for the new period in BEM. Note that E-Settlement system follows a different reconciliation schedule; system statistics are not affected.

2: Payment settlement in performed for some time.

3–4: BAPS request total cumulative settlement statistics from BEM.

5: There is a discrepancy in BAPS and BEM settlement statistics. This can happen for instance if BAPS has ignored some notifications sent by BEM: E-Settlement system may have implicitly canceled a settlement transaction, the notification of which has been ignored, or not received, by BAPS.

6–7 BAPS studies BEM cumulative bank-to-bank statistics to find the bank(s) where the differences lie.

8–10: The bank in question is found. Transaction log for the bank for the period is requested from BEM, and a comparison between BAPS and BEM transaction logs is performed. The differing payments are found.

11: BAPS performs corrective actions. For example, if BEM has implicitly canceled a transaction and BAPS has not, BAPS needs to cancel it at this stage.

### 4.3.2 Continuous reconciliation

Continuous reconciliation is conducted by the Receiving BEM by comparing payment information collected by the BEM to the information received from the Sending BEM. If discrepancies are found, bank-to-bank reconciliation process is started immediately.

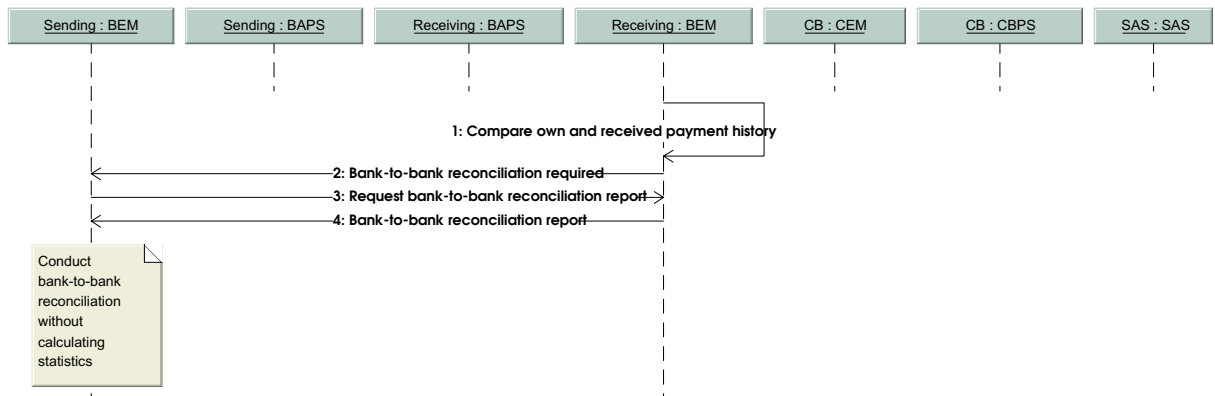


Figure 20. Continuous reconciliation

- 1: If there are discrepancies in the reported payment histories, a request for bank-to-bank reconciliation to Sending BEM is sent.
- 2: Receiving BEM requests sending BEM to start bank-to-bank reconciliation.
- 3: Normal bank-to-bank reconciliation process continues, including possible problem resolving.

### 4.3.3 Bank-to-bank reconciliation, success

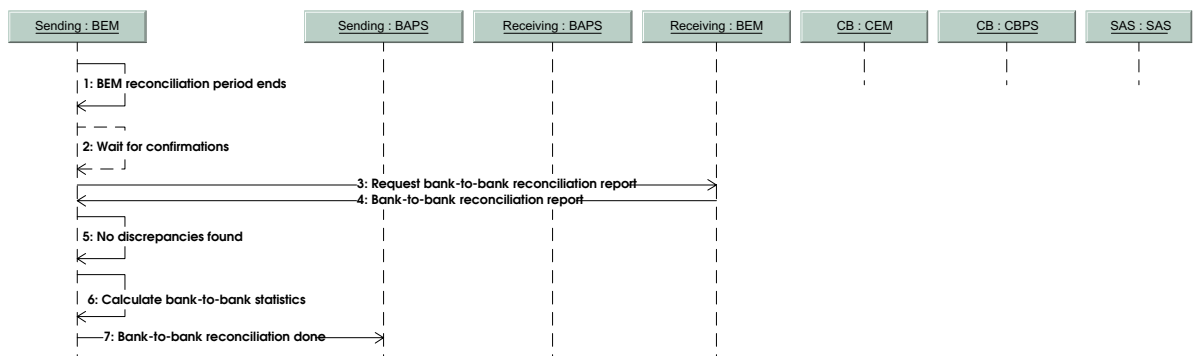


Figure 21. Bank-to-bank reconciliation, success

- 1: BEM concludes that the reconciliation period has ended.
- 2: Sending BEM can wait for the last few payment confirmations to arrive before attempting bank-to-bank reconciliation.
- 3: Since Sending BEM is responsible for the settlement transaction validity, it requests the bank-to-bank consolidation report from the receiving BEM.
- 4: Report is a log of E-Settlement transactions of the reconciliation period in question from Sending to Receiving BEM.

5-7: No discrepancies were found by the Sending BEM. Bank-to-bank statistics are calculated and BAPS is informed of the successful bank-to-bank reconciliation for the period.

#### 4.3.4 Bank-to-bank reconciliation, differences in unconfirmed payments

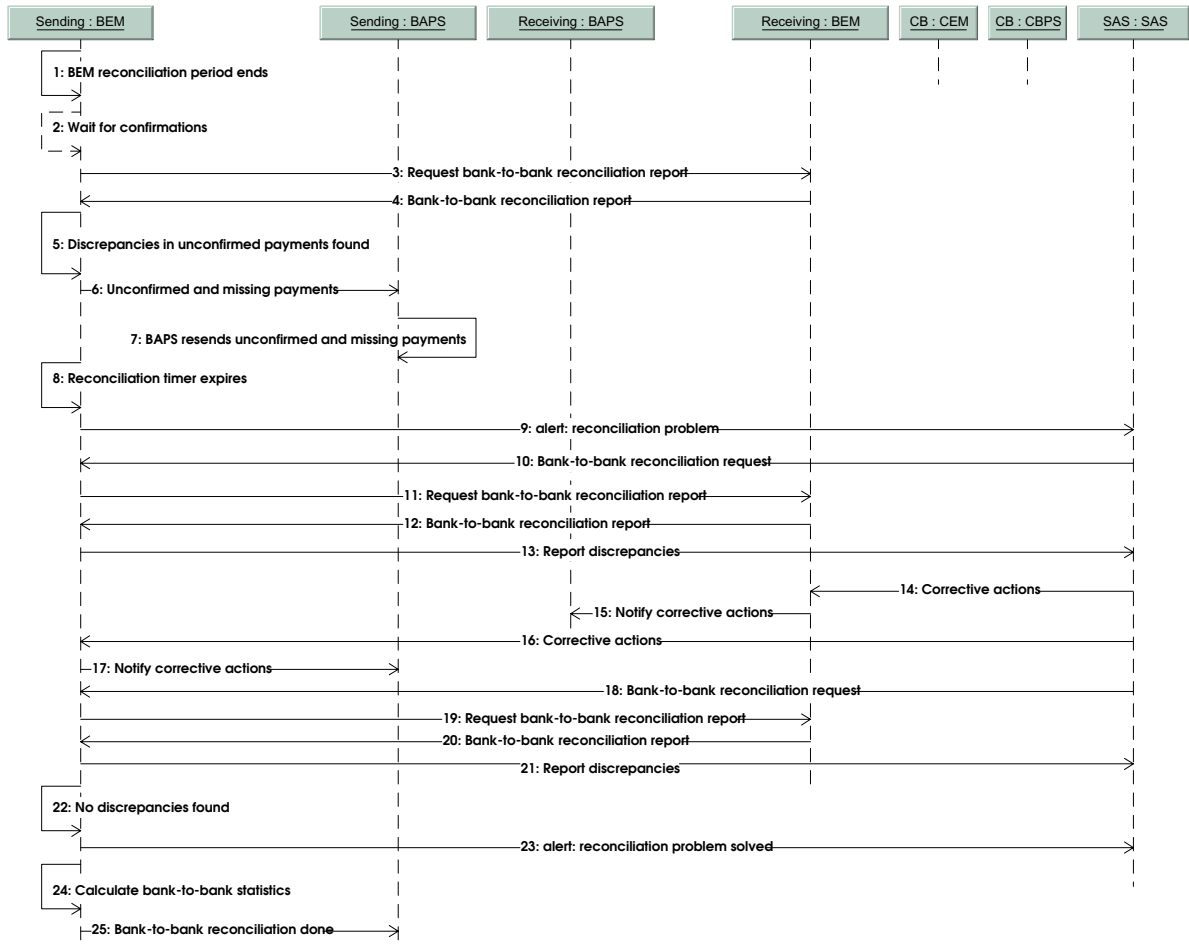


Figure 22. Bank-to-bank reconciliation, differences in unconfirmed payments

In this case, discrepancies in unconfirmed payments (some payments have not been confirmed and some payments may be missing from the receiving side) are found. We can expect this to happen, for example bank-to-bank reconciliation is performed immediately at the end of a reconciliation period. Note that this process is actually not so different from time-out and missing payment handling during the course of the reconciliation period.

1–4: Bank-to-bank reconciliation in begun normally.

5: Sending BEM studies the reconciliation report sent by the receiving BEM. There are discrepancies in unconfirmed payments – confirmation has not been received by the Sending BEM or settlement has not yet been received by the Receiving BEM.

6–8: Sending BAPS resends the unconfirmed and missing payments.

8–9: If BAPS does not manage to confirm the payments in due time, the problem is escalated to SAS. If BAPS does solve the problem, we would proceed to step 24.

10–13: SAS requests bank-to-bank reconciliation report from the Sending BEM.

14–17: SAS performs corrective actions. Unconfirmed transactions may be confirmed, missing transactions can be performed within the E-Settlement network. Both Sending and Receiving BAPSEs are notified of the forced operations.

18–21: To validate the corrective actions, SAS requests bank-to-bank reconciliation report from the Sending BEM.

24–27: The problem was solved, BEM informs SAS of this. Bank-to-bank payment statistics are calculated and BAPS is informed of the successful bank-to-bank reconciliation for the period.

### 4.3.5 Bank-to-bank reconciliation, differences in confirmed payments

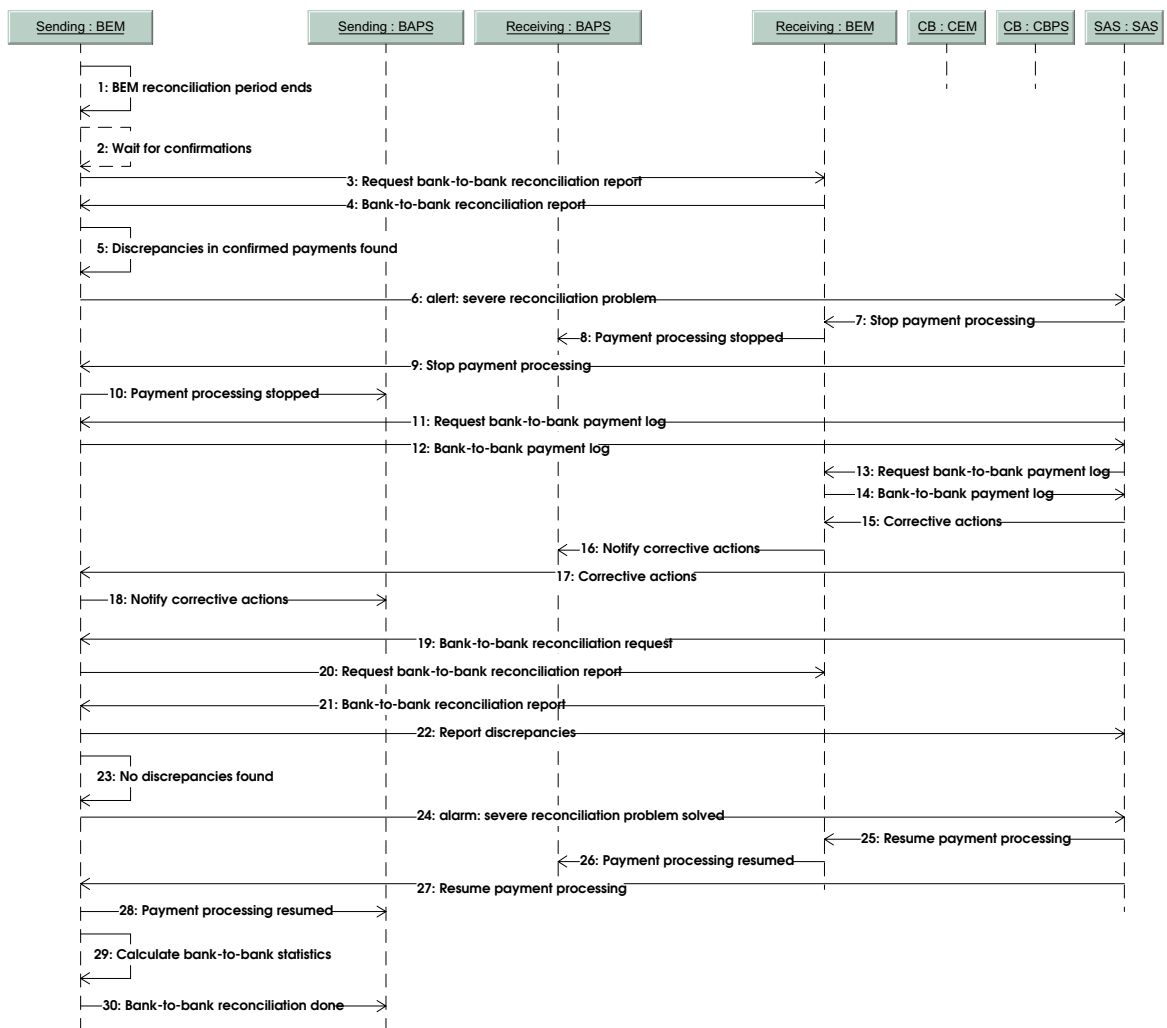


Figure 23. Bank-to-bank reconciliation, differences in confirmed payments

Differences in confirmed payments are serious problems for information consistency point of view. It basically means that liquidity may have “disappeared” or “created” in the system. In this solution these problems are immediately escalated to SAS.

1–5: Bank-to-bank reconciliation in begun normally. There are discrepancies in confirmed payments.

6: Sending BEM immediately alerts SAS on the problem.

7–10: Since the problem is severe, SAS orders both Sending and Receiving BEMs to stop payment processing. BEMs notify their respective BAPSEs of this.

11–14: SAS reads the bilateral payment logs of the banks in question.

15–18: SAS forces corrective actions to BEMs. For example, it can cancel all the transactions where problems lie. BEMs notify their BAPSEs of the actions performed.

19–22: To validate the corrective actions, SAS requests bank-to-bank reconciliation report from the Sending BEM.

23–30: The problem is solved. SAS allows the BEMs in question to resume payment processing. Bank-to-bank payment statistics are calculated and BAPS is informed of the successful bank-to-bank reconciliation for the period.

### 4.3.6 System reconciliation

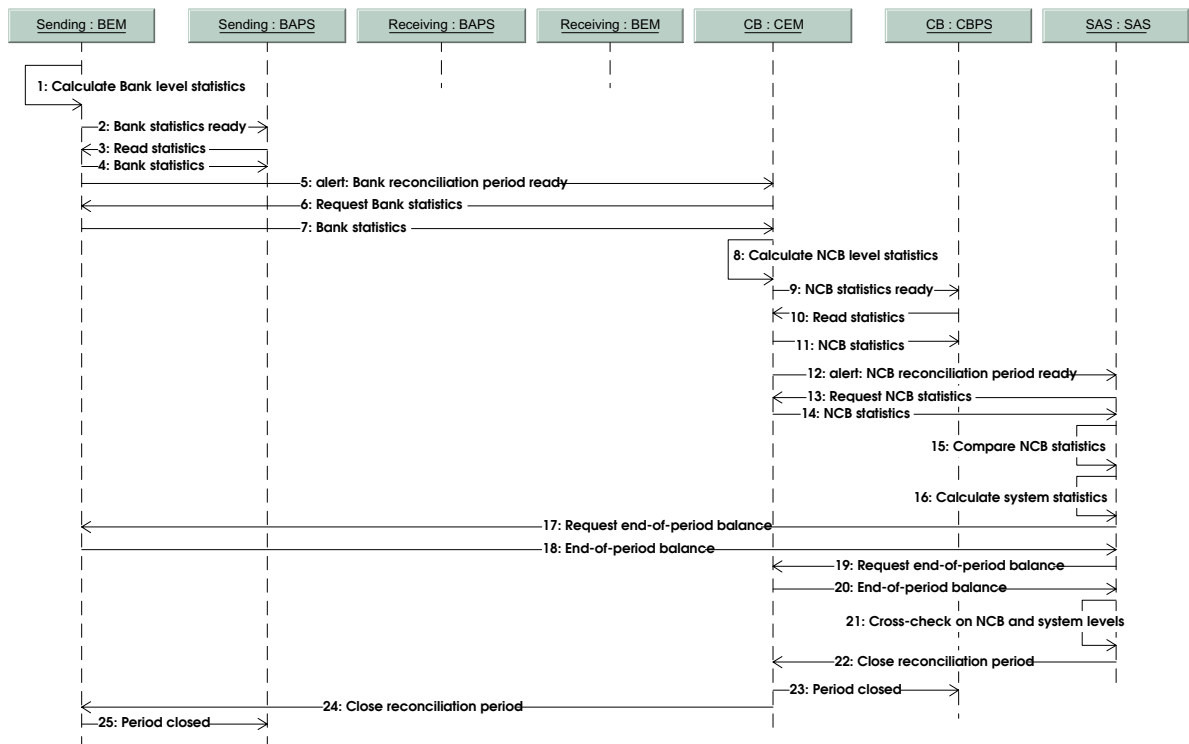


Figure 24. System reconciliation

1: BEM calculates its statistics for the reconciliation period based on the bank-to-bank statistics. BEM calculates payment sums and numbers on bank-to-bank and CB levels.

2–4: BEM informs BAPS that it has its statistics ready and BAPS reads these.

5–7: BEM informs CEM that it has its statistics ready and CEM reads these.

8: Once CEM has the statistics from all of its BEMs, it cross-tabulates the information and calculates CB level positions for the reconciliation period.

9–11: CEM informs CBPS that it has its statistics ready and CBPS reads these.

12–14: CEM informs SAS that it has its statistics ready and SAS reads these.



15: SAS compares the CB level positions reported by CEMs.

16: System statistics are calculated based on the CEM statistics.

17–20: SAS requests all BEMs and CEMs their liquidity balances.

21: There are multiple cross-checks on the amount of liquidity reported by the E-Settlement modules:

1. SAS reads the amount of liquidity issued from all CEMs and compares the figure to the amount reported to it.
2. SAS reads the amount of liquidity received from CEM and the current liquidity balance from all BEMs.
3. For each CEM, the amount of liquidity received as reported by its BEMs is compared to the amount of liquidity reported issued by CEM.
4. The total amount of liquidity in circulation as reported by CEMs is compared to the total of liquidity balances as reported by BEMs.

22–23: SAS instructs CEM to “officially” close the period. CEM informs CBPS of this.

24–25: CEM instructs BEM to close the period. BEM informs BAPS of this.

## 4.4 Control processes

### 4.4.1 BEM start-up and beginning-of-day activities

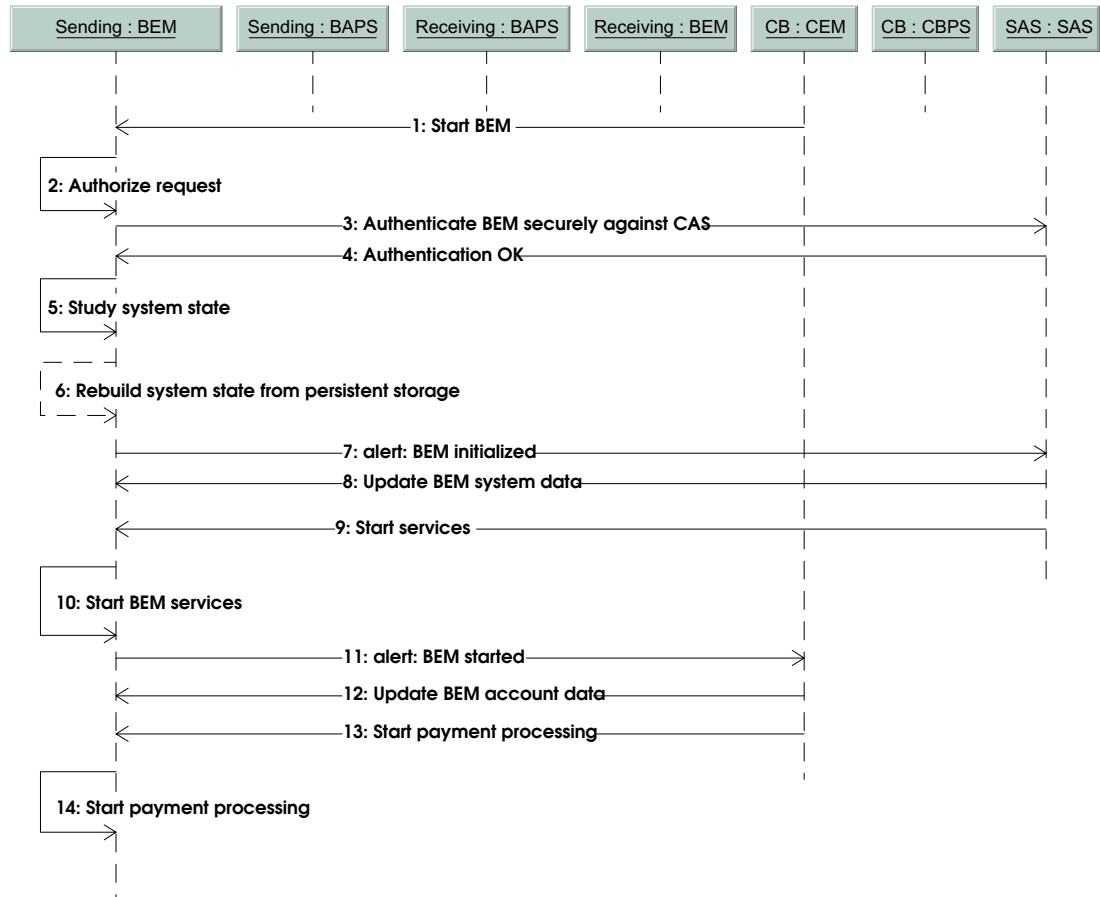


Figure 25. BEM start-up and beginning-of-day activities

The BEMs in E-Settlement system are all similar. BEM receives an identity from the smart card stored inside it, containing identity information and digital certificate created by CEM. The only pre-configured information BEM has to know is the address of SAS to which BEM sends an alarm that it has woken up. SAS can then provide all other information from system directory.

1–2: CEM sends BEM a request to start, together with authorization information. BEM authorizes the request.

3–4: After successful authorization BEM authenticates itself securely against the network using the encryption facilities of the smart card.

5–6: Previous system state is examined based on persistent information. If there is liquidity balance and/or the transaction log shows that the system was not closed “cleanly”, system state is regenerated by going through the transaction log from the beginning of the reconciliation period.

7: BEM sends an alarm to SAS that it is ready for initialization.

8: This includes: opening hours, reconciliation period rules, payment time-outs 1 (warn SAS, inform BAPS) and 2 (alert SAS), reconciliation time-out, BEM’s IBAN, address of CEM controlling this BEM, netting module information plus warning and alert thresholds for payment/system control tasks.

9–10: SAS orders BEM to start services.

11–12: BEM contacts its CEM and requests account information. This includes funds available to the bank, “liquidity low” threshold and bank profile information.

13–14: CEM orders BEM to start payment processing. BEM can now receive payments from other banks, but first needs some liquidity from CEM to be really able to process payments.

### 4.4.2 End-of-day activities

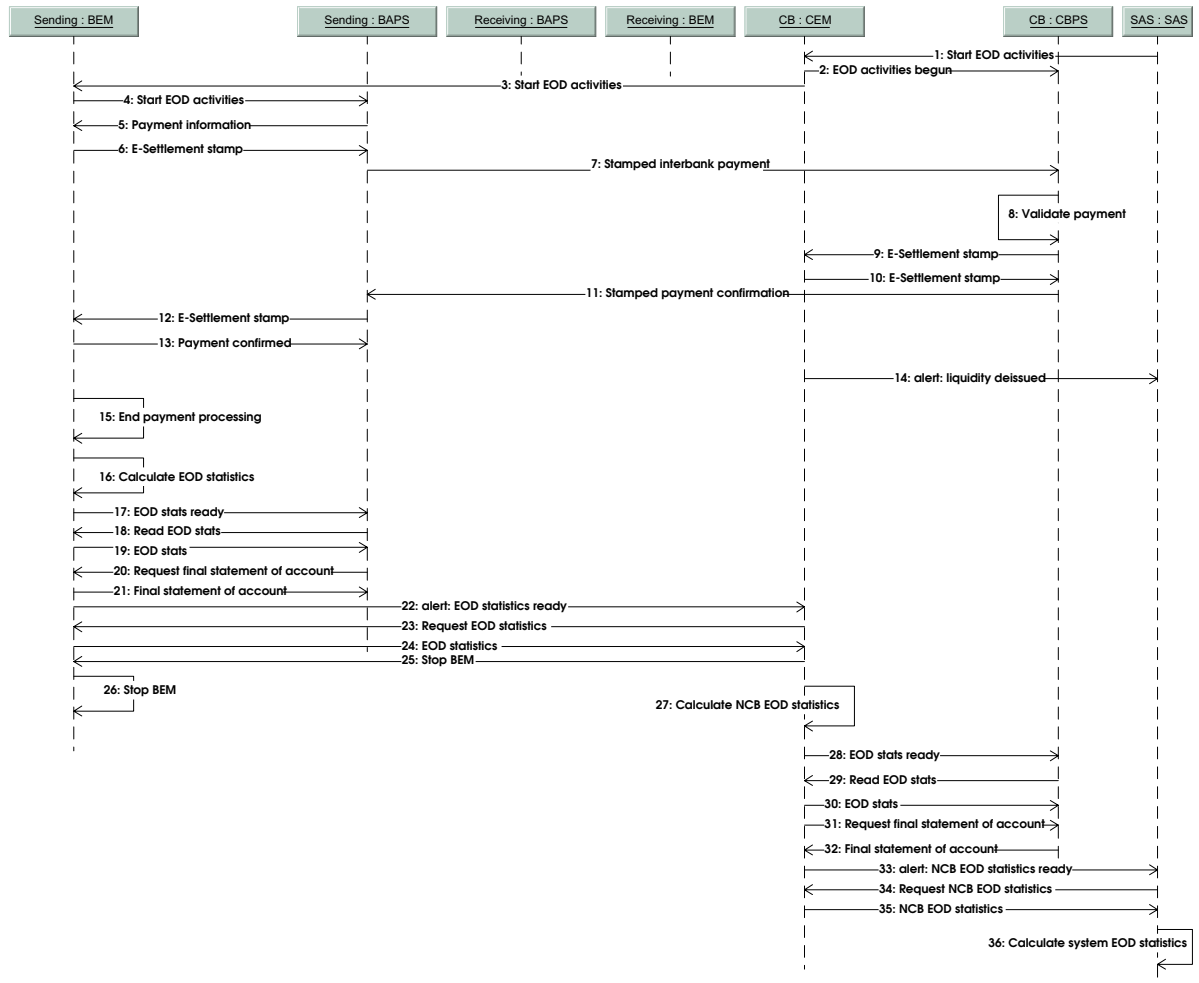


Figure 26. End-of-day activities

1–4: SAS orders EOD activities to commence. The call is forwarded by CEM its CBPS and its BEMs , and from BEMs to their BAPS.

5–14: EOD liquidity de-issuing process takes place.

15: BEM stops payment processing.

16: EOD statistics = sum and number of payments with each bank, sum and number of payments to/from each CB.

17–21: BEM informs BAPS that its EOD statistics are ready. BAPS reads the EOD statistics and BEM’s final statement of BEM account for the day.

22–26: BEM informs CEM that its EOD statistics are ready. CEM reads and stores the statistics and orders BEM to stop.

27: CEM calculates its EOD statistics when all BEMs under its control have been stopped. CEM EOD statistics are a cross-tabulation of all BEM's statistics on per-bank and per-CB level.

28-32: CEM informs CBPS that it has EOD statistics ready. CBPS reads these, as well as the final statement of CEM account.

33-35: CEM informs SAS that its EOD statistics are ready. SAS reads the statistics.

36: once SAS has the EOD statistics of the day of all CBs, it calculates system EOD statistics.

CB level EOD liquidity transfer can be easily added to this process.

#### **4.4.3 Payment control**

Payment control process follows up payment traffic and calculates various payment statistics needed in analyzing the performance of the E-Settlement system. During this process BEM does the following:

- Signals to CEM and SAS that it is OK
- Calculates activity information, i.e. how many and how large payments it has processed during the period in which activity is followed up
- Calculates availability and downtime statistics. There are availability requirements for settlement systems, and these are frequently controlled by ECB, for example.
- Calculates payment performance statistics

SAS reads and stores the following:

- Activity information
- Availability information
- Downtime statistics
- Payment performance statistics

#### **4.4.4 IT system control**

IT system control process follows up BEM resource usage. In addition, SAS can upload and activate new versions of software.

BEM does the following

- Checks IT resource usage and alerts SAS if warning or alert limits are crossed.

SAS does the following:

- Reacts to warnings and alerts regarding IT resource usage
- Reads software versions currently used by BEM
- Uploads a new software version to BEM
- Set the software versions to use upon next startup

## 5 EXCEPTION HANDLING

### 5.1 Exceptional situations

A great number of exceptional situations can happen in a complex distributed system. E-Settlement system must be prepared to handle the most common exceptional situations automatically (procedural exceptions), and be able to handle some more exceptions with manual help (manual exceptions). There are also exceptions that are difficult to handle in a distributed system without fallback systems and reprocessing routines (fatal exceptions). What follows is a list of exceptions in each of these three groups group.

#### Procedural exceptions

- Gaps in audit trail of received payments
- Double reception of E-Settlement stamps by Receiving BEM
- Double reception of confirmation by Sending BEM
- Incorrect Receiving BEM address
- Payment timeout in Sending BEM
- Bank profile violation
- E-Settlement network partly down
- BAPS temporarily down
- Interbank network temporarily down
- BEM temporarily down

#### Manual exceptions

- Bank-to-bank reconciliation when Receiving BEM is not responding
- Bank-to-bank reconciliation when no confirmation to some payments has been received from Sending BAPS

#### Fatal exceptions

- E-Settlement network totally down (SAS not reachable)
- Security infrastructure compromised
- Major (WTC type) catastrophe, resulting in many bank sites being completely destroyed simultaneously

## 5.2 Disaster recovery

A distributed system coupled with redundant system components can be designed to be very disaster resistant. E-Settlement modules have redundant parts for the crucial elements such as Hardware Security Modules, hard disks, write-only transaction log devices. E-Settlement modules can be duplicated, and other system elements in the system (BAPSEs, CBPSs and SAS) need to have a (possibly even more than one) mirrored Disaster Standby Site (DSS). E-Settlement follows a layered approach in ensuring recovery from disasters:

1. If a crucial BEM component (HSM, hard disk, write-only transaction device) breaks down, the redundant component is taken into use transparently. This is notified to management systems and corrective maintenance actions are begun immediately.
2. If a bank's primary BEM breaks down irrecoverably, BAPS sends the secondary BEM the settlement stamps sent and received by the primary BEM from the beginning of reconciliation period. Every stamp includes the BEM as redundant information. This makes the transfer process easier to handle. Secondary BEM reconstructs its state based on the received information. SAS checks the situation of the secondary BEM, and only after a command from SAS can the secondary BEM start payment processing.
3. If a bank's primary BAPS site goes down or is destroyed, the mirrored DSS BAPS is taken into use. DSS BAPS contains a BEM, which will be taken into use as described above.
4. If a bank's all BAPSEs and BEMs are destroyed, the state of the BEM can still be reconstructed. SAS requests the transactions performed with the destroyed BEM from other BEM since the beginning of the current reconciliation period. These transactions can be used in forming the state of BEM at the time of destruction.
5. If several bank's all BAPSEs and BEMs are destroyed simultaneously, some transactions may be lost. Banks' liquidity situation at the end of the previous reconciliation period is known, and to some extent the transaction histories of the destroyed BEMs may be reconstructed manually.

It is worth noting that in a distributed system, even if some parts break down, the other parts continue operating normally. This is in contrast to a centralized system where the breakdown of the centralized system incapacitates the whole system.

## 6 MANAGEMENT ARCHITECTURE

System management plays a big part in E-Settlement system. To analyze management issues in a complex system, it makes sense to introduce two concepts widely used in telecommunications field. FCAPS (Fault, Configuration, Accounting, Performance and Security) model divides management issues into *task areas*. Three different *levels* of management are introduced in TMN (Telecommunications Management Network) thinking: business management, network management and element management.

These two concepts allow us to make a tentative analysis of E-Settlement system management needs on different levels. It has been further assumed here that the actual network will be managed by a network provider, and have concentrated here on “E-Settlement management” issues.

**Business management:** organizational processes related to E-Settlement system

- Fault management: incident handling process
- Configuration management: continuous development of E-Settlement system
- Accounting: storing the transaction logs and statistics
- Performance management: analyzing the cost effectiveness of the system
- Security management: risk analysis

**Network management:** taking care of the networked application

- Fault management: reacting to alarms sent by E-Settlement modules
- Configuration management: adding new banks / CBs / users, maintaining system directory information
- Accounting: event and change logs on network level
- Performance management: following up the performance of the system
- Security management: PKI management, smart card management, maintaining user profiles

**Element management:** managing the E-Settlement modules

- Fault management: tracking hardware and software problems
- Configuration management: managing OS and software versions
- Accounting: keeping system event logs, transaction logs and payment statistics up to date
- Performance management: tracking performance of payment traffic and HW resource usage
- Security management: secure authentication with smart cards, PKI, enforcing system controls on audit trails and user profiles

## 7 SECURITY ARCHITECTURE

A solid security solution is crucial for the E-Settlement system concept. E-Settlement Security document [SECURITY] points out some central security management issues of the system. The approach chosen in E-Settlement is to follow a standard process for creating an information security management framework. The process specified in [ISO 17799]/[BS 7799] will be used in defining the security management framework for the E-Settlement system. The process entails stating a security policy, defining the information that needs to be secured, assessing the risks that the secured information can be exposed to and choosing the security controls appropriate for the risks, and then applying these controls to a system.

The main benefits of using a standard process for security and risk management is that the process and its risks can be audited by external parties, and that the use of a standard process works towards ensuring that no common security vulnerabilities will be overlooked during the system development process. Chosen process should also lead to security architecture that will allow use of new security controls against new security threats arising in course of time.

This section describes the different facets of information security and the key security mechanisms used in the E-Settlement system on a quite general level. More detailed description of security solution is in separate documents, see [SECURITY\_POLICY] and [SECURITY\_ANALYSIS].

### 7.1 Layered approach to security

The guiding principle in designing E-Settlement security is that multiple layers of security are built into the system right from the beginning. In case one of the layers is compromised, the other layers still prevent misuse. From security point of view, the layers are independent of each other and can use different mechanisms for ensuring security. The following are the main layers of E-Settlement security:

1. Tamper-proof storage level. Critical information is stored in a tamper-proof storage called Hardware Security Module (HSM) inside the BEMs and the CEMs.
2. E-Settlement module level. E-Settlement modules (BEMs and CEMs) are physically secured and implement a number of security controls to ensure end-to-end security of messaging. Module level security is controlled at a system level i.e. it is configured from system level and it will escalate alerts to system level.
3. Communications level. All information exchange between E-Settlement system elements, as well as E-Settlement stamps transferred between banks, is secured using a set of technologies to ensure confidentiality, authenticity, integrity, and repudiation of information transfer.
4. System level. Payment information is cross-checked many times over by different system elements during payment processing and reconciliation. System level is also responsible for monitoring and incident handling of E-settlement modules.
5. Storage level. Settlement transaction logs are stored in an encrypted form so that only the processes that need to be able to decrypt the transactions can do so. In addition, the transactions are stored in “write-once, read multiple times” memory to further add to the security of data storage.



## 7.2 Facets of E-Settlement security

Different kinds of security solutions are needed in different parts of the E-Settlement system. below presents the facets of E-Settlement security, which are presented shortly in this chapter.

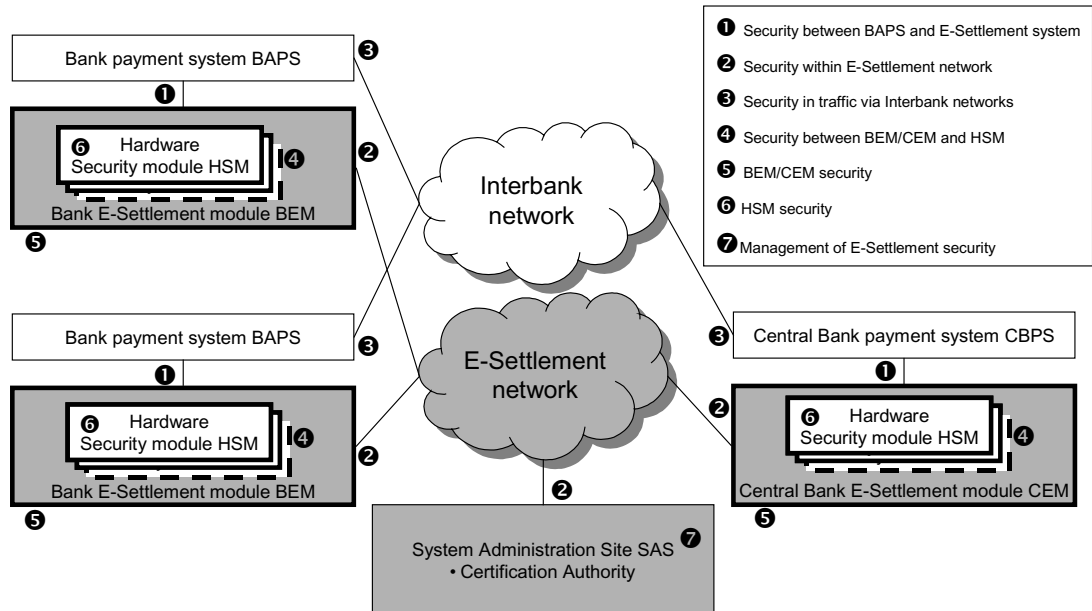


Figure 27. Facets of E-Settlement security

### 7.2.1 Security between bank payment systems and the E-Settlement system

Between BEM/CEM and BAPS/CBPS, a range of methods is used for ensuring security:

- Physical security (banks and central banks are responsible for ensuring that traffic between BAPS/CBPS and BEM/CEM is secure)
- Physical security with communications secured with a certificate issued by the E-Settlement CA
- Physical security with communications secured with a smart card authentication and certificate, both issued by the E-Settlement CA

### 7.2.2 Security within E-Settlement network

Although E-Settlement network is a dedicated network, communications must be secured, since banks' confidential settlement information is being transferred in the network. BEM, CEM and SAS utilize IPSec technologies with authentication provided by PKI for this purpose. See 7.2.7 for information regarding how the PKI infrastructure in E-Settlement system is managed.

### 7.2.3 Security in traffic via Interbank networks

One of the central ideas of E-Settlement is that settlement information is carried inside the payment information in single flow of information. To do this securely requires that the settlement information is transferred encrypted, "tunneled" in the payment message. Figure 28 below presents the idea of tunneled E-Settlement stamps. Reflecting the layered security approach of the E-Settlement system, E-Settlement stamps are secured at two levels: between BEMs/CEMs and between the respective HSMs.

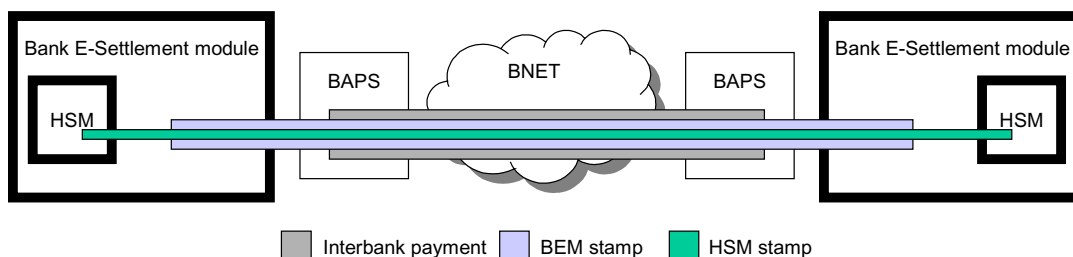


Figure 28. Tunneling of E-Settlement stamps

### 7.2.4 Security between BEM/CEM and HSM

BEM/CEM and HSM operate independently. There is a strictly defined interface between BEM/CEM and HSM. HSM stamps are created by HSM upon request by BEM/CEM, but the information contained in HSM stamp is encrypted so that BEM/CEM is not able to understand or modify the HSM stamp. To further strengthen the security between BEM/CEM and HSM, these parts will be designed and implemented by different companies, so that no outside party has complete knowledge of the E-Settlement security solution.

### 7.2.5 BEM/CEM security

BEMs and CEMs are situated in the central banks' and banks' premises. The premises must be physically secure, with access only by authorized personnel. In addition, the BEMs and CEMs must fulfill the following criteria:

- BEM/CEM need to be secured with physical and procedural means so that only authorized personnel can maintain BEM/CEM hardware and software, both locally and remotely.
- BEM and CEM need to be under strict configuration management and change control. Only tested and approved software versions are run and no unnecessary services are run.
- BEM and CEM need to store their transaction logs so that they cannot be modified later, for example into a WORM (write once, read multiple times) drive.

### 7.2.6 HSM security

HSMs store critical information – such as private keys and liquidity balance - in the E-Settlement system. The modules need to be tamper proof and fulfill a specified security standard (FIPS 140-1).

### 7.2.7 Management of E-Settlement security

The security of E-Settlement system is based on Public Key Infrastructure managed by E-Settlement system Certification Authority at SAS site. A CA is responsible of the following tasks, see [ECBS TR402]:

- Subscriber registration

- Certificate revocation and suspension
- Certificate distribution and usage
- Certificate change management
- Notarial functions
- Archiving functions
- Time stamping
- Key escrow and key recovery
- Attribute authority

## 7.3 Key security mechanisms used in E-Settlement

This chapter describes the key security mechanisms in E-Settlement system from a mainly technical point of view.

### 7.3.1 Public Key Infrastructure (PKI)

Public Key Infrastructure and its related cryptographic services provide means for secure communications over an untrusted network. PKI can be applied in dedicated network, too, if security requirements for messaging are high, as is the case in the E-Settlement system. Keys used in the E-Settlement need to be long enough for the purpose. PKI is used in E-settlement system for ensuring authentication, transaction integrity and non-repudiation of messaging between the E-Settlement system elements.

### 7.3.2 Tamper-proof storage

Tamper-proof storage contains the critical system data securely and persistently. This includes the identity of the element, liquidity balance and private encryption key. Tamper-proof storage can be either a smart card or a secure memory module inside BEM and CEM. Within E-Settlement system, tamper proof storage is used for ensuring availability and confidentiality of critical system data.

### 7.3.3 Write-only secure transaction logs

All E-Settlement transactions are stored to a write-only log such as a WORM (write once, read many times) drive. This ensures that all transactions can be later analyzed by the relevant parties. Information in the transaction log is encrypted so that only the parties that need to know the transaction information can do so. Within E-Settlement system, write-only transaction log is used for ensuring integrity, non-repudiation and auditability of payment information.

### 7.3.4 Encryption of E-Settlement stamps

E-Settlement stamps are digitally signed by the sending BEM and then encrypted for the receiving BEM to read. Stamp is end-to-end secured (i.e. BEM-to-BEM, BEM-to-CEM). Thus, the stamps can be securely transmitted over an untrusted network, with the receiving BEM able to ensure the identity of the sender. In addition, in E-Settlement system the stamps are actually doubly secured, since HSMs encrypt a part of the settlement information independently of the BEM using a key pair of its own, with the receiving HSM validating this part of the stamp. Within E-Settlement system, encrypted stamps are used for ensuring integrity, authentication, and confidentiality of payment messaging.

### **7.3.5 Distributed storage**

E-Settlement payment and stamp information are stored both by the sending and receiving banks' payment systems and E-Settlement modules. Furthermore E-Settlement information can be collected and stored by CEMs and SAS as required. Thus, in the cases of severe system failure, payment and settlement information can be gathered from other, geographically separated sources, thereby reducing single-point-of-failure risks. Within E-Settlement system, distributed storage is used for ensuring availability and integrity of payment and settlement information.

### **7.3.6 Programmed controls**

Several programmed security controls are implemented by the system:

- The lifetime of Central Bank money is limited by the system
- Settlement traffic can be stopped altogether or to a specific bank at any time
- Bank profiles are enforced
- Reconciliation is performed at many levels and continuously
- Liquidity situation in the E-Settlement network can be checked at real time

Within E-Settlement system, procedural controls are used for ensuring availability, authenticity, proper authorization and auditability of information.



## **Section 3**

### **E-SETTLEMENT**

### **SYSTEM SECURITY SPECIFICATION**

The views expressed here are those of the project. They are at the moment preliminary and open for all comments. The views do not necessarily reflect the views of the Bank of Finland.

---

## TABLE OF CONTENTS

<b>TABLE OF CONTENTS .....</b>	<b>68</b>
<b>ABBREVIATIONS.....</b>	<b>69</b>
<b>NOTATIONS.....</b>	<b>71</b>
<b>1 E-SETTLEMENT SECURITY .....</b>	<b>72</b>
<b>2 E-SETTLEMENT SECURITY POLICY.....</b>	<b>73</b>
2.1 OBJECTIVES .....	73
2.2 SCOPE .....	73
2.3 PRINCIPLES .....	73
2.4 AREAS OF INFORMATION SECURITY.....	74
2.5 SECURITY RESPONSIBILITIES .....	75
2.6 INCIDENT HANDLING.....	76
<b>3 E-SETTLEMENT SECURITY ANALYSIS.....</b>	<b>77</b>
3.1 OBJECTIVES .....	77
3.2 SECURITY MEASURES .....	77
3.3 CLASSIFICATION OF THREATS .....	80
3.4 ANALYSIS .....	81
<b>4 E-SETTLEMENT SYSTEM SECURITY SPECIFICATION .....</b>	<b>87</b>
4.1 INTRODUCTION .....	87
4.2 FACETS OF E-SETTLEMENT SECURITY.....	87
4.3 E-SETTLEMENT STAMP .....	90
4.4 TRANSACTION LOG .....	102
4.5 HSM INTERFACE .....	104
<b>5 CONCLUSIONS ON E-SETTLEMENT PROTOTYPE SECURITY .....</b>	<b>107</b>
<b>APPENDIX A : DETAILED STAMP HANDLING PROCESSES .....</b>	<b>108</b>
<b>APPENDIX B : ALGORITHMS.....</b>	<b>113</b>

## ABBREVIATIONS

Abbreviation	Meaning
API	Application Programming Interface
BADR	Bank Driver
BAPS	Bank Payment System
BEM	Bank E-Settlement Module
BNET	Inter-bank Network
CA	Certification Authority
CBDR	Central Bank Driver
CBPS	Central Bank Payment System
CEM	Central bank E-Settlement Module
DES	Data Encryption Standard
ECSTAMP	E-Settlement Confirmation Stamp
ESN	E-Settlement Network
ESTAMP	E-Settlement Stamp
FIPS	Federal Information Processing Standards
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPsec	IP Security Protocol
ISO	International Standards Organization
PKC	Public Key Certificate
PKI	Public Key Infrastructure
RSA	Public-key algorithm developed by Rivest, Shamir and Adelman
SAS	System Administration Site
SDI	System Directory
SHA	Secure Hash Algorithm
SSL	Secure Sockets Layer



TLS	Transport Layer Security
WORM	Write Once Read Many times

## NOTATIONS

Notation	Meaning
$E_K(M)$	Encrypt message $M$ with key $K$
$D_K(C)$	Decrypt cipher text $C$ with key $K$
$S_K(M)$	Sign message $M$ with private key $K$
$V_K(M)$	Verifying signature with public key $K$

---

# 1 E-SETTLEMENT SECURITY

One of the cornerstones behind E-Settlement system is security implementation. The main focus has been largely on information security issues and there are areas of security such as personnel, premises, contracts etc. that have not been under considerations.

In early stage of the prototype work the BS7799 standard was chosen as a security management framework. This framework offers good and commonly understandable guidelines for planning security. The process followed consists of building a security policy, making risk assessments and writing the specifications for the security features/system. Based on this process the security implementation plan for the prototype was built with focus on areas that are new to distributed system such as E-Settlement and those that are utilising new technologies.

One key element for security planning in the E-Settlement prototype has been the design of a layered security structure. The idea is to have at least two preventive measures and one detective measure for each significant risk. Layered security is also a major challenge for the implementation work.

In the technology area there are solutions that are characteristic for the E-Settlement system

- Electronic segregation of duties
- Hardware security module
- Public Key Infrastructure
- Logging

One of the key measures for ensuring security in E-Settlement is electronic segregation of duties. In each bank module (BEM) there is a separate hardware security component (HSM). For each operation both BEM and HSM has to participate in order to complete any monetary or security operation.

Public Key Infrastructure (PKI) certificate-based authentication, integrity control and encryption are used in E-Settlement prototype. Implementing trustworthy PKI infrastructure is a major effort. Based on experiences of the project, the technology as such is fairly straightforward to implement, but the effort is in defining and implementing policies and practices. In the prototype project separate certificate authority (CA) has been created for system as well as directory services that carry certificate information and certificate revocation lists.

PKI implementation in the E-Settlement system is layered based on electronic segregation of duties i.e. independent PKI-based cryptography is used in communication from HSM to HSM and from BEM to BEM. There is even third layer of encrypt communication as the prototype uses secured network (VPN) for carrying information.

Audit trails and transaction logs are part of security control forming its own layer. Extra attention has to be paid to make sure this information cannot be modified. Procedures for writing audit trail and log information should be included in application software components.

---

## 2 E-SETTLEMENT SECURITY POLICY

### 2.1 Objectives

The purpose of information security is to ensure business continuity and minimize business damage by preventing and minimizing the impact of security breaches. This goal is achieved by protecting the integrity, confidentiality and availability of critical data and processes.

This document guides different parties to manage risks related to information security in the E-Settlement environment. The information security level should be higher than in banking sector generally. As the E-Settlement system concept is of highly distributed nature, it sets special requirements for securing communications. Also the level of availability of network has to be set high, which leads to a requirement for eliminating single-point-of-failure components. Distributed nature sets special requirements for physical security, as part of the hardware and software is located in the premises of each participant. Also, the very large amounts of liquidity transferred stresses the importance of the security solutions.

### 2.2 Scope

This security policy covers the E-Settlement concept divided in three parts:

E-Settlement	That is the whole concept/system as it is defined in the System Architecture Specification
Prototype	That is the system prototype that is to be built before any production version. One of the key objectives for the prototype is to verify the feasibility of the security framework architecture.
Development work	That is the system development work performed during the design, implementation and testing stages of the E-Settlement and prototype parts.

The information in E-Settlement environment regarded as highly critical includes:

- All balances and information related to liquidity
- Bank profile data
- Private keys
- Transactions and transaction logs

### 2.3 Principles

- 1) Critical information related to security issues shall be classified and handled as secret information.
- 2) Technology used for implementing security controls should be qualified in open forums or certified according to security standards (such as CC, FIPS) whenever applicable.
- 3) Security measures shall be based on risk analysis. The first risk analysis is done as part of defining the system architecture. Risk analysis shall be reviewed at regular intervals or per need basis.

- 4) For each significant threat there must be at least two preventive measures and one detective measure implemented.
- 5) All faults in the system must be traceable.
- 6) Critical data should be stored in encrypted format.
- 7) All communications must be encrypted.
- 8) 4-eyes principal and the segregation of duties should be used for critical functions.
- 9) Corrupted or lost information due to system or power malfunction must be restorable to its former condition.
- 10) It should be possible to build a complete audit trail using transaction logs at any given moment.

## **2.4 Areas of information security**

### **2.4.1 Availability**

Information must be accessible to and usable by an authorized entity on demand. Critical tasks shall be performed when they are necessary, not earlier or later, and non-critical task shall not be made time critical. Access to resources must be possible when it is needed, and resources should not be requested or retained unnecessarily. System must include error detection and error recovery functions intended to restrict the impact of errors on the operation of the system and so minimize disruption or loss of service.

### **2.4.2 Data integrity**

Information between processes must be passed without alteration. System must include functions to determine, establish and maintain the accuracy of the relationship between pieces of information. It must also include functions to ensure that when information is passed between processes, users and objects, it is possible to detect or prevent loss, addition or alteration. It should neither be possible to change the claimed or actual source and destination of the information transfer.

### **2.4.3 Authorization**

Necessary and sufficient permission shall only be granted for the intended purpose. Users and processes acting on their behalf must be prevented gaining access to information or resources that they are not authorized to access or have no need to access. Similarly, unauthorized creation or amendment of information must be prevented. System should include functions to set up and maintain any lists or rules governing the rights to perform different types of access, as well as functions concerned with temporarily restricting access to objects that are simultaneously accessible by several users or processes and are needed to maintain the consistency and accuracy of such objects.

### **2.4.4 Authentication**

It must be possible to determine and control the subjects – that is users, computers, processes and other components of the E-Settlement system – who are permitted access to resources controlled by the system. This involves not only establishing the claimed identity of a subject, but also verifying that the subject is indeed who the subject has claimed to be. This is done by the subject providing the system with some information, token or physical characteristic that is known by the system to be associated with the subject in question. It should include functions to enable new subject identities to be added,

and old subject identities to be removed. Similarly, system should include functions to change the authentication information required to verify the identity of a particular subject.

### **2.4.5 Non-repudiation**

Party to a session or transaction must not be able to make a false claim as to whether or not the exchange took place.

### **2.4.6 Data confidentiality**

Any information must not be made available or disclosed to unauthorized individuals, entity or processes. Sensitive information must be protected against unauthorized access of external people as well as against risks coming from internal personnel. Methods to limit rights to perform different types of access or to masquerade information should be developed.

### **2.4.7 Auditability**

All the main events that might represent a threat to security must be recorded. Information about actions performed by users or processes acting on their behalf shall be recorded so that the consequences of those actions can later be linked to the user in question, and the user held accountable for his actions. Sufficient information must be recorded about both routine and exceptional events that later investigations can determine if security violations have actually occurred, and if so what information or other resources were compromised. System should include functions related to the collection, protection and analysis of such information.

## **2.5 Security responsibilities**

### **2.5.1 Bank's responsibilities**

A bank participating in E-Settlement is required to:

- Protect software and hardware modules of E-Settlement according to regulations set by central banks.
- Execute its operating procedures in accordance with the regulations.
- Build interface to other systems needed to connect to E-Settlement environment. Interface must fulfill all the requirements and follow the regulations and policies set in the E-Settlement environment.
- Ensure that every user connecting to E-Settlement module is properly authenticated. Authenticated credentials have to be attached to every transaction sent to E-Settlement module.

### **2.5.2 Central bank's responsibilities**

A central bank participating in E-Settlement is required:

- Set regulations for banks in co-operation with other central banks.
- Prepare and develop principles, instructions and practices relevant to security and ensures that the security objectives pay attention to altering risks, technical evolution, regulations and standards.

Notice: Bank's responsibilities apply to a central bank as well.

### **2.5.3 Developer's responsibilities**

A developer developing the E-Settlement environment is required to:

- Follow the security policies and instructions in accordance with best practice when developing the E-Settlement system. Any potential risk discovered or suspected in any stage of development must be promptly brought to project management's knowledge.

### **2.5.4 System security auditor's responsibilities**

A system security auditor of E-Settlement environment is required to:

- Support and coordinate the information security processes related to E-Settlement environment.
- Monitor that security policies and instructions are adhered to. This happens usually simultaneously with reviews and audits.

## **2.6 Incident handling**

Security incidents must be reported through management channels as quickly as possible.

A formal reporting procedure must be established, together with an incident response procedure, setting out the action to be taken on receipt of an incident report. All participants must be made aware of the procedure for reporting security incidents, and shall be required to report such incidents as quickly as possible.

## 3 E-SETTLEMENT SECURITY ANALYSIS

### 3.1 Objectives

The purpose of this part of security work is to analyze the potential security risks in the E-Settlement environment. In addition to that, the essence is to find the safeguards to prevent security incidents.

The main elements of E-Settlement system are analyzed individually as they have specific threats, which may be insignificant in other elements or h. The elements considered individually are Bank E-Settlement module (BEM), Central Bank E-Settlement module (CEM) and System Administration Site (SAS).

### 3.2 Security measures

Security features in E-Settlement system are designed to safeguard the integrity, authenticity and confidentiality of critical data and processes, as well as to protect against losses due to fraudulent duplication or repudiation of transactions.

Security measures can be grouped into several categories based on whether the measure is designed primarily to prevent, detect or contain threats. The primary objective of measures categorized here as preventive is to ensure that attacks on components of the system will be thwarted before a fraudulent transaction can be executed. Detection measures are those taken to alert the issuer or system operator to an occurrence of fraud and to identify the source of the fraud. Containment measures are intended to limit the extent of any fraud once it has been committed. Of course, measures to detect and contain fraud may also have an important deterrent function and thus serve to prevent fraud as well. In addition, certain security measures, notably cryptographic techniques, are critical throughout the stages of prevention, detection and containment.

#### 3.2.1 Prevention measures

##### **Tamper-resistance of devices**

Special arrangements are needed for securing critical information, as part of hardware is to be located in premises of each participant. In terms of hardware, one must also make sure cabling ("short cabling") and other arrangements are at required level.

There are at least two different types of tamper resistant devices, which could be used

- chip cards with secure reader
- special boards for storing critical information (to be plugged in PCI slot or attached on peripheral bus)

Typically these devices offer secure storage of data, key generation inside the device and random number generation.

In planning phase it is important to recognise that there cannot be copies of tamper resistance devices with the same keys. Instead the system has to be constructed with parallel devices with different key sets in order to fill the high availability requirements.

##### **Public key infrastructure (cryptography)**

Public Key Infrastructure (PKI) offers tools for certificate-based authentication, integrity control and encryption. All these are key elements in securing a distributed computing model.



Implementing trustworthy PKI infrastructure is a major effort. Technology as such is fairly straight forward, but the effort is in defining and implementing policies and practices. It is also to be considered who should act as a certificate authority in E-Settlement system.

There is an ongoing discussion in maturity and interoperability of PKI products. Anyhow it is fair to say, that there are good examples such as SWIFT using PKI already today. In fact the SWIFT PKI features will be part of the E-Settlement security features in the probable case that SWIFT will be the telecommunication provider (that is the network will be based on SWIFTNet). The E-Settlement prototype itself is also an example of successful PKI-implementation using interoperable tools.

Cryptography is used to authenticate transactions and devices and to protect data confidentiality and integrity.

#### **Online authorization**

Online authorisations from central banks are needed for the execution of exceptional functions or payment flows (eg larger than the pre-set control limits).

#### **Redundant systems**

System architecture has to be planned so that there are no single-point-of-failure components for critical processes.

#### **Secure TCP/IP network**

It is essential to take care of communications security, as E-Settlement concept is distributed. The secured TCP/IP-network is preferably established and controlled by one telecommunication provider giving complete services for all communication needs. Example of this type of services is SWIFTNet, which also has good coverage among potential participants.

To secure TCP/IP network for E-Settlement concept, one should use dedicated network for application. Even in a dedicated network, there is a requirement for authentication and encryption for network traffic with means of VPN, IPSec and PKI. In case of using value added service network service (such as SWIFTNet) attention is to be paid in this area in co-operation with service provider.

#### **Server hardening**

Even though E-Settlement concept is based in use of standard components, it does not mean components are used without any changes in their configuration. Operating system and other system software must be checked and fixed against known vulnerabilities. Fixes can be based on vendor patch or changes in system setups.

#### **Defined procedures and user policies**

Security is an important part of defining operating procedures for running E-Settlement. Some of operating procedures can be forced by software, but there are also procedures that are based on agreements. Part of procedures is also personnel security including nominated persons for certain procedures, background screening and 4-eyes principle (either procedural or enforced) to be used for critical functions.

#### **Access control**

The E-Settlement module of banks will be accessed by the banks and by the central banks as well as the system administrator. There should be a secure PKI-based access control and record in the log of all successful and unsuccessful accesses. All unsuccessful access trials are automatically reported to the central bank in charge. There should be an access profile defining the access rights of different "users". Banks should be able to define the access rights within their own organization.

### **3.2.2 Detection measures**

#### **System controls for forcing procedures and detection of misbehavior**

System controls in E-Settlement software components (that is payment component, liquidity

component, control and user profile components, IT system control component) can be used for forcing operating procedures and detection of misbehavior. In further specification of software structure it should also be considered whether a separate software component is needed for managing security (that is security component).

#### **Transaction traceability and monitoring**

Audit trails and transaction logs are part of security control forming its own layer. Extra attention has to be paid to make sure this information can not be modified. Use of WORM (Write Once Read Many times) media and suitable control are needed. Procedures for writing audit trail and log information should be included in application software components. The log information should at least have digital hash signatures to hinder modification, and all critical parts should be encrypted.

All transactions will have unique audit trail numbers that identify them. The transactions are numbered in sequences and a missing number should immediately start the process of finding the reason for a missed transaction in the sequence.

There should be the possibility to reconcile the liquidity (= to check that the issued liquidity is the liquidity in circulation) of the system automatically at any moment. For this purpose consolidation breaks (end-of-day, predefined during the day or ad hoc during the day) are defined. All transactions belong to a consolidation period i.e. the time between two consolidation breaks. This information is carried as data with each transaction. When the occasion for a consolidation break is reached the E-Settlement modules are informing the central site about their liquidity situation at the end of the consolidation period and the payment turnover for the period. This can be done some minutes after the consolidation break when all transactions in progress have been processed. The central site should be able to initiate ad hoc consolidations during the day when required.

#### **Interaction with a central system**

Online interaction with central site allows the central operator to check security parameters for consistency, to update security measures on the device, such as cryptographic keys, and, in some cases, to gather additional transaction data from the device. The transaction log and records of any errors or incomplete transactions can be read and stored by the central system. Such measures increase the probability that any attempted fraud will be detected within a short period.

#### **Statistical analysis**

It is possible to implement procedures to analyze system-level data on payment flows in order to detect unusual volumes of payments that could be indicative of fraud. Issuers or a central system may utilize the automated procedures for pattern recognition to detect abnormal activity, such as those using artificial intelligence and neural network techniques. At the highest level, the system can track the volume of balances issued and redeemed each day; any level of redemption outside the norm could trigger more detailed investigations and online authorisations.

Statistical analysis procedures require the accumulation of a database covering normal payment activity over a given period. These data could be analyzed for payment patterns, taking into account seasonal patterns and differences across geographic locations. It is not clear, at this stage, how effective statistical analyses will be for detecting specific instances of fraud, or how difficult it would be for sophisticated attackers to disguise their activity within these normal payment patterns. The preventive measures in usage will most probably be so strong that there are very little to detect by statistical methods.

### **3.2.3 Containment measures**

#### **Limits on transferability**

Every participant will be assigned a user profile, which controls and limits his potential usage of the e-settlement system. These are limits on transaction volumes, balances, transaction types, transaction frequency etc. The user profile should have two different limits, that is warning limits and stop limits. When participants reach a warning limit the central bank in charge of that participant receives a warning alert to its control system. The central bank starts to analyse the reason before the participant

hits the stop limit. When the stop-limit is reached the processing in the E-Settlement module will be stopped.

Expiration dates on devices and on value also serve to contain the extent of any fraud, as a fraudulently altered device would only be usable for a limited period. Importantly, such measures may also be used to force the user to interact with the central system, where fraud could be more easily detected. The devices will generally contain limits on the maximum number of transactions, values etc that a particular device can perform before an online connection/authorisation has to be done.

#### **Registration of devices**

Registration of the identity and address of the holders of devices with the issuer or central authority would facilitate investigation of any attempted fraudulent activity.

#### **System suspension**

It is possible to implement facilities to rapidly change the cryptographic keys or revoke parties in E-Settlement network. These operations prevent potentially compromised part of operation to be misused.

### **3.2.4 Organizational measures**

#### **Development process control**

Strict manufacturing and software development procedures are implemented. If programmers work as independent units, bad or malicious code could be copied into the source code with malicious or fraudulent intent. Software under development can become confused with operational software and potentially disrupt live operations.

#### **Auditing**

The system must be audited regularly and especially when changes are made.

#### **Segregation of duties**

The development, initialization and personalization processes are strictly controlled and carried out by different organizations. Inside these organizations, there is separation of staff responsibilities. Responsibilities are clearly defined. Security evaluation of devices, components and procedures is carried out by third parties.

## **3.3 Classification of threats**

The threats are divided into seven main categories, which are not exclusive. The categories are:

- Availability,
- integrity,
- authorization,
- authentication,
- non-repudiation,
- confidentiality and
- auditability.

These categories are explained more in detail in the security policy. In addition to the above main categories, the threats are classified based on whether the source of the threat is external or internal or whether the threat is a result of an intentional or unintentional action.

### **3.4 Analysis**

Analysis contains following parts:

1. Common E-Settlement threats
2. Bank E-Settlement Module specific threats
3. Central Bank E-Settlement Module specific threats
4. System Administration Site specific threats

# Common E-Settlement threats

Threat	Applicable for information security concept										Measures		
	Availability	Integrity	Authorization	Authentication	Non-reputation	Confidentiality	Audability	Source	Intentional Prevention	Detection	Containment	Organizational	
	x	x	x	x	x	x	x	int/ext	Redundancy	Poling, software procedural control	Suspension	Network management	
Communication failure	x	x					x	int/ext	Redundancy	Poling, software procedural control	Suspension	Network management	
Hardware fault	x	x				x		int	Redundancy (cross border redundancy possible), ISO certification, clustering, UPS	OS alerts, visual control, software procedural control	Suspension	On-site replacement plan, network management, facilities management	
Insufficient hardware performance	x							int	Stress testing, load balancing	OS alerts, forced control			
Virus, Trojan horse	x	x	x	x	x	x	x	int/ext	Virus protection, dedicated devices	Virus protection, IDS			
Environmental problems (fire, flood, etc)	x	x					x	ext	Redundancy, DSS	fire alarm	Fire control, physical firewall		
Terror	x	x					x	ext	Redundancy, DSS, physical security	Monitoring	Physical security		
Denial of service (DOS)	x							ext	Dedicated network, interface mechanisms	Forced control (awareness, monitoring tool)	Procedural control (fall back solution), stop payment processing		
In-house software failure	x	x	x			x		int	ISO certification minimizes risk (ISO 9000, BS 7799), fault tolerant code	Testing	Error recovery	Error reporting and correction procedure, change management	
Vendor software failure	x	x	x			x		int	ISO certification, reliable vendors, combat proven	Testing	Error reporting, corrective services, change management	On-site replacement plan, segregation of duties, 4-eyes control	
Manipulation of software	x	x	x	x	x	x	x	int/ext	Access control, tamper proof devices	PKI, forced control, IDS			
Man-in-middle	x	x	x	x	x	x	x	int/ext	Dedicated network, PKI	PKI, software procedural control		PKI policy, key management	
Unauthorized user	x	x	x	x	x	x	x	int/ext	PKI (digital certificates)	PKI		PKI policy, key management	
Misuse of remote access	x							int/ext	PKI, access control	Forced control			
Theft of devices	x	x						int/ext	Physical security		PKI, suspension		

## Common E-Settlement threats continued

Threat	Applicable for information security concept							Measures				
	Availability	Integrity	Authorization	Authentication	Non-repudiation	Confidentiality	Audability	Source	Intentional Prevention	Detection	Containment	Organizational
Theft of product design documents						x		int/ext x	Physical security, encrypted electronic documents			Document classification, duplication
Theft or compromising of cryptographic keys									Tamper resistant devices, physical security, frequent invalidation of keys			
Reverse engineering	x	x	x	x	x	x	x	int/ext x	Forced control			4-eyes control
Social engineering			x	x				int/ext x	System monitoring	Algorithm replacement		
Duplication of devices								int/ext x	Segregation of duties, personnel security			
Lack of experience	x	x	x	x	x	x	x	int/ext x	Tamper resistant devices, FIPS standards	Forced control, physical security		
Immaturity of technologies	x	x	x	x	x	x	x	int	Education	Forced control		Partnering
								int	Testing			
Platform attractive for hacking									Private networks, high level of confidentiality, PKI, software service, server hardening			
False timestamp information	x	x	x	x	x	x	x	int/ext x	Intrusion detection system			
Inconsistent data								int	PKI, clock synchronizing	Forced control		
Transaction confidentiality	x	x						int/ext	Transaction mgmt, replication mechanism	Forced control	Recovery from transaction logs	
			x					int/ext	Private networks, high level of confidentiality, PKI	PKI, Forced control		

## Bank E-Settlement module specific threats

Threat	Applicable for information security concept										Measures		
	Availability	Integrity	Authorization	Authentication	Non-repudiation	Confidentiality	Audability	Source	Intentional Prevention	Detection	Containment	Operational	
BAPS communication failure	x	x				x	x	int/ext	Redundancy	Polling	Inform SAS		
ESN communication failure	x	x				x	x	int/ext	Redundancy	Polling	Stop sending, shutdown		
Manipulation of data						x	x	int/ext	PKI, (enforced 4-eyes control)	PKI, forced control	Inform SAS, stop sending		
Manipulation of audit trail or transaction log						x	x	int/ext	WORM, PKI, hardened log server	PKI, forced control	Inform SAS, stop payment processing		
Duplicated transaction data						x	x	int	Unique transaction identifiers	forced control			
Reputiate settlement transaction								int	Digital signature, transaction logs	forced control	Inform SAS, stop sending		
Incorrect BAPS behavior	x	x						int	BAPS certification testing, fault tolerance	forced control	Inform SAS, stop payment processing	Reporting mechanisms	

## Central Bank E-Settlement module specific threats

Threat	Applicable for information security concept										Measures		
	Availability	Integrity	Authorization	Authentication	Non-reputation	Confidentiality	Audibility	Source	Intentional Prevention	Detection	Containment	Operational	
CBPS communication failure	x	x				x	x	int/ext	Redundancy	Polling	Inform SAS		
ESN communication failure	x	x				x	x	int/ext	Redundancy	Polling	Stop sending, shutdown		
Manipulation of data		x				x	x	int/ext	PKI	PKI, forced control	Inform SAS, stop sending		
Manipulation of audit trail or transaction log		x				x	x	int/ext	WORM, PKI	PKI, forced control	Inform SAS, stop payment processing		
Duplicated transaction data		x				x	x	int	Unique transaction identifiers	Forced control			
Reputate settlement transaction					x			int	Digital signature, transaction logs	Forced control	Inform SAS, stop sending		
Incorrect CBPS behavior	x	x				x		int	CBPS certification testing, fault tolerance	Forced control	Inform SAS, stop payment processing	Reporting mechanisms	
Incorrect bank control information	x	x				x	x	int		Forced control			
Incorrect funds data	x	x						int	PKI, transaction logs	Forced control	Inform SAS		



## SAS specific threats

Threat	Applicable for information security concept							Measures				
	Availability	Integrity	Authorization	Authentication	Non-repudiation	Confidentiality	Audibility	Source	Intentional Prevention	Detection	Containment	Organizational
ESN communication failure	x	x					x	int/ext	Redundancy	Polling	Stop sending, shutdown	
Manipulation of data		x						int/ext	PKI	PKI, procedural control	Human intervention	
Manipulation of audit trail or transaction log		x			x	x	x	int/ext	WORM, PKI	PKI, procedural control	Human intervention	
Duplicated transaction data		x					x	int	Unique transaction identifiers	Procedural control	Human intervention	
False system control information					x	x	x	int	Tamper resistant devices, PKI, replication	PKI		4-eyes control
Incorrect staff behavior	x	x	x	x	x	x	x	int	Psychological tests, high salary, background checking	Human control, procedural control		Dual control

---

## 4 E-SETTLEMENT SYSTEM SECURITY SPECIFICATION

### 4.1 Introduction

System security specification is a part of the Security Management framework (BS7799) utilized for keeping the security of E-Settlement system valid in a changing environment. First a policy was specified in E-Settlement Security Policy, after this current risks were analysed

This part of section presents in detail the E-Settlement specific security controls. More widely used security technologies utilized in E-Settlement are presented on a general level only. The potential measures are mentioned in risk analysis. The application of these technologies to E-Settlement system will be specified in detail later.

#### 4.1.1 Utilized technologies

The E-Settlement security solution utilizes a number of standard security technologies:

- Public Key Infrastructure (PKI), see [RFC 2459]
- IP Security Protocol (IPsec), see [RFC 2401], [RFC 2402], [RFC 2406] and [RFC 2409]
- Secure Sockets Layer (SSL), see [RFC 2246]
- Hardware Security Modules (HSM), see [FIPS 140-1] and [ISO 13491-1]
- Hybrid cryptographic algorithms (i.e. combination of public key secret key algorithms)
- Write Once Read Many times (WORM) transaction logs

#### 4.1.2 Structure

First, the facets of E-Settlement security are presented. Then the main features of E-Settlement security are presented in detail: the structure of the E-Settlement stamps is presented, the structure of the transaction logs is introduced and HSM interface is described on API level.

## 4.2 Facets of E-Settlement security

Different parts of security solutions are needed in different parts of the E-Settlement system. The Figure 1 presents the facets of E-Settlement security. See [ESET SYSTEM] for more information about the Security Architecture of the E-Settlement system.

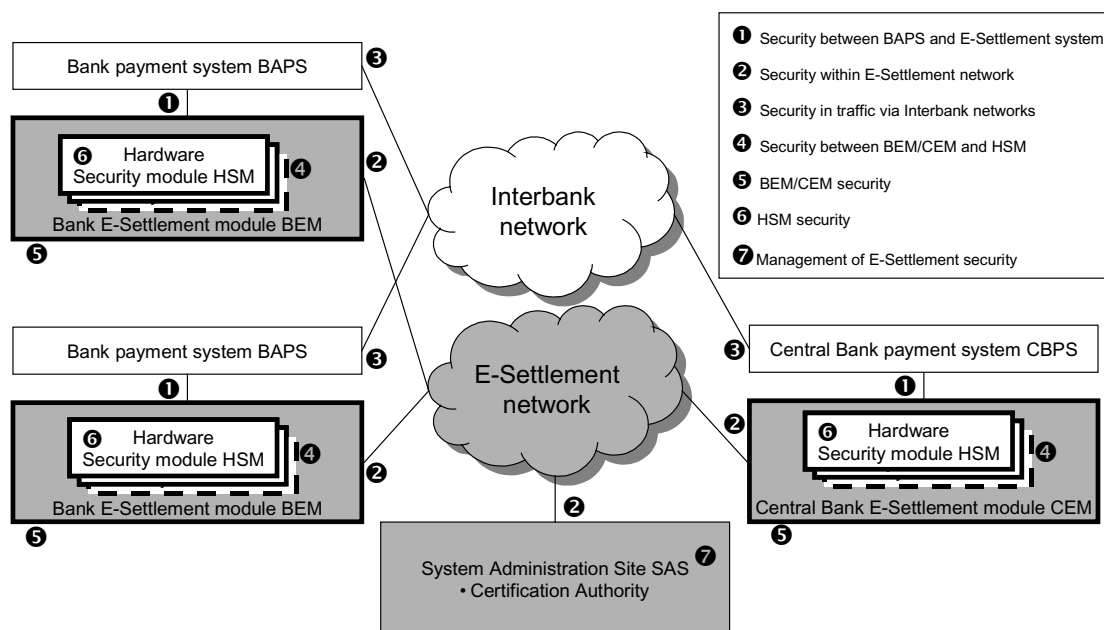


Figure 1. Facets of E-Settlement security

#### 4.2.1 Security between Bank Payment System and the E-Settlement System

The secure connection between bank payment system and E-Settlement module is principally secured physically. The bank that hosts a module has a responsibility to ensure that traffic between its payment system and the E-Settlement module is secure, therefore a suitable placement for the E-Settlement module is in the bank data center.

Communication confidentiality between BAPS and BEM is achieved using SSL. The payment system authenticates the E-Settlement module using the module's E-Settlement certificate. After the payment system has authenticated the E-Settlement module the connection is encrypted for confidentiality.

The E-Settlement authenticates payment system's users and applications after the encrypted SSL connection has been established. Authentication is done against identification and password. After successful authentication the client is authorized to one or many services provided by E-Settlement module via bank interface. To ensure auditability all events in the E-Settlement module transaction log are stored with user information.

#### 4.2.2 Security within E-Settlement Network

The E-Settlement network (ESN) is a dedicated IP network, with high requirements for physical protection. ESN confidentiality, non-repudiation, authentication and integrity are achieved using IPsec. The public key management of IPsec is founded on PKI.

#### 4.2.3 Security in traffic via Interbank networks

The confidentiality, authentication and integrity of the ESTAMP are ensured with cryptographic algorithms, since the ESTAMP is transferred through networks and elements that are considered untrusted from the E-Settlement system point of view. The ESTAMP consists of four separate parts. All of these parts are signed to enable integrity verification. Two of these parts are in clear text and two are encrypted to ensure confidentiality, the encrypted parts are signed with means that enable origin authentication.

For the encrypted parts the hybrid crypto-algorithm is used. The format and transportation of ESTAMP is described in more detail in chapter 4.3.

The format of the Bank Network (BNET) messages is not discussed in this document. It has no effect on ESTAMP tunnelling as long as BNET messages can handle fields for additional information that are sufficiently long.

#### **4.2.4 Security between BEM/CEM and HSM**

Chapter 4.5 specifies an interface between E-Settlement module and HSM. HSM always returns encrypted information so that only the receiving HSM can decrypt it.

#### **4.2.5 BEM/CEM security**

The E-Settlement module is principally secured physically and using organizational procedures. The bank hosting the module protects the module from physical threats. E-Settlement system enforces policies that limit local and remote access to the E-Settlement module to authorized personnel only.

The E-Settlement module is “server hardened”, which means that it is stripped of all unnecessary services. The E-Settlement module also has programmed controls that monitor the behavior of the module and reports all unusual events to SAS. All transactions that the E-Settlement module executes are logged in transaction log. Transaction logging is discussed in more detail in chapter 4.4.

Each E-Settlement module has a public key pair that is dedicated for encrypting and decrypting messages and one public key pair that is dedicated for creating and verifying signatures.

#### **4.2.6 HSM security**

The E-Settlement module HSMs are capable of signing and verifying signatures, encrypting and decrypting messages both with public and secret key algorithms, and using PKCs. also checking their validity. In E-Settlement system, the HSMs store independently of E-Settlement modules the liquidity balance, outstanding liquidity amount and a counter of sent transaction

Each E-Settlement module has one primary HSM and one or more secondary HSMs, which all share the same public key pair that is dedicated for encrypting and decrypting messages. Each HSM also has an own public key pair that is dedicated for creating and verifying signatures.

#### **4.2.7 Management of E-Settlement security**

The E-Settlement system security management is handled by the SAS. In the prototype the SAS Certificate Authority (CA) handles the PKI infrastructure and SAS IPsec server handles the IPsec profiles.

The E-Settlement requires CA and Trusted Third Party (TTP) functions. The maintained Certificate Revocation List (CRL) is updated periodically and upon request. The issued certificates are copied into SDI and can be accessed by other elements using Lightweight Directory Access Protocol (LDAP). The SAS also stores all the public key pairs (also the secret key) it has generated securely.

The HSM public key pair that is dedicated for encryption and decryption purpose is generated and stored by CA, since the same key pair is used in all HSMs used by one single E-Settlement module. This also gives the SAS a possibility to read HSM encrypted stamp parts. This key pair generation, registration and certificate issuing models shall be defined in Certificate Policy and Certificate Practise Statement.

The HSM public key pair that is dedicated for creating and validating signatures is generated inside a HSM and no other element has access to the private key, not even the other HSMs within the same E-Settlement module. This key pair generation, registration and certificate issuing models shall be defined in Certificate Policy and Certificate Practise Statement.

The E-Settlement module also has separate public key pairs for encrypting and signing. Both key pairs are generated by SAS CA and stored by SAS.

### 4.3 E-Settlement Stamp

This chapter describes the structure and use of E-Settlement Stamps (ESTAMP) and E-Settlement Confirmation Stamps (ECSTAMP). The ESTAMP and ECSTAMP consist of four parts:

- **BAPS\_ESTAMP/ECSTAMP:** A plaintext part for information received from BAPS. This is used by BAPSEs to verify that the payment information, based on which settlement stamps have formed, is correct.
- **BEM\_PL\_ESTAMP/ECSTAMP:** A plaintext part of BEM information. This is used in situations where the encrypted BEM part cannot be decrypted to find out the possible cause of the problem.
- **BEM\_CR\_ESTAMP/ECSTAMP:** An encrypted part for BEM information. The information in this part is used for actual BEM settlement processing. Some of the fields (such as liquidity balance) are present for disaster recovery purposes and not used for actual settlement processing.
- **HSM\_ESTAMP/ECSTAMP:** An encrypted part for HSM information. This part is used to check the validity of BEM information.

The stamp parts are interlinked so that some items from BAPS part are included in the hash value of the encrypted BEM part, which is again included in the encrypted HSM part.

### 4.3.1 E-Settlement Stamp format

#### 4.3.2 Fields that originate from BAPS (BAPS\_ESTAMP)

Field name	Size bytes	Field description
BAPS_STAMP_VER	1	The version number of the BAPS_ESTAMP fields. Content of BAPS_ESTAMP fields may change in future.
BAPS_ID_SEND	10	The sending BAPS_ID.
BAPS_ID_REC	10	The receiving BAPS_ID.
BAPS_ID_TRANS	34	A transaction id provided by the BAPS.
BAPS_TIMESTAMP	8	A timestamp representing milliseconds since beginning of year 1970, time when request from BAPS to BEM was made.
BAPS_PAY_ACC	34	The payer's account number.
BAPS_BENE_ACC	34	The beneficiary's account number.
BAPS_SUM	16	The amount of liquidity being transferred, IEEE 128 bit float.
BAPS_REPEAT	2	The code identifying how many times this payment has been sent previously from BAPS.
BAPS_HASH	20	A signature calculated by BAPS.

#### Plaintext fields that originate from BEM (BEM\_PL\_ESTAMP)

Field name	Size bytes	Field description
BEM_STAMP_VER	1	The version number of the BEM_PL/CR_ESTAMP fields. Content of BEM_PL/CR_ESTAMP fields may change in future.
BEM_ID_SEND	10	The sending BEM_ID. The id can be used to look up the PKC certificate of the sending BEM in SDI.
BEM_ID_REC	10	The receiving BEM_ID.
ES_ID_TRANS	20	An identifier that uniquely identifies a payment transaction in the whole E-Settlement System. This identifier is constructed using the SHA hash algorithm with a timestamp, and sending and receiving BEM ids as input for the hash.
BEM_REPEAT	2	The code identifying how many times this payment has been sent previously from BEM.
BEM_TIMESTAMP	8	A timestamp representing milliseconds since beginning of year 1970, time when ESTAMP was created by BEM.

**Encrypted fields that originate from BEM (BEM\_CR\_ESTAMP)**

Field name	Field size	Field description
BEM_SESSION_KEY	128	Encrypted symmetric session key. Before the session key is encrypted, it is used to encrypt other BEM_CR_ESTAMP fields. The session key is encrypted using the receiving BEM's public encryption key, so that only the receiving BEM can decrypt the session key.
BEM_HASH	20	A hash value that BEM calculates using the BAPS_PL_ESTAMP fields and the BAPS_HASH field as input.
BEM_SUM	16	The amount of liquidity being transferred, IEEE 128 bit float.
BEM_BALANCE	16	The current liquidity balance of the sending BEM, IEEE 128 bit float.
BEM_ID_SEND	10	The sending BEM_ID.
BEM_ID_REC	10	The receiving BEM_ID.
BEM_TIMESTAMP	8	A timestamp representing milliseconds since beginning of year 1970, time when ESTAMP was created by BEM.
BEM_CUMUL	16	The cumulative bank-to-bank amount of confirmed transactions that has been transferred during the day from sending to receiving BEM, IEEE 128 bit float.
HSM_ID_SEND	10	The HSM_ID for the device that is in use when this stamp is created. This information is needed to find the HSM PKC from SDI (the PKC containing the signature verification key).
BEM_AUDIT_NUMBER	8	Transaction's bank-to-bank audit trail number for the day, starting from zero and increasing monotonously on bank-to-bank level.
BEM_PERIOD_NUMBER	4	Transaction's reconciliation period number within the day. The number starts from zero and increases monotonously.
BEM_SIGN	128	All other fields in this stamp part except BEM_SESSION_KEY are signed using the BEM's private signature key. The resulting signature is inserted here.

**Encrypted fields that originate from the HSM (HSM\_ESTAMP)**

Field name	Field size	Field description
HSM_SESSION_KEY	128	Encrypted symmetric session key. Before the session key is encrypted, it is used to encrypt other HSM_ESTAMP fields. The session key is encrypted using the receiving BEM HSMs' shared public encryption key, so that only the receiving BEM HSMs can decrypt the session key.
HSM_STAMP_VER	1	The version number of the HSM_ESTAMP fields. Content of HSM_ESTAMP fields may change in future.
BEM_HASH	20	A hash value that BEM calculates using the BAPS_PL_ESTAMP fields and the BAPS_HASH field as input.
HSM_SUM	16	The amount of liquidity being transferred according to HSM, IEEE 128 bit float.
HSM_BALANCE	16	The current liquidity balance stored in HSM, IEEE 128 bit float.
HSM_OUTSTAND	16	The amount of liquidity not confirmed to HSM (the receiver has not sent an acknowledgement), IEEE 128 bit float.
HSM_AUDIT_NUMBER	8	Transaction audit trail number for the day, starting from zero and increasing monotonously on per day level.
HSM_SIGN	128	All other fields in this stamp part except HSM_SESSION_KEY are signed using the HSM's own private signature key. The resulting signature is inserted here.

**4.3.3 E-Settlement Confirmation Stamp format**

The ECSTAMP has two functions:

1. It acknowledges the payment transaction.
2. It allows sending BEM and receiving BEM to check the payment flow to the "other direction", i.e. from receiving BEM to sending BEM.

To these ends, the acknowledging ECSTAMP contains a BEM\_STATUS field, which informs the sending BEM whether the transaction was successful or not, as well as BEM\_CUMUL and BEM\_AUDIT\_NUMBER fields, which are key figures of cumulative payment flow from receiving to sending BEM.

**Fields that originate from BAPS (BAPS\_ECSTAMP)**

BAPS\_ECSTAMP is identical with BAPS\_ESTAMP.



**Plaintext fields that originate from BEM (BEM\_PL\_ECSTAMP)**

Field name	Size bytes	Field description
BEM_STAMP_VER	1	The version number of the BEM_PL/CR_ECSTAMP fields. Content of BEM_PL/CR_ECSTAMP fields may change in future.
BEM_ID_SEND	10	The confirmation sending BEM_ID. The id can be used to look up the PKC certificate of the sending BEM in SDI.
BEM_ID_REC	10	The confirmation receiving BEM_ID.
ES_ID_TRANS	20	This transaction id is copied from ES_ID_TRANS of ESTAMP.
BEM_REPEAT	2	The code identifying how many times this confirmation has been sent previously from BEM.
BEM_TIMESTAMP	8	A timestamp representing milliseconds since beginning of year 1970, time when ECSTAMP was created by BEM.

**Encrypted fields that originate from BEM (BEM\_CR\_ECSTAMP)**

Field name	Field size	Field description
BEM_SESSION_KEY	128	Encrypted symmetric session key. Before the session key is encrypted, it is used to encrypt other BEM_CR_ECSTAMP fields. The session key is encrypted using the receiving BEM's public encryption key, so that only the receiving BEM can decrypt the session key.
BEM_CONF_HASH	20	The hash value that BEM calculates using the BAPS_PL_ECSTAMP fields and the BAPS_HASH field as input.
BEM_STATUS	1	The return status of the transaction.
BEM_SUM	16	The amount of liquidity being transferred, IEEE 128 bit float.
BEM_BALANCE	16	The current liquidity balance of the receiving BEM, IEEE 128 bit float.
BEM_ID_SEND	10	The confirmation sending BEM_ID.
BEM_ID_REC	10	The confirmation receiving BEM_ID.
BEM_TIMESTAMP	8	A timestamp representing milliseconds since beginning of year 1970, time when ECSTAMP was created by BEM.
BEM_CUMUL	16	The cumulative bank-to-bank amount of transactions that has been transferred during the day from receiving to sending BEM, IEEE 128 bit float.
HSM_ID_REC	10	The HSM_ID for the device that is in use when this stamp is created. This information is needed to find the HSM PKC from SDI (the PKC containing the signature verification key).
BEM_AUDIT_NUMBER	8	The audit-trail number of the last settlement transaction sent from the receiving BEM. This can be used by the sending BEM to check if it misses settlement transactions that receiving BEM has initialized.
BEM_PERIOD_NUMBER	4	Transaction's reconciliation period number of the last settlement transaction sent from the receiving BEM. The number starts from zero and increases monotonously.
BEM_SIGN	128	All other fields in this stamp part except BEM_SESSION_KEY are signed using the BEM's private signature key. The resulting signature is inserted here.

**Encrypted fields that originate from the HSM (HSM\_ECSTAMP)**

Field name	Field size	Field description
HSM_SESSION_KEY	128	Encrypted symmetric session key. Before the session key is encrypted, it is used to encrypt other HSM_ECSTAMP fields. The session key is encrypted using the receiving BEM HSMs' shared public encryption key, so that only the receiving BEM HSMs can decrypt the session key.
HSM_STAMP_VER	1	The version number of the HSM_ECSTAMP fields. Content of HSM_ECSTAMP fields may change in future.
BEM_CONF_HASH	20	A hash value that BEM calculates using the BAPS_PL_ECSTAMP fields and the BAPS_HASH field as input.
HSM_STATUS	1	The return status of the transaction.
HSM_SUM	16	The amount of liquidity being transferred according to HSM, IEEE 128 bit float.
HSM_BALANCE	16	The current liquidity balance stored in HSM, IEEE 128 bit float.
HSM_OUTSTAND	16	The amount of liquidity not confirmed to HSM (the receiver has not sent an acknowledgement), IEEE 128 bit float.
HSM_AUDIT_NUMBER	8	Transaction audit trail number for the day, starting from zero and increasing monotonously on per day level.
HSM_SIGN	128	All other fields in this stamp part except HSM_SESSION_KEY are signed using the HSM's own private signature key. The resulting signature is inserted here.

**4.3.4 ESTAMP creation and validation**

A BEM will have one primary HSM and one or more secondary HSMs, all of which have the same encryption key but differing signing keys. The liquidity balances of all of the HSM are updated during each operation, but to save on the slow PKI operations, only one HSM is asked for a (settlement or confirmation) stamp. If the primary HSM fails, CEM/SAS is alerted so that maintenance activities can begin. The secondary HSM is taken into "full" use by BEM, i.e. it starts to ask for stamps from the secondary BEM.

The order of operations performed in the processes is based on the idea that BEM does all of its processing first, and HSM validation happens when all BEM validation, signing and encryption has been performed.

The numbering in the figures refers to the numbering in the detailed process descriptions. These figures present an illustrative overview of the processes.

## ESTAMP creation in sending BAPS/BEM

Operation	BAPS	BEM	Primary HSM	Secondary HSM	Exceptions
2. BEM validation		presentSettlementTransaction()			Fault code returned to BAPS. SAS/CEM possibly alerted.
3. BAPS stamp creation		BEM validates the transaction against bank profile and available liquidity. BEM creates BAPS_ESTAMP.			
4. BEM stamp creation		BEM creates BEM_PL_ESTAMP and BEM_CR_ESTAMP			CEM/SAS alerted. Fault code returned to BAPS
5. HSM calls			HSMpresentSettlementTransaction()	HSMpresentSettlementTransaction()	
5.1. BEM/HSM balance validation			HSM compares BEM and HSM saldo	HSM compares BEM and HSM saldo	
5.2 HSM stamp creation			HSM creates HSM_ESTAMP	HSM decreases liquidity balance and increases outstanding balance	HSM closes itself, no more settlement processing possible. HSM returns a fault code to BEM
5.3. HSM balance update			Return value + HSM_ESTAMP	Return value	
7. BEM update		BEM decreases liquidity balance and increases BEM outstanding balance. The transaction is put to queue of unconfirmed payments. BEM bank-to-bank payment information is updated. BEM forms the ESTAMP. BEM transaction log is updated.			BEM stops payment process immediately. SAS and CEM are alerted.
9. Payment message formation	Return value + ESTAMP	ECSTAMP is sent back to sending BAPS in interbank confirmation			BAPS rolls back the payment transaction

Figure 2. Stamp handling: creation

## ESTAMP validation in receiving BAPs/BEM

Operation	BAPS	BEM	Primary HSM	Secondary HSM	Exceptions
1. BAPS validation	Receiving BAPS checks that the received payment transaction is valid.	presentReceivedSettlementTransaction()			Continue with "Rejection by receiving BAPS"
3. BEM signature check		BEM validates BEM_CR_ESTAMP signature			Alert SAS/CEM. Continue with rejection stamp creation.
4. BEM validation		BEM validates BEM hash value			Alert SAS/CEM. Continue with rejection stamp creation.
5. BEM confirmation stamp formation		BEM forms BEM_PL_ECSTAMP and BEM_CR_ECSTAMP			Alert SAS/CEM. Continue with rejection stamp creation.
6.1 BEM / HSM balance validation			HSMpresentReceivedSettlementTransaction()	HSMpresentReceivedSettlementTransaction()	
6.2 Signature validation			Compare BEM and HSM balance HSM_ESTAMP signature is validated	Compare BEM and HSM balance HSM_ESTAMP signature is validated	HSM closes itself, no more settlement processing possible. HSM returns a fault code to BEM.
6.3 HSM stamp field validation			HSM_ESTAMP fields are validated.	HSM_ESTAMP fields are validated.	
6.4 Update HSM balance			Increase HSM liquidity balance. Form HSM_ECSTAMP	Increase HSM liquidity balance.	
6.5 Form HSM stamp			Return value + HSM_ECSTAMP	Return value	
8. BEM update		BEM increases liquidity balance bank-to-bank payment information is updated. BEM forms the ECSTAMP. BEM transaction log is updated.			BEM stops payment process immediately. SAS and CEM are alerted.
9. BAPS commit/rollback		Return value + ECSTAMP Receiving BAPS commits the payment transaction.			BAPS rolls back the payment transaction
10. Interbank confirmation message	ECSTAMP is sent back to sending BAPS in interbank confirmation				

Figure 3. Stamp handling: validation

## Rejection by receiving BAPS

Operation	BAPS	BEM	Primary HSM	Secondary HSM	Exceptions
2. BEM signature check	recordReceivedSettlementIncident()	BEM validates BEM_CR_ECSTAMP signature			Alert SAS/CEM.
3. BEM validation		BEM validates BEM hash value			Alert SAS/CEM.
4. BEM confirmation stamp formation		BEM forms BEM_PL_ECSTAMP and BEM_CR_ECSTAMP			Alert SAS/CEM.
6 BEM / HSM balance validation		HSMrequestRejectedConfirmationStamp()	Form rejection HSM_ECSTAMP		HSM closes itself, no more settlement processing possible. HSM returns a fault code to BEM.
8. BEM update		BEM bank-to-bank payment information is updated. BEM forms the ECSTAMP. BEM transaction log is updated.	Return value + HSM_ECSTAMP		BEM stops payment process immediately. SAS and CEM are alerted.
10. Interbank cancellation message	Return value + ECSTAMP ECSTAMP is sent back to sending BAPS in interbank confirmation				

Figure 4. Stamp handling: rejection by receiving BAPS

## Confirmation ECSTAMP handling in sending BAPS/BEM

Operation	BAPS	BEM	Primary HSM	Secondary HSM	Exceptions
2. BEM signature check		confirmSettlementTransaction() ↑			Alert SAS/CEM. Fault coc returned to BAPS.
3. BEM confirmation check		BEM validates BEM_CR_ESTAMP signature BEM validates BEM_CONF hash value			Alert SAS/CEM. Fault coc returned to BAPS.
4.1 BEM / HSM balance validation			HSMconfirmSettlementTransaction() ↑		
4.2 Signature validation			Compare BEM and HSM balance HSM_ECSTAMP signature is validated	Compare BEM and HSM balance HSM_ECSTAMP signature is validated	
4.3 HSM stamp field validation			HSM_ECSTAMP fields are validated.	HSM_ECSTAMP fields are validated.	HSM closes itself, no more settlement processing possible. HSM returns a fault code to BEM.
4.4 Update HSM outstanding balance			Decrease HSM outstanding balance. Return value	Decrease HSM outstanding balance. Return value	
6. BEM update		BEM decreases outstanding balance. The transaction is taken of queue of unconfirmed payments. BEM bank-to-bank payment information is updated. BEM transaction log is updated.			BEM stops payment process immediately. SAS and CEM are alerted.
7. BAPS commit/rollback	Receiving BAPS commits the payment transaction. ↓	Return value			BAPS rolls back the payment transaction

Figure 5. Confirmation ECSTAMP handling

## Rejections ECSTAMP handling in sending BAPS/BEM

Operation	BAPS	BEM	Primary HSM	Secondary HSM	Exceptions
2. BEM signature check	rejectSettlementTransaction()	BEM validates BEM_CR_ESTAMP signature			Alert SAS/CEM. Fault code returned to BAPS.
3. BEM confirmation check		BEM validates BEM_CONF hash value			Alert SAS/CEM. Fault code returned to BAPS.
4.1 BEM / HSM balance validation			HSMrejectSettlementTransaction()		
4.2 Signature validation			Compare BEM and HSM balance	Compare BEM and HSM balance	
4.3 HSM stamp field validation			HSM_ECSTAMP signature is validated	HSM_ECSTAMP signature is validated	HSM closes itself, no more settlement processing possible. HSM returns a fault code to BEM.
4.4 Update HSM outstanding balance			HSM_ECSTAMP fields are validated. Increase liquidity balance and decrease HSM outstanding balance. Return value	HSM_ECSTAMP fields are validated. Increase liquidity balance and decrease HSM outstanding balance. Return value	
6. BEM update		BEM increases liquidity balance and decreases BEM outstanding balance. The transaction is taken of queue of unconfirmed payments. BEM bank-to-bank payment information is updated. BEM transaction log is updated.			BEM stops payment process immediately. SAS and CEM are alerted.
7. BAPS commit/rollback	Receiving BAPS commits the payment transaction.	Return value			BAPS rolls back the payment transaction

Figure 6. Rejection ECSTAMP handling



## 4.4 Transaction log

To ensure auditability, all transactions that E-Settlement module executes are logged in a transaction log. The transactions are logged on a Write Once Read Many times (WORM) device to ensure that the information cannot be overwritten or altered at a later stage and all messages are signed by the E-Settlement module to enable log writer authentication.

Only SAS has full right to read all information that is logged, therefore the hybrid crypto-algorithm is used to enable authorization and confidentiality. A transaction log row can contain plaintext information that SAS, CEM and BEM may read, information that SAS and CEM may read, and information that only SAS may read. ESTAMPs and ECSTAMPs are stored in the log as plaintext, since the confidential information is already encrypted in these. Figure 7 shows the transaction log design.

SK1 is the BEM's public signature key and K is the session key generated for log message encryption. The first column contains transaction information that is signed with SK1 and possibly encrypted with a random session key K. The second column contains the session key encrypted with the SAS public key, if the logged message is not plaintext. The third column contains the session key encrypted with the CEM public key, if CEM should be able to read the information in the first column and if the message is not plaintext.

Logged information	SAS	CEM
$E_{K_1}(S_{SK1}(M_1))$	$E_{PKCAS}(K_1)$	$E_{PKNEM}(K_1)$
$E_{K_2}(S_{SK1}(M_2))$	$E_{PKCAS}(K_2)$	
$E_{K_3}(S_{SK1}(M_3))$	$E_{PKCAS}(K_3)$	$E_{PKNEM}(K_3)$
$S_{SK1}(M_4)$		
$E_{K_5}(S_{SK1}(M_5))$	$E_{PKCAS}(K_5)$	
$E_{K_6}(S_{SK1}(M_6))$	$E_{PKCAS}(K_6)$	$E_{PKNEM}(K_6)$
$S_{SK1}(M_7)$		
$E_{K_8}(S_{SK1}(M_8))$	$E_{PKCAS}(K_8)$	
.....	.....	.....
.....	.....	.....

Figure 7. Transaction log

The hybrid crypto-algorithm is used to create a process for transaction log message writing:

1. BEM signs the message  $M$  with its private key  $SK1$ . If the inserted message is plaintext the signed message is inserted into the first column and the log writing process ends here.

$$S_{SK1}(M)$$

- 
2. BEM generates a random session key  $K$  and encrypts the signed message with the random session key.  
 $E_K(S_{SKI}(M))$
  3. BEM encrypts the generated session key with the SAS public key  $PK_{SAS}$ .  
 $E_{PK_{SAS}}(K)$
  4. BEM encrypts the generated session key with the CEM public key  $PK_{CEM}$ , if CEM should be able to read the message.  
 $E_{PK_{CEM}}(K)$
  5. BEM inserts the encrypted and signed message into the first column of the transaction log.  
 $E_K(S_{SKI}(M))$
  6. BEM inserts the session key that was encrypted with SAS public key into the second column of the transaction log.  
 $E_{PK_{SAS}}(K)$
  7. BEM inserts the session key that was encrypted with CEM public key into the third column of the transaction log, if CEM should be able to read the message from transaction log.  
 $E_{PK_{CEM}}(K)$

When SAS, CEM or BEM reads from the transaction log the process is as follows:

1. The column that contains the encrypted session key is checked. SAS checks the second and CEM checks the third column. If the message is only signed the BEM the message can directly be verified and read by all parties, if this is the case the log reading process ends here. If the message is encrypted, the session key has to be obtained from the columns two or three. In the rest of this process description it is assumed that SAS is trying to read the transaction log, the rest of this process is equivalent for SAS and CEM instances.
2. The column content is decrypted with the own private key  $SK_{SAS}$  and session key  $K$  is obtained.  
 $D_{SK_{SAS}}(E_{PK_{SAS}}(K))=K$
3. The session key is used to encrypt the signed message.  
 $D_K(E_K(S_{SKI}(M)))=S_{SKI}(M)$
4. Finally the SAS can verify that the message is signed by BEM and read the logged message.

---

## 4.5 HSM Interface

### 4.5.1 General HSM API

This API contains general HSM functions.

#### HSMopen

Syntax: `HSMopen (PIN)`

Description: E-Settlement module calls this function upon command from controlling system element (CEM for BEM, SAS for CEM). The controlling element provides the *PIN*. If this function was successful the E-Settlement module can begin using the HSM.

Returns: a status message describing the validity of the PIN code for this HSM.

#### HSMclose

Syntax: `HSMclose ()`

Description: After this function has been called a new open-call is needed to continue using other HSM functions.

Returns: indication of success or failure.

#### HSMupdateCRL

Syntax: `HSMupdateCRL (CRL)`

Description: Update HSM CRL. Parameter *CRL* is X.509v2 CRL.

Returns: indication of success or failure.

#### HSMreadInfo

Syntax: `HSMreadInfo ()`

Description: This function is used by SAS when a HSM has been replaced with a new one. First, HSM reads the core HSM info (liquidity balance, outstanding balance, HSM transaction number) from the still functioning HSM using this method, and the info is then written to the new HSM using `HSMwriteInfo`.

Returns: a record containing HSM liquidity balance, outstanding balance and HSM transaction number, signed by the HSM and encrypted for SAS.

#### HSMwriteInfo

Syntax: `HSMwriteInfo (encryptedHSMinfo)`

Description: Updates HSM information (liquidity balance, outstanding balance, HSM transaction number). This function is generally used by SAS when a HSM has been replaced (see **Virhe. Viitteen lähde ei löytynyt.**). Parameter *encryptedHSMinfo* contains the HSM information in the order described, signed by SAS and encrypted for HSM.

Returns: indication of success or failure.

---

## 4.5.2 Sending HSM API

This API contains HSM functions needed when for payment transaction on the sending E-Settlement module side.

### HSMpresentSettlementTransaction

Syntax: `HSMpresentSettlementTransaction(sum, BEM_liqbalance, BEM_outbalance, receiving_HSM_encrypt_PKC, BEM_hash, stamp_requested)`

Description: This function is called when the E-Settlement module is creating the ESTAMP and needs the HSM contribution. The HSM decreases its liquidity balance, increases the outstanding balance and creates the signed HSM\_ESTAMP stamp. PKC parameter is in X.509v3 format.

Returns: the HSM\_ESTAMP stamp.

### HSMconfirmSettlementTransaction

Syntax: `HSMconfirmSettlementTransaction(sum, BEM_liqbalance, BEM_outbalance, BEM_conf_hash, receiving_HSM_signature_PKC, HSM_ECSTAMP)`

Description: This function is called when the sending E-Settlement module has received a positive acknowledgement (which is an ECSTAMP) from the receiving side. The HSM decreases its outstanding balance. PKC parameter is in X.509v3 format.

Returns: the status after the function has been executed.

### HSMrejectSettlementTransaction

Syntax: `HSMrejectSettlementTransaction(sum, BEM_liqbalance, BEM_outbalance, BEM_conf_hash, receiving_HSM_signature_PKC, HSM_ECSTAMP)`

Description: This function is called when the sending E-Settlement module has received a negative acknowledgement (which is an ECSTAMP) from the receiving side. The HSM increases its liquidity balance and decreases its outstanding balance. PKC parameter is in X.509v3 format.

Returns: the status after the function has been executed.

## 4.5.3 Receiving HSM API

This API contains HSM functions needed when for payment transaction on the receiving E-Settlement module side.

### HSMpresentReceivedSettlementTransaction

Syntax: `HSMpresentReceivedSettlementTransaction(sum, BEM_liqbalance, BEM_outbalance, BEM_hash, BEM_conf_hash, sending_HSM_encrypt_PKC, sending_HSM_signature_PKC, HSM_ESTAMP, stamp_requested)`

Description: This function is called when the E-Settlement module has received an ESTAMP from the sending BEM. The HSM increases its liquidity balance. PKC parameters are in X.509v3 format.

Returns: HSM\_ECSTAMP

### HSMrequestRejectedConfirmationStamp

Syntax: `HSMrequestRejectedConfirmationStamp(BEM_conf_hash, sending_HSM_encrypt_PKC, sending_HSM_signature_PKC)`

Description: This function is called when either receiving BAPS or receiving BEM has rejected the settlement transaction. PKC parameters are in X.509v3 format.

Returns: HSM\_ECSTAMP

---

## 5 CONCLUSIONS ON E-SETTLEMENT PROTOTYPE SECURITY

One target for the E-Settlement prototype was to check whether new security technology is viable. Technology infrastructure build for the prototype proved to be well functioning. The only limitation faced was current functionality of smartcards. As there was not an off-the-self Java-ready card available at the start of the project, the prototype HSM (hardware security module) was emulated with a combination of smartcard and emulator software. Based on information from smartcard vendors Java-ready cards are available today. The other possibility would be to make dedicated smartcards for the E-Settlement, which in the production environment could provide some extra security features. Also other types of HSM-devices will become generally available.

Security is an important part of further development of E-Settlement concept. For guidance in next phases security audit of the prototype implementation is needed. This area was not covered in our work. Already in the prototype work segregation of the development work was implemented, as HSM and BEM modules were built by separate companies. This principle should be used also in next phases.

Further development of E-Settlement security can be based on the same security management framework as prototype. Documentation including security policy and risk assessment should be reviewed based on results of a security audit.

On a security technology side major developments can be seen. Even in a half-year period we have seen evolution of both PKI technology and hardware security modules. In a foreseeable future we can expect new faster and more sophisticated hardware devices that can store and process data in tamper-proof way.

---

## Appendix A: DETAILED STAMP HANDLING PROCESSES

### ESTAMP creation in sending BAPS/BEM

ESTAMP is created in the sending BEM using the following process:

1. Sending BAPS calls `presentSettlementTransaction` in Settlement API of Bank Interface.
2. BEM validates the transaction against bank profile and the liquidity available in BEM. If successful, update BEM bank-to-bank payment information and decrease BEM liquidity balance. If not successful, return with a fault code.
3. BEM forms BAPS\_ESTAMP.
4. BEM uses its internal bank-to-bank settlement information to create the fields of BEM\_PL\_ESTAMP and BEM\_CR\_ESTAMP. At this point, BEM sets the primary HSM to form HSM\_ESTAMP, reflected in HSM\_ID\_SEND. A hash value (BEM\_HASH) is calculated over BEM\_CR\_ESTAMP fields and BAPS\_HASH. BEM\_CR\_ESTAMP is signed with BEM signing key and encrypted for the receiving BEM using BEM encryption key.
5. BEM calls the `HSMpresentSettlementTransaction` function of the HSM interface of its HSMs. The arguments to this call are sum of the settlement transaction, current BEM liquidity balance, outstanding BEM liquidity balance, receiving BEM address, BEM\_HASH and a boolean value indicating whether stamp is requested or not (true for the primary HSM, false for the secondary ones). HSM validates the transaction as follows:
  - 5.1 HSM validates the transaction by comparing reported BEM balance to its own balance
  - 5.2 If the balances match and if stamp was requested, HSM forms HSM\_ESTAMP, which is signed by HSM signing key and encrypted for receiving HSMs, the key needed for encryption is given to HSM in the function call.
  - 5.3 If stamp creation was successful, HSM decreases HSM liquidity balance and increases HSM outstanding balance.
  - 5.4 A return value and HSM\_ESTAMP (if requested) are returned to BEM.
6. If HSM\_ESTAMP creation was not successful, the payment process is immediately stopped and SAS is informed.
7. If HSMs were successfully updated, BEM decreases liquidity balance and increases outstanding balance. The transaction is put to queue of unconfirmed payments. BEM bank-to-bank payment information is updated. BEM forms the ESTAMP by concatenating BAPS\_ESTAMP, BEM\_PL\_ESTAMP, BEM\_CR\_ESTAMP and HSM\_ESTAMP. The successful creation of ESTAMP is written to BEM transaction log, together with the newly created ESTAMP.
8. ESTAMP and BEM\_HASH are returned to BAPS.
9. Sending BAPS joins ESTAMP to interbank payment message and sends it to receiving BAPS.

---

## ESTAMP validation in receiving BAPS/BEM

ESTAMP is validated in the receiving BAPS/BEM using the following process:

1. Receiving BAPS checks that the received payment transaction is valid in the payment system and that the BAPS\_HASH value contained in ESTAMP is valid. See “Rejection by receiving BAPS” for the process followed if the transaction is rejected at this point.
2. Receiving BAPS calls `presentReceivedSettlementTransaction` in Settlement API of Bank Interface, with ESTAMP as argument.
3. Receiving BEM extracts the ESTAMP parts. BEM decrypts BEM\_CR\_ESTAMP and validates BEM\_CR\_SIGN. If validation of signature fails, SAS and CEM must be informed of potential security problem and a rejection stamp created.
4. If validation of signing is successful, BEM validates the payment transaction information by checking the BEM\_CR\_ESTAMP fields by calculating a hash value and comparing it to BEM\_HASH. If hash validation fails, SAS and CEM must be informed of potential security problem and a rejection stamp created.
5. If validation of fields is successful, BEM continues by forming BEM\_PL\_ECSTAMP and BEM\_CR\_ECSTAMP fields. BEM also selects the primary HSM to form the confirmation stamp (reflected in HSM\_ID\_SEND of BEM\_CR\_ECSTAMP). BEM\_STATUS is set to indicate successful reception by BEM. BEM\_CONF\_HASH is calculated over BEM\_CR\_ECSTAMP and BAPS\_HASH fields. BEM\_CR\_ECSTAMP is signed with BEM signing key and encrypted for the sending BEM using the receiving BEM’s encryption key.
6. BEM calls `HSMpresentReceivedSettlementTransaction` of HSM interface for its HSMs, the arguments being transaction sum, current BEM liquidity balance, BEM outstanding balance, BEM\_HASH, BEM\_CONF\_HASH, HSM\_ESTAMP, sending HSM encryption key PKC and a boolean value indicating whether confirmation stamp is requested or not (true for the primary HSM, false for others). HSM validates the transaction (and possibly forms the confirmation stamp) as follows:
  - 6.1 HSM compare reported BEM balance to HSM liquidity balance plus the sum of the transaction.
  - 6.2 If the balances match, decrypt HSM\_ESTAMP and check validity of signature using the signature key containing PKC given by BEM.
  - 6.3 If validation of signature is successful, validate the HSM\_ESTAMP fields (compare BEM\_HASH received as parameter to BEM\_HASH).
  - 6.4 If validation of fields is successful, increase HSM liquidity balance.
  - 6.5 If HSM was requested to form HSM\_ECSTAMP, the fields are formed (HSM\_STATUS indicates successful or failed validation by HSM). HSM\_ECSTAMP is signed by HSM signing key and encrypted for sending HSMs, the key needed for encryption is given to HSM in the function call.
  - 6.6 HSM returns a status value and HSM\_ECSTAMP (if requested) to BEM.
7. If HSM\_ECSTAMP creation was not successful, the payment process is immediately stopped and SAS is informed.



8. If HSMs were successfully updated, BEM updates bank-to-bank payment information and increases BEM liquidity balance. BEM forms the ECSTAMP by concatenating BAPS\_ESTAMP, BEM\_PL\_ECSTAMP, BEM\_CR\_ECSTAMP and HSM\_ECSTAMP. The successful validation and creation of ECSTAMP is written to BEM transaction log, together with the newly created ECSTAMP.
9. A return value and ECSTAMP are returned to receiving BAPS.
10. If return value indicates a successful validation of E-Settlement stamp, receiving BAPS can now commit the payment transaction, otherwise it is rolled back. In both cases, receiving BAPS joins ECSTAMP to interbank payment confirmation message and sends it to sending BAPS.

### Rejection by receiving BAPS

If a payment transaction is rejected by the receiving BAPS, the following process is followed:

1. Receiving BAPS calls `recordReceivedSettlementIncident` in Settlement API of Bank Interface, with ESTAMP as argument.
2. Receiving BEM extracts the ESTAMP parts. BEM decrypts BEM\_CR\_ESTAMP and validates BEM\_CR\_SIGN. If validation of signature fails, SAS and CEM must be informed of potential security or consistency problem and fault code returned to BAPS.
3. If validation of signature is successful, BEM validates transaction information by checking the BEM\_CR\_ESTAMP fields by calculating a hash code and comparing it to BEM\_HASH. Return a fault code to BAPS if validation is not successful.
4. If validation of fields is successful, BEM requests the primary HSMs for a rejection stamp (HSM\_ID\_REC). BEM\_PL\_ECSTAMP and BEM\_CR\_ECSTAMP fields are created. BEM\_STATUS shows cause of failure (rejection by receiving BAPS). BEM\_CONF\_HASH is calculated over BEM\_CR\_ECSTAMP and BAPS\_HASH fields. BEM\_CR\_ECSTAMP is signed with BEM signing key and encrypted for the sending BEM using the receiving BEM's encryption key. BEM updates BEM bank-to-bank payment information.
5. BEM calls the chosen HSM's `HSMrequestRejectedConfirmationStamp` HSM interface function. The parameters to this function are sending HSM encryption key PKC and BEM\_CONF\_HASH.
6. HSM creates the HSM\_ECSTAMP fields. HSM\_STATUS indicates rejection by BAPS. HSM\_ECSTAMP is signed by HSM signing key and encrypted for sending HSMs, the key needed for encryption is given to HSM in the function call. A return value and HSM\_ECSTAMP are returned to BEM.
7. If HSM rejection stamp creation was not successful, the payment process is immediately stopped and SAS is informed.
8. If HSM rejection stamp creation was successful, BEM updates bank-to-bank payment information. BEM forms ECSTAMP by concatenating BAPS\_ESTAMP, BEM\_PL\_ECSTAMP, BEM\_CR\_ECSTAMP and HSM\_ECSTAMP. The successful creation of rejection stamp is written to BEM transaction log together with ECSTAMP.
9. A return value (indicating OK) and ECSTAMP are returned to receiving BAPS.

10. Receiving BAPS joins ECSTAMP to interbank payment confirmation message and sends it to sending BAPS.

### **Confirmation ECSTAMP handling in sending BAPS/BEM**

Confirmation ECSTAMP is handled in the sending BEM using the following process:

1. Sending BAPS calls `confirmSettlementTransaction` in Settlement API of the bank interface, with ECSTAMP as an argument.
2. Sending BEM extracts the ECSTAMP parts. BEM decrypts `BEM_CR_ECSTAMP` and validates `BEM_CR_SIGN`. If validation of signature fails, SAS and CEM must be informed of potential security problem and a fault code returned to BAPS.
3. If signature is successfully validated, BEM validates the payment transaction information by checking the `BEM_CR_ECSTAMP` fields by calculating a hash code and comparing it to `BEM_CONF_HASH`. Return a fault code to BAPS if validation is not successful.
4. If field validation is successful, BEM calls `HSMconfirmSettlementTransaction` of HSM interface for its HSMs, the arguments being transaction sum, current BEM liquidity balance, BEM outstanding balance, `BEM_CONF_HASH`, receiving HSM's signature key containing PKC and `HSM_ECSTAMP`. HSM validates the confirmation stamp as follows:
  - 4.1 HSM compares reported BEM balance to HSM liquidity balance.
  - 4.2 If the balances match, decrypts `HSM_ECSTAMP` and checks the validity of the signature using the certificate given as an argument by BEM.
  - 4.3 If validation of signature is successful, validate the `HSM_ECSTAMP` fields (compare `BEM_CONF_HASH` received as parameter to `BEM_CONF_HASH` in `HSM_ECSTAMP`).
  - 4.4 If validation of fields was successful, decrease HSM outstanding balance.
  - 4.5 HSM returns a status value to BEM.
5. If stamp validation was not successful, the payment process is immediately stopped and SAS is informed.
6. If all HSMs were successfully updated, BEM decreases its outstanding liquidity balance. The transaction is taken away from the queue of unconfirmed transactions. BEM updates bank-to-bank payment information. The successful confirmation is written to BEM transaction log, together with the ECSTAMP. A positive return value is returned to BAPS.
7. If return value indicates a successful confirmation of E-Settlement confirmation stamp, the sending BAPS can now commit the payment transaction, otherwise it is rolled back.

---

**Rejection ECSTAMP handling in sending BAPS/BEM**

Rejected ECSTAMP is validated in the sending BEM using the following process:

1. Sending BAPS calls `rejectSettlementTransaction` in Settlement API of the bank interface, with ECSTAMP as an argument.
2. BEM extracts the ECSTAMP parts. BEM decrypts `BEM_CR_ECSTAMP` and validates `BEM_CR_SIGN`. If validation of signature fails, SAS and CEM must be informed of the potential security problem and a fault code must be returned to BAPS.
3. BEM calls all of its HSMs' `HSMrejectSettlementTransaction` function. The parameters to this function are transaction sum, current BEM liquidity balance, BEM outstanding balance, `BEM_CONF_HASH`, receiving HSM signature key containing PKC and `HSM_ECSTAMP`
4. HSM validates the rejection stamp as follows:
  - 4.1 HSM compares reported BEM balance to HSM liquidity balance.
  - 4.2 If the balances match, decrypt `HSM_ECSTAMP` and check the validity of the signature using the PKC given as argument by the BEM.
  - 4.3 If validation of signature is successful, validate the `HSM_ECSTAMP` fields (compare `BEM_CONF_HASH` received as parameter to `BEM_CONF_HASH` in `HSM_ECSTAMP`).
  - 4.4 If validation of fields was successful, increase HSM liquidity balance and decrease HSM outstanding balance.
  - 4.5 HSM returns a status value to BEM.
5. If HSM stamp validation was not successful, the payment process is immediately stopped and SAS is informed.
6. If all HSMs were successfully updated, BEM increases its liquidity balance and decreases BEM outstanding balance. The transaction is taken away from the queue of unconfirmed transactions. BEM updates bank-to-bank payment information. The successful rejection is written to BEM transaction log, together with the ECSTAMP. A positive return value is returned to BAPS.
7. If return value indicates a successful confirmation of E-Settlement rejection stamp, the sending BAPS can now commit the payment transaction, otherwise it is rolled back.

---

## Appendix B: ALGORITHMS

This appendix briefly presents some central cryptographic algorithms.

### Secret key algorithms

Triple DES (EDE-DES) using is today's standard for assuring confidentiality with a symmetric cipher. ECBS recommends that a triple DES using a 112 bit key should be used for medium and high risk applications [ECBS TR406]. Triple DES is based on the DES block cipher, which means that it encrypts data in blocks. The DES algorithm takes a plaintext block of 64 bits as an input. The resulting cipher text is also 64 bits in length. If the input plaintext length is not a multiple of 64 bits the plaintext has to be padded, since DES operations are always performed on 64 bit blocks. The key usage period for a secret key should be as short as possible and therefore usage of non-reusable session keys should always be preferred.

See [APPCRY] for more details on secret key algorithms and triple DES.

### Public key algorithms

Public key algorithms use two different keys – one public and the other private. It is computationally hard to deduce the private key from the public key. Anyone with the public key can encrypt a message but not decrypt it. For assuring data integrity and data origin authentication, the asymmetric RSA algorithm with the key length of *1024 bits* is recommended by ECBS [ECBS TR406]. RSA could also be used for confidentiality, but this is infeasible, since private key operations are a very compute intensive operation, requiring nearly one-tenth of a second on the fastest Pentium processor.

See [APPCRY] for more details on public key algorithms and RSA.

### Hash algorithms

For assuring data integrity Secure Hash Algorithm (SHA) based digital signature is recommended by the ECBS [ECBS TR406]. Currently there exist no known cryptographic attack against SHA. The SHA is called secure because it is designed to be computationally infeasible to recover a message corresponding to a given signature or to find two different messages, which produce the same signature. Any change to a message in transit will, with a very high probability, result in a different signature. SHA produces a *160 bit* signature of any message that is shorter than  $2^{61}$  bytes.

See [APPCRY] for more details on signature algorithms and SHA.

### The hybrid crypto-algorithm

Public-key algorithms are seldom used directly for achieving message confidentiality. Instead a session key is generated and the message is encrypted using a symmetric algorithm, and only the session key is encrypted using a public-key algorithm. One reason for this is that symmetric algorithms are generally at least 1000 times faster than public-key algorithms. Another important reason for this is that public-key cryptosystems are vulnerable to chosen-plaintext attacks, this attack doesn't help recovering the encryption key but the encrypted message can be determined.

Algorithms using symmetric session keys encrypted with public-key algorithms are usually called hybrid crypto-algorithms. Next a concrete example of such an algorithm is presented, the message that is sent is denoted with the letter *M*.

1. The sender signs the message *M* with its private key *SKI*.  
$$S_{SK1}(M)$$

- 
2. The sender generates a random session key  $K$  and encrypts the signed message with the random session key.  
$$E_K(S_{SK1}(M))$$
  3. The sender encrypts the generated session key with the receiver public key  $PK2$ .  
$$E_{PK2}(K)$$
  4. The sender sends the encrypted and signed message and the encrypted session key to the receiver.  
$$(E_K(S_{SK1}(M)), E_{PK2}(K))$$
  5. The receiver decrypts the encrypted session key with its private key  $SK2$  and obtains the session key  $K$ .  
$$D_{SK2}(E_{PK2}(K)) = K$$
  6. The receiver uses the session key to decrypt the encrypted and signed message.  
$$D_K(E_K(S_{SK1}(M))) = S_{SK1}(M)$$
  7. The receiver verifies the signature and obtains the message  $M$ .

Encryption ensures the message confidentiality if the cryptographic algorithms and key lengths are chosen wisely. The integrity is checked when the signature is verified. If the encrypted and signed message is altered the signature will be invalid.

See [APPCRY] for more details on hybrid systems.

## **Section 4**

### **E-SETTLEMENT**

### **COST / BENEFIT ANALYSIS**

The views expressed here are those of the project. They are at the moment preliminary and open for all comments. The views do not necessarily reflect the views of the Bank of Finland.

---

## **TABLE OF CONTENTS**

<b>TABLE OF CONTENTS .....</b>	<b>116</b>
<b>1 INTRODUCTION.....</b>	<b>117</b>
<b>2 BENEFIT FACTORS.....</b>	<b>118</b>
<b>3 COST FACTORS.....</b>	<b>119</b>
<b>4 ESTIMATING DIFFERENT FACTORS' EFFECTS.....</b>	<b>121</b>
4.1 E-SETTLEMENT COMPARED TO RTGS-SOLUTIONS.....	121
4.2 E-SETTLEMENT AND NETWORK-BASED SOLUTIONS COMPARED TO ACH PROCESSING	123
4.3 E-SETTLEMENT COMPARED TO CORRESPONDENT BANKING .....	124
4.4 E-SETTLEMENT IN PARTLY CUSTOMER DRIVEN PAYMENT SOLUTIONS AND E-COMMERCE .....	125
<b>5 OVERALL ASSESSMENTS AND NEED FOR FURTHER ANALYSIS .....</b>	<b>126</b>

# 1 INTRODUCTION

This is a general analysis of the different cost- and benefit factors of the e-settlement proposal. The main focus is in trying to identify the different factors in question and their relationships with other changes in the general payment environment. At this stage it is not possible to make concrete numerical analyses because information is not currently available for such calculations. Such analyses can only be made for given specified application areas. However, general analyses are made for some given possible application areas (RTGS, ACH, correspondent banking and user-driven network and e-commerce payments).

The interbank settlement convention is a distinct part of payment systems and has to be assessed against general payment system developments. It has also to be reviewed against the general information and communication technology (ICT) developments, because the possibilities to easily exploit new techniques used in other sectors and environments are improving. The time horizon is an important element, the effect of which is generally overestimated in the short run and underestimated in the long run.

The presumption of this analysis is that the reader is familiar with the e-settlement proposal and has read the introduction to e-settlement and has a general knowledge of the e-settlement architecture and hardware/software requirements.



## 2 BENEFIT FACTORS

Following benefit factors can be recognised

- final settlement is achieved simultaneously as part of the payment processing, the settlement information is transported as part of the payment
- settlement and systemic risks are reduced when final settlement is immediate and the liquidity can be directly reused
- liquidity management becomes easier when the payment liquidity can be consolidated to one account completely managed by the bank itself
- in a central bank money based e-settlement scheme the risks related to private settlement money and interbank credit lines can be avoided
- the payment process is straightforward between the sending and receiving bank (one payment message sent and one confirmation message in reply)
- the e-settlement convention is easy to implement in an TCP/IP network environment like SWIFTnet, and the current investments in eg CBTs can be fully utilised
- e-settlement can be implemented both for transaction and batch based transmissions
- STP from end-to-end can be achieved both regarding payments and settlements in a network based e-settlement environment given that basic payment standardisation is in place
- e-settlement provides a general, neutral, automated and standardised interbank settlement facility that could replace several of the current nonstandardised solutions
- e-settlement includes automated reconciliation and error detection as well as improved encryption solutions
- E-settlement facilitates improved speed and risk reduction (settlement risk, credit risk and operational risks) in payment systems in general and can increase electronic processing by transferring volumes from paper-based and less efficient payment methods
- Modern low cost and standardised technology is used that can also be used for other purposes (for instance the required PKI/CA, the certification authority for public key infrastructure could also be used to certify other time of messages than needs to be transport securely between two banks)
- Considerable cost savings should be achieved but for these detailed analyses are needed regarding specific implementation areas.

Overall benefits are very dependent on how wide the acceptance is among banks regarding the new settlement convention. Also in e-settlement, as in any network service, the statement – the larger user community, the larger benefits – is true. One basic requirement is also that one or more network based and standardised payment infrastructures are used (eg SWIFTnet).

### 3 COST FACTORS

The ICT cost factors are recognisable

- software costs for the applications
- hardware cost for the decentralised equipment (server and high secure module)
- integration costs at bank/participant level
- centralised monitoring facilities and operations
- maintenance costs.

Most of the ICT costs are fixed and therefore not dependent on the total volumes processed in the e-settlement system. Because of the decentralised structure the system is able to carry a large load of transactions sharing the fixed costs. The larger volumes the lower costs per transactions will be.

The variable costs are very low because the process is so automated. In fact the cost per transaction could become so marginal that it can be neglected in large volumes. As compared to the current settlement routines the centralised site and personnel will be very small indeed. In banks/participants the need for systems and staff for reconciliation and nostro/loro/clearing account management will almost disappear, when the settlement process is automated and consolidated to one or very few accounts. What is needed is somebody monitoring the liquidity situation and making transfers to/from the central bank account once in awhile.

The change and integration costs at bank/participant level will probably be the largest costs seen by the individual banks, because a large community would share all the other costs. The integration costs of the individual bank will depend on its current ICT structure, whether it is still based on old batch processes or modernised using real-time processes and network interfaces. The costly change is from batch based processes to a real-time network environment. This is the essential prerequisite for a network-based infrastructure making e-commerce and e-banking possible. In order to be able to serve e-customers most banks, which have not yet changed their internal ICT structures, will make this change-over during the next years. This will facilitate the implementation of e-settlement. From cost calculation point of view it will be a difficult task to separate the cost of change to different sub-areas of the overall network based processing.

The maintenance in distributed systems requires interoperable software and hardware to be in place in a large number of sites. Modern network solutions facilitate in a very efficient way software distribution. The system structure in the e-settlement solution will split the process into very rudimentary services (eg get-stamp, present-stamp etc) which will need little maintenance after the initial implementation. The maintenance costs will also be shared between other sub-systems in the interbank communication area, like communication platforms, security (PKI) modules etc which have in any case to be distributed to each participant.

Some resources will also be needed for marketing the new approach, making initial tests and co-ordinating the implementation phase. These kinds of overheads are difficult to estimate and depend on how intense the process will be. If a clearly defined first implementation area can be found and the participants coordinate well their interrelationships then the costs will be lower compared to a prolonged process with less clearly defined interrelationships. The most probable outcome will be the general network dispersion model eg something compared to the email introduction with some first early entrants and the mass coming on gradually when the benefits become apparent and the pressures from other users become effective.

Direct ICT costs are easier to estimate than different kinds of indirect costs, which need sometimes also political considerations eg

- Are there old investments that become obsolete? How should these kinds of sunk costs be treated?
- Banks have also concerns regarding float benefits due to the slow processing in the current systems. Will e-settlement as such speed up processing or is this a more general trend

- independent of the settlement convention? Should this be seen as a cost factor? Would it be better with more transparent pricing methods?
- E-settlement will automate the settlement process and the network-based payment system will make the complete payment process more efficient. This will reduce the number of human resources needed and result in some cost of change. How should these be distributed to e-settlement calculations?
  - It would not be very efficient to introduce e-settlement as a new parallel settlement convention to all the old ones. In order to achieve cost benefits some of the old ones need to be replaced, which is also creating costs of change. Indeed running old and new systems for a long time in parallel could even increase costs. How much of these kind of cost of change should be included in the e-settlement system costs?
  - Changing the payment processing from mainly batch based processes to partly or almost completely real-time processing will change the system and application architecture. When is the most cost effective time slot to do this kind of major change? How big is the pressure from e-banking and e-commerce for speeding payments.

## 4 ESTIMATING DIFFERENT FACTORS' EFFECTS

It is always difficult to estimate ICT costs and especially mid and long range. According to the prototype findings following general figures can be given in order to get the feel of the cost range

- the e-settlement hardware module costs would be in the range of 15.000–30.000 euros per site (ie per bank). If specialised security processing boxes are designed then approximately 10.000 euros should be added. These costs depend much on the volumes to be implemented and the costs will probably decrease considerably over time with PKI being generally in use.
- The e-settlement software module building costs will be in the range of 3 to 6 million euros for a basic functionality and could increase in maximum to 12 million euros when a lot of sophisticated add-on services are added. The costs per bank will be rather low when it is shared on a community of some thousands bank.
- Implementation and adaptation costs per bank will depend on banks' internal ICT structures and can be low and sharable between many banks if the integration is done directly between the SWIFT CBT and the e-settlement module. In the CBT case the integration costs per CBT software would be clearly below 1 million euros if good application interfaces (APIs) are available in the CBT software.
- For smaller banks the hardware modules could be cheaper, low volume type, when large banks would probably benefit from installing systems with fast dedicated PKI processors (most of the IT resources are needed for the PKI processes).

This level of costs will generally bring the settlement costs well below one cent per transaction for an average bank if a major part of the payment transactions are sent via e-settlement.

Assessing the benefits requires some kind of comparisons with current and potential competing settlement solutions. However, in most cases we lack the cost information of current systems. In some cases the costs for centralised systems, like ACHs, are available, but the costs accrued in the different banks using the centralised systems are not available. The benefits of the e-settlement proposal are also difficult to separate from the overall benefits of a network-based payment system model. The e-settlement model is part of a larger picture in which the complete payment system is re-engineered using modern technology. Below is general cost-benefits analysis and comparisons for some potential implementation cases where e-settlement and network-based solutions may replace old infrastructures ie conventional RTGS solutions, ACH clearing centres, correspondent banking and customer driven payment solutions.

### 4.1 e-settlement compared to RTGS-solutions

The traditional RTGS-solution is based on a V-shaped model involving one centralised processing site or a Y-shaped model involving two different centralised processing site. In an environment with more than one central bank, like in the EMU-area, even an X-shaped model is proposed with three centralised sites involved in processing the same payment. Compared to these centralised models each payment will be processed in the e-settlement system only in the sending and receiving bank. (See figure 1).

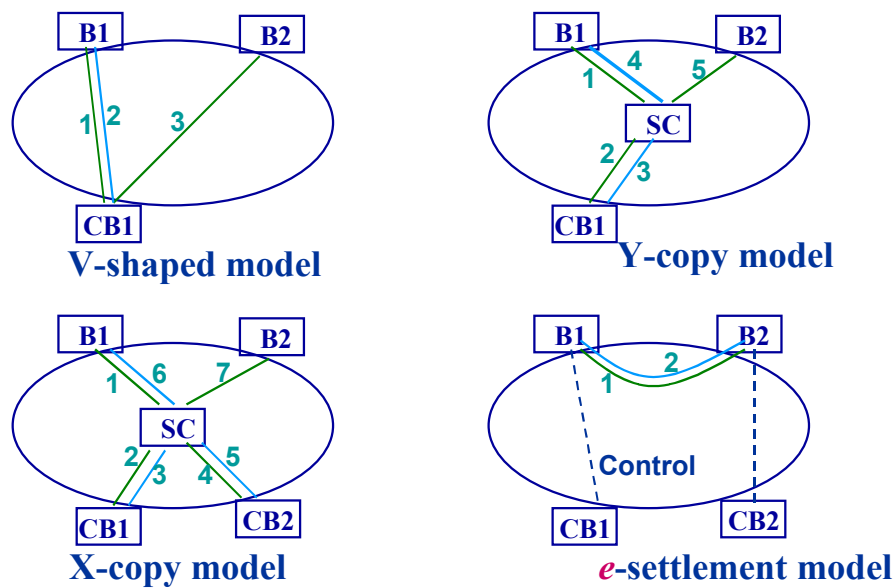


Figure 1: Message structures in different RTGS-models and the e-settlement proposal

Compared to the traditional RTGS-solutions following benefits can be found in the e-settlement proposal

- no centralised transaction processing costs and thereby clearly lower operative costs
- payments are processed directly end-to-end between banks and not just on the central bank accounts of the banks
- final e-settlement is completely integrated to the payment process (no need to separate between payment and cover/settlement messages)
- low cost server equipment for bank interface
- improved and automated security and continuous reconciliation features detecting errors and exceptions immediately
- fewer telecommunication messages and direct end-to-end control between sending and receiving bank. E-settlement needs two messages (sending payment and receiving confirmation), while the RTGS solutions need two to four messages for sending the payment and one to three confirmation messages between the different processing participants
- more efficient back-up and emergency solutions in a decentralised environment where the “unhit” part will/can continue the processing with interruptions.

Most of the service features of RTGS systems is found in the e-settlement like final real-time settlement in central bank money, queuing and bilateral netting if needed.

Some additional costs will be involved in some cases

- the intraday control and monitoring of the system needs resources
- multilateral netting will require a centralised site if/when multilateral netting is needed for part of the transaction volumes. However, the possibility for netting is diminishing when customers press for real-time performance, because netting will always require queuing and delaying in order to find payments to net
- authorisation checks for transferring exceptionally large transactions
- the telecommunication and server environment must have high availability which is almost already now required of RTGS-solutions.

As a general assessment one can safely state that the overall costs of an e-settlement approach is clearly lower than of traditional RTGS approaches for normal transaction flows. However, in some cases where exceptionally large transactions requiring special authorisations are processed almost only and very frequently the cost benefits will decrease.

## 4.2 e-settlement and network-based solutions compared to ACH processing

In the traditional ACH approach all banks are sending their transactions in batches to the ACH, which is sorting the batches according to the receiving banks and sending them onwards and calculating at the same time the net positions between the banks. The settlement is then generally done once in the end of the day. In some ACH systems there are several settlement occasions a day in the settlement bank. In a continuous net settlement system payments are transferred as separate transactions via the center to receiving banks and the center keeps settlement balances in real-time. In both cases a network-based model with the e-settlement would replace the center with the TCP/IP network as the routing/sorting mechanism and the e-settlement balances and stamps would provide immediate final settlements. In the batch-based solution the sending/initiating bank's payment system or network server will sort the overall payment batch into sub-batches according to receiving/beneficiary's bank (See figure 2).

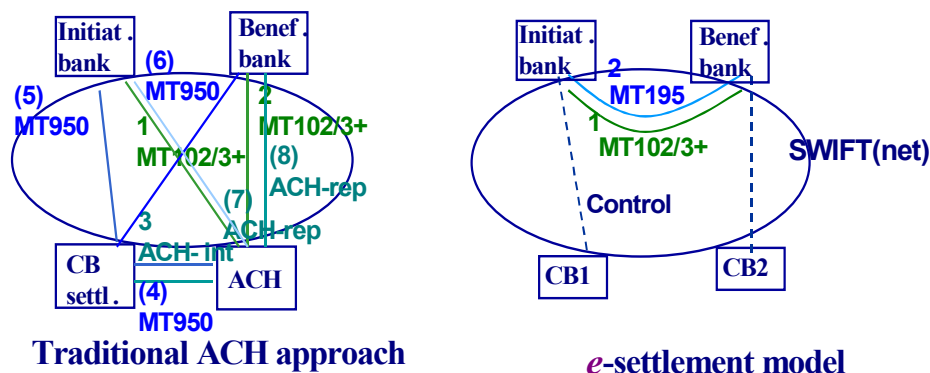


Figure 2: Payment and message flows in traditional ACH and e-settlement environment (national message standards are used in most systems instead of the SWIFT messages shown in the figure)

Compared to the traditional ACH approach following benefits can be found in the e-settlement proposal

- payments or batches of payments are sent directly to receiving bank and no centralised sorting and processing is needed. The routing is a general feature of the network eg SWIFTnet
- the settlement will be automated and continuous during the day, but still delivering the same kind of settlement report at the end of the day (or at given reconciliation intervals during the day) as the traditional ACH settlement report
- payments can be processed freely between parties without the strict timetables required in the ACH processing
- no costs for centralised processing just centralised monitoring
- the dependency on the centralised services and its availability will be very low
- problem situations at different participants are easy to handle and the “unhit” participants can continue processing without problems.
- a move from batch-based transmissions to transaction-based transfers can be made gradually an in pace with the press for e-commerce supporting services.

Some additional costs can be found in the e-settlement approach

- in the batch environment the payment batches has to be sorted by receiver in the network server or in the banks payment systems. However, this cost is marginal with the computer power of modern servers
- if the payment flows are very much unbalanced then a direct settlement of batches and/or bilateral net settlements will increase somewhat the liquidity needs. The more payment batches are split into smaller batches during the day the more even the liquidity need will generally be. Different kinds of timing rules will also reduce the liquidity need. In the case of transaction based settlement the liquidity/credit line need will be the same in continuous net settlement as

- in e-settlement. If the same liquidity is used both for large-value and retail payments, then the retail payment needs will be marginal compared to the large-value requirements
- in a batch system the number of transmitted batches will increase when these are done by receiving bank. However, this is a very insignificant cost factor in a TCP/IP network where the costs are based on the total bitvolume transmitted which will be the same in both cases but differently distributed per receiver.

The network-based system approach and e-settlement are together able to deliver significant cost reductions in the ACH case by replacing the ACH sorting and routing process by network-based routing. The telecommunication costs will probably also decline because the transactions are only transmitted once in the e-settlement case instead of twice as in the ACH case. There are considerable variations in the different ACH solutions so detailed analysis can only be made case by case.

### 4.3 e-settlement compared to correspondent banking

In correspondent banking payments are routed through the SWIFT-net based on bilateral relationships. If SWIFT keys are exchanged messages can be transmitted directly to beneficiary's bank. If there are no direct relationship then an intermediate bank has to be used. Settlement is provided via a common settlement bank when the banks involved have not open accounts for each other. This can make the correspondent processing structure very complicated in particular when payments are sent between small or middle sized banks. In figure three a somewhat complex situation is shown where the initiating bank has to use a different correspondent bank and settlement bank to reach beneficiary's bank (eg a Finnish bank using a German commercial bank for settlement in order to reach a savings bank via a savings bank's central bank). In the e-settlement case all banks (in the SWIFTnet) are using PKI-keys so every bank can securely send messages directly to any bank in the SWIFT network. The settlement will be included by the e-settlement module. All payments can thereby be routed and settled by a direct transfer. (See Figure 3).

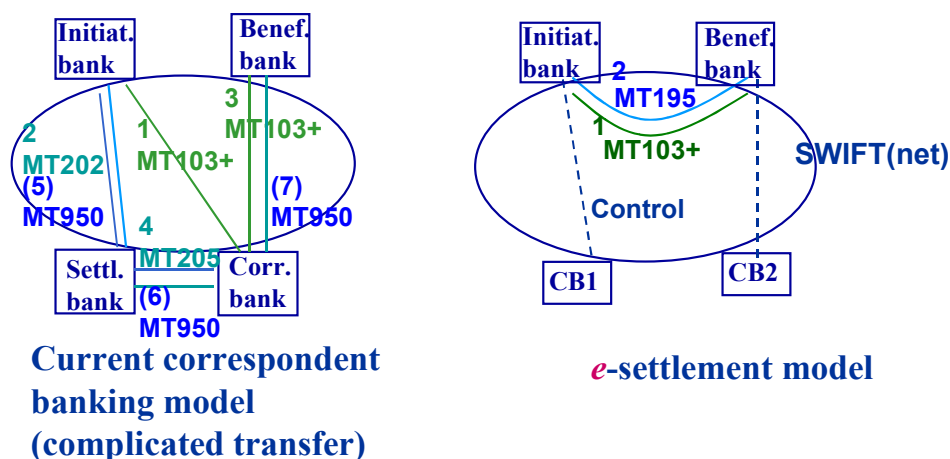


Figure 3: Message structure in correspondent banking compared to e-settlement

Compared to the current correspondent banking model following benefits can be found in the e-settlement proposal and the open payment network

- the PKI solution gives a secure possibility for direct messages between all banks in the network
- the number of intermediary processing phases can be reduced
- the process is an end-to-end dialogue between the two involved banks, which means that any errors can be corrected immediately
- the need for cumbersome nostro/loro reconciliation with message flows will disappear and it will be replaced by automatic and continuous reconciliation and settlement using the e-settlement balance and module
- the payment process will be very straight forward and the possibility to introduce full STP will become simpler

- it will be easy to speed up the payment process according to customers needs.

In the correspondent banking case there cannot be found any additional costs in applying the e-settlement approach except for the cost of change, which is associated with all modifications. In the payment process itself all different type of costs (eg communication costs, processing , key management, settlement etc costs) should be lower in the e-settlement approach.

International transfers are today complex, costly and slow. The situation would improve considerably by introducing the e-settlement system and using modern PKI and TCP/IP solutions, as they will be provided by SWIFTnet.

#### **4.4 e-settlement in partly customer driven payment solutions and e-commerce**

Some customer groups have found it important to speed up the change-over to network-based solutions and direct customer-to-customer communication in order to efficiently exchange electronic data. Most of these initiatives are based on XML, PKI and TCP/IP technology. These groups would also want to have electronic and integrated interfaces directly with banks. In this case also the customer interfaces would be included in addition to the interbank integration of the previous examples. One example of such customer cooperation is Papinet, the paper industry network, which together with SWIFT and some banks (eg Nordea) is building a direct interface for paying. For this customers immediate payments and complete liquidity control is a basic requirement. With e-settlement banks could have a possibility to provide this kind of service which cannot be done with traditional payment systems and instruments.

The same kind of demands will be facing banks generally in the e-commerce environment. Different kinds of e-services and e-goods are provided in realtime and interbank e-payments will be expected to do the same.

E-settlement in a real-time payment-network could provide the solutions in the interbank area of e-commerce payments. There is no general current solution in place so a comparison of benefits and costs is difficult to do. There are interbank real-time payment solutions in some countries but these vary a lot so any comparison has to be done on case by case basis.



## **5 OVERALL ASSESSMENTS AND NEED FOR FURTHER ANALYSIS**

An overall assessment of the benefits and costs of the e-settlement approach could best be done for a given concrete case in which the model would be implemented for a given payment flow using one or more settlement systems presently. The cost for the current systems could be compared to the estimated costs of the new system.

Another situation when a clear assessment can be made is when a clearly outdated system is in use and it should be replaced by a more modern one. Then the e-settlement model can be compared to the competing proposals. In principle, this kind of comparison could also be made generally by comparing different solutions in a situation when starting from scratch.

The time horizon for the analysis is also important. The longer the time horizon is the greater will the need be for new systems replacing the old ones. The immediate real-time service in Internet will affect larger parts of the society and banks will have to provide immediate e-banking services also in interbank context.

In order to make a detailed assessment and investment calculation one or several clear implementation cases should be found/chosen. These cases should be described including volumes so that the calculation and comparison angles are clear. One suggestion for further work would be to find suitable calculation cases and the cost and volume information for meaningful comparisons. The general analyses done in part four is suggesting that considerable cost savings could be found.

## **Section 5**

### **E-SETTLEMENT**

#### **PROTOTYPE DESCRIPTION AND PRELIMINARY EXPERIENCES**

The views expressed here are those of the project. They are at the moment preliminary and open for all comments. The views do not necessarily reflect the views of the Bank of Finland.

---

## TABLE OF CONTENTS

<b>TABLE OF CONTENTS .....</b>	<b>128</b>
<b>ABBREVIATIONS.....</b>	<b>129</b>
<b>1 INTRODUCTION.....</b>	<b>130</b>
<b>2 DESCRIPTION OF THE PROTOTYPE.....</b>	<b>132</b>
2.1 SYSTEM ADMINISTRATION SITE.....	132
2.2 BANK DRIVER APPLICATION.....	138
2.3 CENTRAL BANK DRIVER APPLICATION.....	149
<b>3 USING THE PROTOTYPE.....</b>	<b>154</b>
3.1 SYSTEM START-UP AND CONFIGURATION.....	154
3.2 PLANNING A SCENARIO.....	155
3.3 RUNNING A SCENARIO.....	155
3.4 TROUBLESHOOTING.....	158
<b>4 PRELIMINARY EXPERIENCES.....</b>	<b>159</b>
4.1 IMPLEMENTATION.....	159
4.2 PERFORMANCE.....	160
<b>APPENDIX 1: APPLIED RJE FILE FORMAT .....</b>	<b>161</b>
<b>APPENDIX 2: CONFIGURATION FILES .....</b>	<b>162</b>
<b>APPENDIX 3. RJE GENERATION TOOL .....</b>	<b>163</b>
<b>APPENDIX 4. LOG FILES .....</b>	<b>164</b>
<b>APPENDIX 5. PROTOTYPE HARDWARE AND SYSTEM SOFTWARE.....</b>	<b>165</b>

## ABBREVIATIONS

<b>Abbreviation</b>	<b>Meaning</b>
BADR	Bank Driver
BAPS	Bank Payment System
BEM	Bank E-Settlement Module
BIC	Bank Identifier Code
BOD	Beginning-of-day
CA	Certificate Authority
CBDR	Central Bank Driver
CBPS	Central Bank Payment System
CEM	Central Bank E-Settlement module
CPU	Central Processing Unit
CRL	Certificate Revocation List
EOD	End-of-day
HSM	Hardware Security Module
PKI	Public Key Infrastructure
RJE	Remote Job Entry
RTGS	Real-Time Gross Settlement
SAS	System Administration Site
SDI	System Directory
UI	User Interface

# 1 INTRODUCTION

Bank of Finland has commissioned a functional prototype demonstrating the E-Settlement system concept based on the E-Settlement system specification work [PAYMENTSYS, ITSTRUCT, SECURITY, ARCHSPEC, TECH\_ARCH, BAIF\_SPEC, CBIF\_SPEC]. The prototype work was carried out in spring and summer of 2002.

The Prototype serves several purposes:

- The implementation ensures that the ideas of E-Settlement really work in practice
- The prototype demonstrates the E-Settlement concept effectively
- The all-crucial security architecture proposed for E-Settlement can be tested
- The prototype can be used for performance estimation

The Prototype does not provide all E-Settlement features. It concentrates on the E-Settlement “Core” functionality, meaning payment processing, liquidity management, reconciliation and centralized control functions. These are all needed during the course of an “E-Settlement day”. In detail, the following functionality described in [ARCHSPEC] is provided:

- All payment processes
- Liquidity issuing and de-issuing
- Continuous reconciliation
- Bank-to-bank reconciliation
- System reconciliation
- Beginning-of-day activities
- End-of-day activities

In addition, not all exceptional situations described in [ARCHSPEC] are covered in the Prototype. The following situations can be demonstrated with the prototype:

- Payment timeout
- Missing payments
- Bank profile violation
- Double reception payment and/or confirmation
- Incomplete or incorrect payment data or E-Settlement stamp

Some further simplifications have been made in the Prototype. For example, there are no separate Bank and E-Settlement networks. In addition, although the E-Settlement concept could be applied in any interbank network, the Prototype presents the banks as SWIFT network participants.

Figure 1 below presents the structure of the E-Settlement Prototype. Note that the prototype can be run in several configurations - with one or more central banks and one or more banks under each central bank. The structure shown is an example configuration, consisting of three banks named MegaBank, GigaBank and TeraBank, two central banks named CentralBank1 and CentralBank2, and the system administration site (SAS).

All system components are connected via a switch to each other. Each bank has a corresponding BEM and central bank has a corresponding CEM, respectively. Bank driver (BADR) is a substitute for a real bank payment system (BAPS) and Central bank driver (CBDR) is a substitute for a real central bank payment system (CBPS). BADRs, CBDRs and SAS have an user interface, other components are not visible to system users.

Both BEM and CEM contain a *HSM emulator*, which in turn communicates with a smart card (HSM). HSM emulator models a type of HSM that could be provided by a JavaCard enabled smart card alone, but which for schedule reasons was implemented as an emulator and a more simple smart card.

BADRs are able to read in payment information in RJE format containing payments in MT103 or MT202 format. CBDRs maintain a deposit info for banks under control of the Central Bank. SAS manages the central repository for system configuration information, System Directory (SDI). The directory contains the following information:

- System participant info (Banks, Central banks and their info)
- PKI certificates of the system participants (published by the Certification Authority)
- PKI Certificate Revocation List (published by the Certification Authority).

IPSec protocol is used for transmission of information in E-Settlement prototype to secure the communications within the network. SAS acts as the IPSec server in the prototype. SAS also houses the Certification Authority of the system.

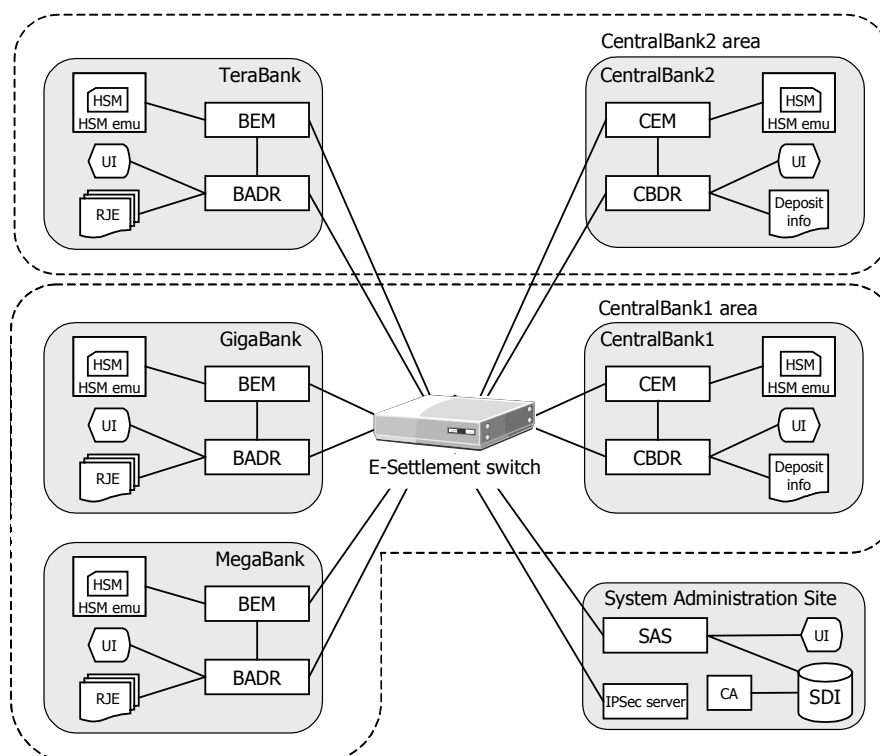


Figure 1. The structure of the E-Settlement Prototype

## 2 DESCRIPTION OF THE PROTOTYPE

This chapter describes the user interfaces of the prototype. For actual usage scenarios see next chapter.

### 2.1 System Administration Site

#### 2.1.1 Open module



*Figure 2. Open module dialog box*

This screen is shown each time a module (SAS, CEM or BEM) is being started. The user must supply the valid credentials to open the module in question. The credentials must be known by the administrator.

##### 2.1.1.1 Fields

**Module password:** the password needed in opening the module – actually, the password of the private key store of the module.

**Smart card PIN:** the 4-digit number needed in opening the HSM of the module.

##### 2.1.1.2 Buttons

**OK:** performs the operation

**Cancel:** cancels the operation

### 2.1.1.3 Columns

## 2.1.2 Bank status

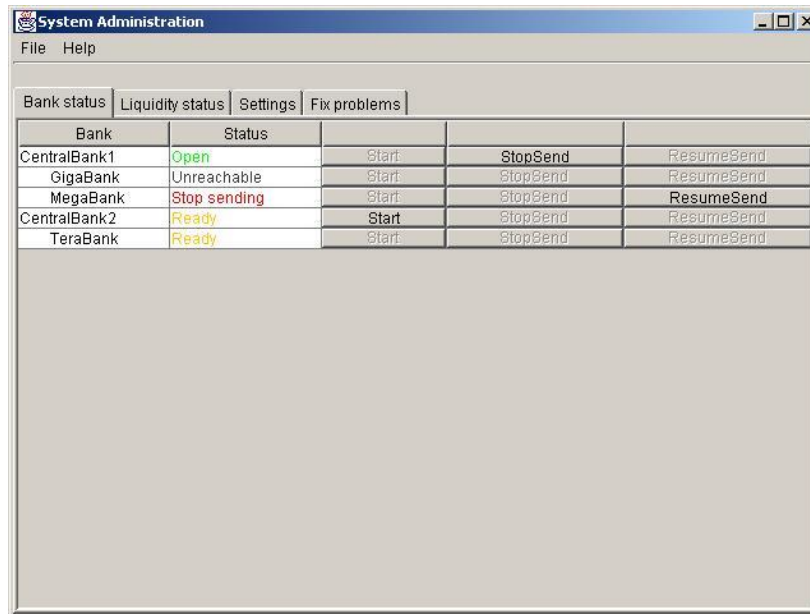


Figure 3. SAS Bank status screen

This screen allows SAS user to monitor the status of E-Settlement modules, to start modules in “Ready” status and switch the modules between normal operation (“Open” status) and “Stop sending” state. The screen shows all system participants grouped by central banks, in alphabetical order. A status is shown for each participant.

### 2.1.2.1 Fields

### 2.1.2.2 Buttons

**Start:** try to open the module. Open module dialog box (see 2.1.1) is opened. Only modules in “Ready” state have this button activated. Furthermore, the central bank must be opened before the banks under its control can be opened.

**StopSend:** set the module status to “Stop sending”. Only modules in “Open” state have this button activated.

**ResumeSend:** resume the module state to “Open”. Only modules in “Stop sending” state have this button activated.

### 2.1.2.3 Columns

**Bank:** the name of the participant bank or central bank. This column is sorted by central bank name, and then for each central bank by bank name in alphabetical order.

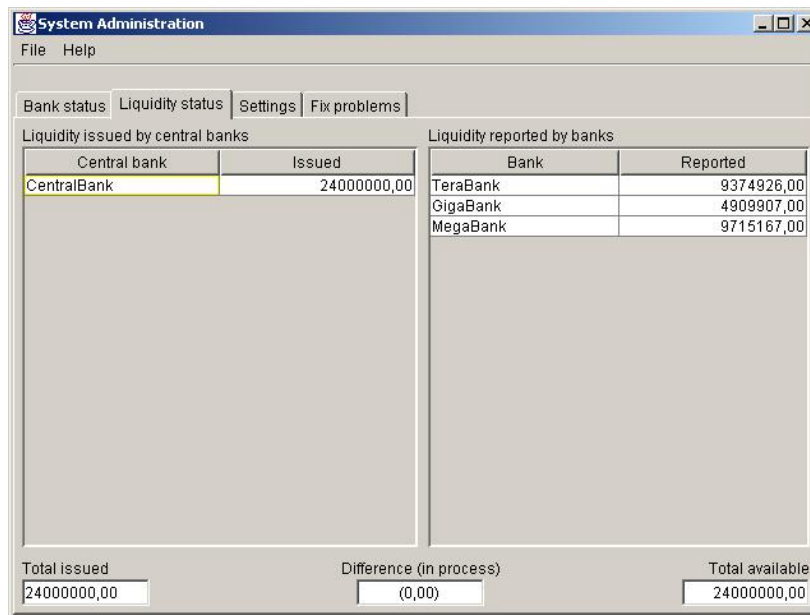
**Status:** the availability status of the system participant. The status can be one of the following:

- Unreachable (black): SAS is not able to reach the module either because it has not yet been started or there is a severe network problem.
- Ready (yellow): SAS can reach the module but the module has not yet been opened, i.e. it is not yet able to process payments.



- Open (green): the module has been opened and can process payments. This is the “normal” state of the module.
- Stop sending (red): the module has been put to “stop sending” state, meaning that the module cannot send payments at the moment, and the other system participants are advised not to send any payments to this module.

### 2.1.3 Liquidity status



The screenshot shows a window titled "System Administration" with a menu bar (File, Help) and tabs (Bank status, Liquidity status, Settings, Fix problems). The "Liquidity status" tab is active, displaying two tables and summary fields.

Liquidity issued by central banks		Liquidity reported by banks	
Central bank	Issued	Bank	Reported
CentralBank	24000000,00	TeraBank	9374926,00
		GigaBank	4909907,00
		MegaBank	9715167,00

Summary fields at the bottom:

Total issued	Difference (in process)	Total available
24000000,00	(0,00)	24000000,00

Figure 4. SAS Liquidity status screen

The E-Settlement administrator can follow the liquidity situation in the E-Settlement system in real time using this screen. Basically, the amount of liquidity issued by central banks during the day is being compared to the amount of liquidity reported by the banks at any given time. The difference in these numbers should be zero or a relatively small negative number. The negative number means that some liquidity is currently “tied” to ongoing settlement transactions.

#### 2.1.3.1 Fields

**Total issued:** The cumulative sum of liquidity issued by central banks.

**Difference (in process):** The difference between total issued and reported liquidity.

**Total available:** The cumulative sum of liquidity reported by banks.

#### 2.1.3.2 Buttons

#### 2.1.3.3 Columns

**Central bank:** The name of central bank

**Issued:** Amount of liquidity issued by the central bank

**Bank:** The name of the bank

**Reported:** Amount of liquidity reported by the bank

## 2.1.4 Settings

Period	Start	Finish	Status
1	08:00:00	09:00:00	NOT STARTED
2	09:00:00	10:00:00	NOT STARTED
3	10:00:00	11:00:00	NOT STARTED
4	11:00:00	12:00:00	NOT STARTED
5	12:00:00	13:00:00	NOT STARTED
6	13:00:00	14:00:00	NOT STARTED
7	14:00:00	15:00:00	NOT STARTED
8	15:00:00	16:00:00	NOT STARTED
9	16:00:00	17:00:00	NOT STARTED
10	17:00:00	18:00:00	NOT STARTED

Figure 5. SAS Settings screen

The E-Settlement administrator can specify the course of the E-Settlement day (start and length of number of reconciliation periods; these together indicate the end of the day) in this screen. The start time refers always to the current calendar day in the Prototype.

### 2.1.4.1 Fields

**Start of E-settlement day:** The start of the E-Settlement day. This is given in hour:minute or hour.minute.second format, such as 8:00 or 07.30.

**Period in minutes:** the length of each reconciliation period in whole minutes.

**Number of periods:** the number of reconciliation periods in the E-Settlement day.

### 2.1.4.2 Buttons

**Recalculate:** recalculates the reconciliation periods. Note that this button is disabled if a module has already been opened. This is because the reconciliation periods must be shared by all modules throughout the E-Settlement day. As a consequence, any changes in reconciliation period settings need to be done before the first module is opened.

### 2.1.4.3 Columns

**Period:** The number of the reconciliation period during the day.

**Start:** The start of the reconciliation period.

**Finish:** The end of the reconciliation period. The end timestamp of the last reconciliation period also marks the end of the E-Settlement day.

**Status:** The status of the period. The status can be one of the following:

- NOT STARTED: The period has not yet been started.
- IN PROGRESS: The period is currently underway.

- RECONCILING: The period has recently finished and bank-to-bank reconciliation is currently taking place.
- OK: Bank-to-bank reconciliation for the period has been finished successfully.
- ERROR: Bank-to-bank reconciliation for the period has been finished with errors. Log files of the modules have to be studied to find the cause of the problems.

### 2.1.5 Fix problems

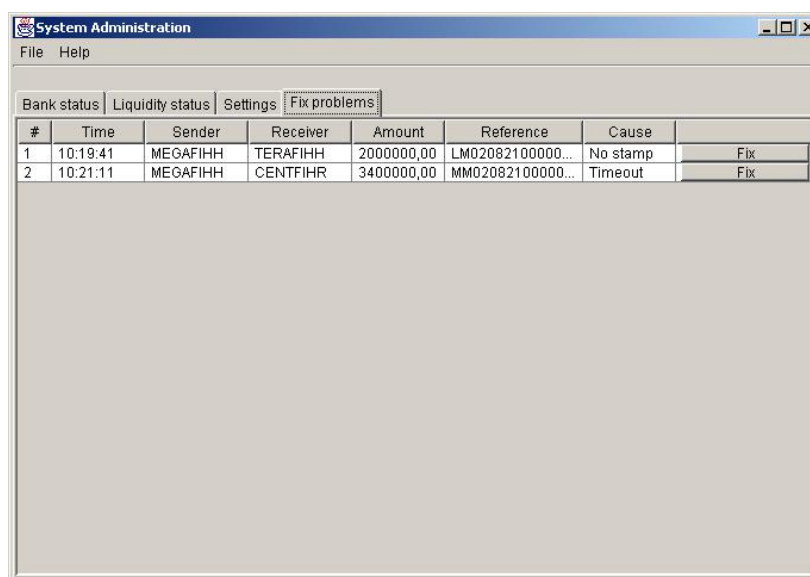


Figure 6. SAS Fix problems screen

This screen allows the SAS administrator to correct settlement problems resulting in bank's liquidity being tied to a settlement transaction. There are two kinds of problems that require this kind of correction in the Prototype:

- Payment timeout: the sending bank has formed the E-Settlement stamp and thus tied liquidity to the transaction, but neither a negative or positive confirmation has been received from the receiver bank in due time.
- E-Settlement stamp related problems: due to a network problem, attempted fraud or some other reason, the receiving bank is unable to decrypt the received E-Settlement stamp and cannot thus form the confirmation stamp. As a result, the sending bank cannot complete the payment process, and this again results in liquidity being tied to a transaction.

The operation is not automatic in the Prototype. The idea is that the SAS administrator must first check the situation with other means, such as by phone. Problems resulting in liquidity being tied to pending transactions are always severe and possibly reflect a fault in the E-Settlement system. Therefore the system cannot be totally relied upon in cases like this.

#### 2.1.5.1 Fields

#### 2.1.5.2 Buttons

**Fix:** Start fixing the problem. This results in a corrective E-Settlement stamp being sent to both sending and receiving modules.

### 2.1.5.3 Columns

**#:** The number of the transaction in the table

**Time:** The timestamp of the transaction

**Sender:** The SWIFT BIC of the sender of the transaction

**Receiver:** The SWIFT BIC of the receiver of the transaction




**Amount:** The amount of liquidity being transferred in the transaction

**Reference:** The unique ID of the settlement transaction

**Cause:** The reason why this transaction needs fixing. This is either “No stamp”, meaning that a confirmation, but no confirmation stamp, was received, or “Timeout”, meaning that no confirmation at all was received from the receiving bank.

## 2.2 Bank driver application

The state of the bank module is shown as a “traffic light” symbol in the status row of the application. The table below shows the meaning of these symbols. The color shown corresponds the status shown in SAS/Central Bank status screen and Bank participants screen. See 2.1.2 for the description of module states. In addition to these symbols, the current liquidity situation of the bank is shown in the status line.

Symbol			
Meaning	Module is in “Open” state	Module is in “Ready” state	Module is in “Stop sending” state.

### 2.2.1 Bank liquidity situation

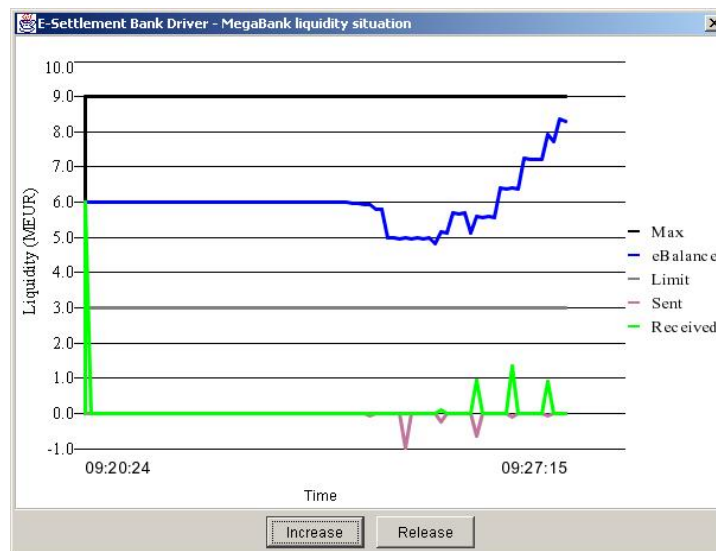


Figure 7. BADR liquidity situation window and liquidity issuing/de-issuing dialog box.

The liquidity situation of the bank is shown graphically as a function of time in a separate window. The Y axis represents liquidity in millions of Euros (MEUR) and the X axis represents time starting from the initial liquidity upload of the bank. The window can be closed from the button at the upper right side of the window, and opened again from Bank application’s View\Liquidity menu option. The window is by default refreshed every five seconds.

The graph has the following five lines:

- Max (black): intraday credit plus the deposits of the bank. This represents the maximum amount of liquidity that can be requested from the Central Bank at BOD.
- eBalance (blue): the current liquidity balance of the bank.

- Limit (light gray): the amount of intraday credit from the Central Bank available to the bank.
- Sent (red): The amount of liquidity sent by the bank to other banks since the graph was previously updated.
- Received (green): The amount of liquidity received by the bank from other banks.

Bank's credit and deposits, as well as the amount of liquidity that will be transferred at BOD, are maintained in the bank's central bank (see 2.3.1).

### 2.2.1.1 Fields

**Amount MEUR:** Millions of Euros for liquidity issuing / de-issuing request.

### 2.2.1.2 Buttons

**Increase:** Request more liquidity from the bank's Central Bank. Opens the liquidity issuing / de-issuing dialog box.

**Release:** De-issues some liquidity back to Central Bank. Opens the liquidity issuing / de-issuing dialog box.

**Send Request:** Performs the liquidity issuing / liquidity de-issuing request. If not enough deposits and credit are available at the central bank to fulfil a liquidity request, a message box for this effect is shown.

**Cancel:** Cancels the operation.

### 2.2.1.3 Columns

## 2.2.2 Payment details

TRN ref	LM02080900000000MEGAFIHH
Sender	MEGAFIHH
Receiver	GIGAFIHH
Amount	1000000,00
Timestamp	10:03.40.442
Payer account	
Beneficiary account	
Priority	10
Status	SENDER COMPLETED
Repeat count	0
Compensated payment	
Compensating payment	
Started	10:03.41.162
Finished	10:05.06.540
Settlement reference	AAAAZT/amYsJuSaiAAA743TVag=
Error code	INVALID DATA

Figure 8. Payment details dialog box

This dialog box can be opened by double-clicking the first column Queue (see 2.2.3), Pending (see 2.2.4) and Bookings (see 2.2.5) screens. It presents the detailed payment information. The information cannot be changed in this screen.

#### 2.2.2.1 Fields

**TRN ref:** The unique transaction reference of the payment. This is generated when the payment is input to the bank's payment queue.

**Sender:** The SWIFT BIC of the sending bank.

**Receiver:** The SWIFT BIC of the receiving bank.

**Amount:** The sum of the transaction in EUR.

**Timestamp:** The original timestamp (in milliseconds) of the payment, i.e. the timestamp when the payment was originally meant to be processed.

**Payer account:** Payer's account number, if any.

**Beneficiary account:** Beneficiary's account number, if any.

**Priority:** Payment's priority (not used at the moment – payments are in strict FIFO queue).

**Status:** A textual description of the status of the payment.

**Repeat count:** The number of times this payment has been resent by the bank.

**Compensated payment:** The TRN ref of the payment that is being compensated by this payment.

**Compensating payment:** The TRN ref of the payment that has compensated this payment.

**Started:** The timestamp (in milliseconds) of the actual start of processing of this payment.

**Finished:** The timestamp (in milliseconds) of the end of processing of this payment.

**Settlement Reference:** The E-Settlement reference of this payment. The E-Settlement Stamp of the payment can be found by this reference.

**Error code:** If an error has occurred during the processing of the payment, this field contains a textual description of the error. If no error has occurred, this field says "NO ERROR".

#### 2.2.2.2 Buttons

**OK:** Closes the dialog box.

#### 2.2.2.3 Columns

### 2.2.3 Payment queue

#	Time	Beneficiary	Amount	TRN Ref	Demo Error	Delete
1	15:38:50	GIGAFIHH	755023,00	MM02082100000...	None	Delete
2	15:38:54	GIGAFIHH	491410,00	MM02082100000...	None	Delete
3	15:38:58	GIGAFIHH	138960,00	LM020821000000...	None	Delete
4	15:39:02	GIGAFIHH	365601,00	SM02082100000...	None	Delete
5	15:39:06	GIGAFIHH	735866,00	LM020821000000...	None	Delete
6	15:39:09	TERAFIHH	485682,00	MM02082100000...	None	Delete
7	15:39:10	GIGAFIHH.13...	540738,00	LM020821000000...	None	Delete
8	15:39:13	TERAFIHH	499877,00	CM02082100000...	None	Delete
9	15:39:14	GIGAFIHH	434088,00	CM02082100000...	None	Delete
10	15:39:17	TERAFIHH	92913,00	CM02082100000...	None	Delete
11	15:39:18	GIGAFIHH	152519,00	LM020821000000...	None	Delete
12	15:39:21	TERAFIHH	209455,00	CM02082100000...	None	Delete
13	15:39:22	GIGAFIHH	58337,00	MM02082100000...	None	Delete
14	15:39:25	TERAFIHH	253155,00	MM02082100000...	None	Delete
15	15:39:26	GIGAFIHH	327645,00	CM02082100000...	None	Delete
16	15:39:29	TERAFIHH	639226,00	CM02082100000...	None	Delete
17	15:39:30	GIGAFIHH	512661,00	LM020821000000...	None	Delete
18	15:39:33	TERAFIHH	472453,00	LM020821000001...	None	Delete
19	15:39:34	GIGAFIHH	289986,00	MM02082100000...	None	Delete

New Import

Liquidity: 409968,00

Insert new payment

Time: 9:43

Receiver: MEGAFIHH

Amount MEUR: 38

OK Cancel

Figure 9. BADR Queue screen and Insert new payment dialog box

The queue screen shows the transaction queue of the bank. The payments on this screen have not yet been sent, i.e. no E-Settlement stamp has been generated for these payments. Payments are sent from this queue in real time according to the Time column of the table. The queue is sorted according to that column.

The payments in the queue are color coded as follows:

- Black: The payment has been queued and the receiving bank is currently in “Open” state, i.e. can receive payments. Payments like this will be sent normally according to their timestamp.
- Light gray: The receiver of the payment is not able to process the payment. It is either in “Unreachable”, “Ready” or “Stop sending” state. This can be checked from the BADR Participants screen. Payments like this will not be sent for further processing until the receiver reaches “Open” state.
- Blue: The payment should have been sent for further processing, but at the moment there is not enough liquidity to process it. Usually only one payment is in this state, but in high-load situations several payments can be in this state.

Payments can be input to the queue one by one by pressing the “New” button, or in batches by pressing the “Import” button. The payment batches are stored in an applied RJE file format – see Appendix 1 for the description of this format.



### 2.2.3.1 Fields

**Time:** The timestamp of the new payment. This is given in hour:minute:second or hour.minute format.

**Receiver:** The receiver of the new payment can be chosen from this list. It contains all system participants minus the bank itself. The SWIFT BIC code is presented for each system participant.

**Amount MEUR:** The sum of the new payment in millions of Euros. Both dot (.) and comma (,) can be used as the decimal separator.

**Demo Error:** A person giving a demonstration of the Prototype can choose various explicitly caused error situations for a payment. “No error” means that the payment will be processed normally without any additional errors. Following kinds of “demo errors” can be chosen for a payment:

- Do not send: an E-Settlement stamp is requested for the payment, but the payment never gets sent to the receiving bank. This results in payment timeout which has to be rectified by SAS using the “fix problems” functionality, see 2.1.5.
- Tamper stamp: an E-Settlement stamp is requested for the payment, but before the payment gets sent to the receiver, the contents of the stamp are modified. As a result, the E-Settlement module of the receiving bank is unable to decrypt and process the payment, and is thus unable to form the confirmation stamp for the payment. This results in “no stamp” error in the sending module, a situation which has again to be corrected by SAS “fix problems” functionality, see 2.1.5.
- Wrong BIC: before the payment is sent to the receiving bank, the receiver’s BIC of the payment is modified. This results in the receiving bank rejecting the payment.
- Tamper payment: before the payment is sent to the receiving bank, the hash code of the payment is modified. This also results in the receiving bank rejecting the payment.

### 2.2.3.2 Buttons

**New:** Opens the “Insert new payment” dialog box for manually entering the a new payment to the bank’s payment queue.

**Import:** Opens a file import dialog box for reading in a batch of payments in applied RJE format (see Appendix 1) to the bank’s payment queue. Only payment batches originating from this bank can be read to the queue, others are rejected. Furthermore, payments that would take place after the end of the E-Settlement day will be skipped if found in the input file.

**OK:** Checks the input, and if successful, inserts the new payment to the bank’s payment queue.

**Cancel:** Cancels the new payment input operation.

**Delete:** Deletes the payment from the queue. To prevent accidental deletions, the operation has to be confirmed by the user.

### 2.2.3.3 Columns

**#:** The number of the payment in the queue. By double-clicking this column the Payment details dialog box (see 2.2.2) can be opened.

**Time:** The scheduled time of the payment. This field is editable, allowing the user to reorder the queue.

**Beneficiary:** The SWIFT BIC of the receiver of the payment and possible account number of the beneficiary (if present at the input file).

**Amount:** The sum of payment in EUR. This field is editable.

**TRN Ref:** The unique identifier for this payment. It is formed when the payment is input to the bank’s payment queue.

## 2.2.4 Pending payments

#	Time	Beneficiary	Amount	TRN Ref	Status	Resend
1	10:11:57	GIGAFIHH	34280,00	MM020809000...	SENT TO RECEIVER	Resend
2	10:12:37	GIGAFIHH	33191,00	CM020809000...	PAYMENT SENT TO BEM	Resend

Liquidity: 6249362,00

Figure 10. BADR Pending payments screen

This screen shows the ongoing transactions – both those that are being sent and received - to the user. The current status of each transaction is shown. Pending transactions can also be resent from this screen. To emphasize the status of transactions, the following color scheme is used:

- Black: normally proceeding transaction
- Yellow: this transaction has timed out at least once and has been automatically resent
- Cyan: this transaction has exceeded the bank profile “number of payments” check and needs to be approved by the bank’s central bank. See 2.3.3 for more information on profile checks.
- Magenta: this transaction has exceeded the bank profile “sum” test and needs to be approved by the bank’s central bank. See 2.3.3 for more information on profile checks.

### 2.2.4.1 Fields

### 2.2.4.2 Buttons

**Resend:** request explicit resending of the transaction in question. Transactions will be resent automatically, too. By default the payment timeout is 30 seconds and transactions will be resent four times before they are considered timed out for good. If this happens, the payment ends up in SAS Fix problems screen.

### 2.2.4.3 Columns

**#:** The number of the payment in the pending payments list. By double-clicking this column the Payment details dialog box (see 2.2.2) can be opened.

**Time:** The scheduled time of the payment.

**Beneficiary:** The SWIFT BIC of the receiver of the payment and possible account number of the beneficiary, if present in the input file.

**Amount:** The sum of the payment in EUR.

**TRN Ref:** The unique identifier for this payment.

## 2.2.5 Bookings

#	Time	Duration sec	Payer	Beneficiary	Amount	TRN Ref	
1	09:39:09		CENTFIHR	MEGAFIHH	+32000000,00	MM0208220...	
2	09:40:00	9.141	MEGAFIHH	GIGAFIHH	-3700000,00	MM0208220...	Request compen...
3	09:40:40	34.266	MEGAFIHH	GIGAFIHH	-7500000,00	LM0208220...	
4	09:41:47	1.766	MEGAFIHH	GIGAFIHH:47...	-511225,00	SM0208220...	Request compen...
5	09:41:51	5.906	MEGAFIHH	GIGAFIHH	-527700,00	CM0208220...	Request compen...
6	09:41:55	2.344	MEGAFIHH	GIGAFIHH	-640978,00	CM0208220...	Request compen...
7	09:41:59	1.735	MEGAFIHH	GIGAFIHH	-342723,00	SM0208220...	Request compen...
8	09:42:03		GIGAFIHH	MEGAFIHH	+22693,00	SM0208220...	
9	09:42:03	2.594	MEGAFIHH	GIGAFIHH	-410829,00	CM0208220...	Request compen...
10	09:42:07	2.516	MEGAFIHH	GIGAFIHH:91...	-758258,00	SM0208220...	Request compen...
11	09:42:07		GIGAFIHH	MEGAFIHH	+752277,00	MM0208220...	
12	09:42:11	2.891	MEGAFIHH	GIGAFIHH:16...	-35485,00	SM0208220...	Request compen...
13	09:42:11		GIGAFIHH:75...	MEGAFIHH	+829220,00	CM0208220...	
14	09:42:15	3.297	MEGAFIHH	GIGAFIHH	-755023,00	CM0208220...	Request compen...
15	09:42:15		GIGAFIHH:18...	MEGAFIHH	+37452,00	CM0208220...	
16	09:42:19	2.266	MEGAFIHH	GIGAFIHH	-481410,00	MM0208220...	Request compen...
17	09:42:19		GIGAFIHH	MEGAFIHH	+115640,00	SM0208220...	
18	09:42:23	1.922	MEGAFIHH	GIGAFIHH	-138960,00	MM0208220...	Request compen...
19	09:42:23		GIGAFIHH	MEGAFIHH	+65849,00	CM0208220...	
20	09:42:27	2.157	MEGAFIHH	GIGAFIHH	-365601,00	CM0208220...	Request compen...
21	09:42:27		GIGAFIHH:67...	MEGAFIHH	+684168,00	MM0208220...	
22	09:42:27	2.275	MEGAFIHH:12...	GIGAFIHH	726966,00	MM0208220...	Request compen...

Figure 11. BADR Bookings screen

This screen shows the completed payments of the bank. Both sent and received transactions are shown. Successful transactions are shown in black and failed transactions in red. “Failed” in this respect means that the transaction in question has not involved any liquidity transfer. The user has also the possibility to request compensation for a payment, which in effect means making another transaction in the “opposite direction”.

### 2.2.5.1 Fields

### 2.2.5.2 Buttons

**Request compensation:** “cancel” a completed, successful transaction by requesting the receiver of the transaction to form a transaction in opposite direction for the same amount. In the Prototype the receiver automatically agrees to compensation transactions when requested.

### 2.2.5.3 Columns

**#:** The number of the payment in the bookings list. By double-clicking this column the Payment details dialog box (see 2.2.2) can be opened.

**Time:** The scheduled time of the payment.

**Duration sec:** The time, in seconds, it took for the transaction to complete from start to finish.

**Payer:** The SWIFT BIC of the payer’s bank and possible account number of the payment, if present in the input file.

**Beneficiary:** The SWIFT BIC of the receiver of the payment and possible account number of the beneficiary, if present in the input file.

**Amount:** The sum of payment in EUR.

**TRN Ref:** The unique identifier for this payment.

## 2.2.6 Statistics

Queue	Pending	Bookings	Statistics	Participants	Settings	Reconciliation
Received			101			26102357,00
Sent			100			18906148,00
Total			201			
CentralBank						
Sent			0			0,00
Received			1			8000000,00
TeraBank						
Sent			50			8010227,00
Received			50			9934105,00
MegaBank						
Sent			50			10895921,00
Received			50			8168252,00

Liquidity: 7196209,00

Figure 12. BADR Statistics screen

This screen shows the cumulative settlement statistics for the bank. Transaction numbers and sums are shown grouped by system participant (bottom), and total cumulative (top) statistics are presented.

### 2.2.6.1 Fields

**Received:** Number and sum of transactions received from the given system participant.

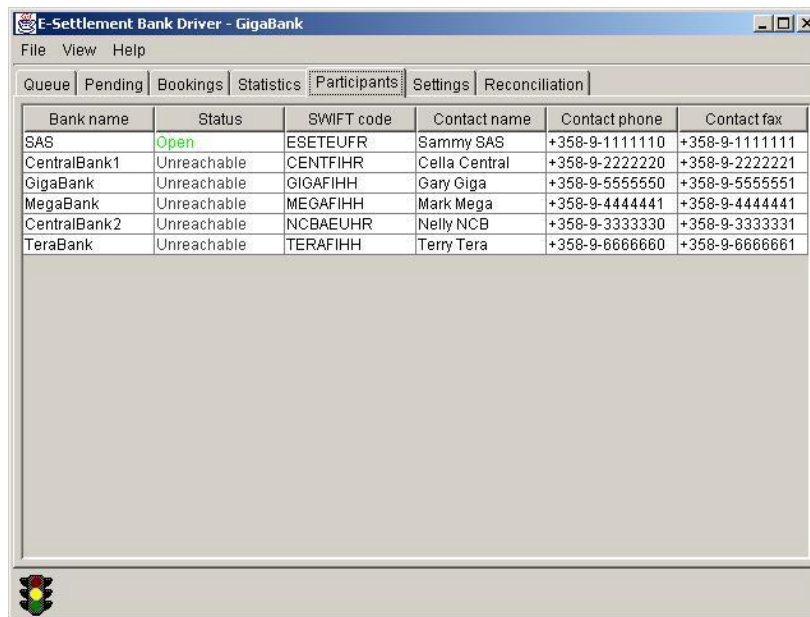
**Sent:** Number and sum of transactions sent to the given system participant

**Total:** Total number of transactions

### 2.2.6.2 Buttons

### 2.2.6.3 Columns

## 2.2.7 Participants



Bank name	Status	SWIFT code	Contact name	Contact phone	Contact fax
SAS	Open	ESETEUFR	Sammy SAS	+358-9-1111110	+358-9-1111111
CentralBank1	Unreachable	CENTFIHR	Cella Central	+358-9-2222220	+358-9-2222221
GigaBank	Unreachable	GIGAFIHH	Gary Giga	+358-9-5555550	+358-9-5555551
MegaBank	Unreachable	MEGAFIHH	Mark Mega	+358-9-4444441	+358-9-4444441
CentralBank2	Unreachable	NCBAEUHR	Nelly NCB	+358-9-3333330	+358-9-3333331
TeraBank	Unreachable	TERAFIHH	Terry Tera	+358-9-6666660	+358-9-6666661

Figure 13. BADR Participants screen

This screen shows the status and contact information of all system participants, including SAS, to bank users.

### 2.2.7.1 Fields

### 2.2.7.2 Buttons

### 2.2.7.3 Columns

**Bank name:** The name of the system participant

**Status:** The status of the system participant. See 2.1.2 for a closer description of possible states and colours.

**SWIFT code:** The SWIFT BIC of the system participant.

**Contact name:** Contact person name for the given system participant.

**Contact phone:** Contact person's phone number for the system participant.

**Contact fax:** Contact person's fax number for the system participant.

## 2.2.8 Settings

Period	Start	Finish
1	08:00:00	09:00:00
2	09:00:00	10:00:00
3	10:00:00	11:00:00
4	11:00:00	12:00:00
5	12:00:00	13:00:00
6	13:00:00	14:00:00
7	14:00:00	15:00:00
8	15:00:00	16:00:00
9	16:00:00	17:00:00
10	17:00:00	18:00:00

Figure 14. BADR Settings screen

This screen shows the main configuration information affecting the bank's activities. These include bank profile sum and profile number as well as the reconciliation periods of the E-Settlement day currently in progress. See 2.3.3 for more information on bank profile and 2.1.4 for a closer description of the reconciliation periods. The profile is controlled by the bank's central bank, and the E-Settlement reconciliation periods are defined by SAS during BOD activities.

### 2.2.8.1 Fields

**Profile sum:** This is the cumulative sum of transactions a bank can send in a 60-second window before the transactions need to be approved by the bank's central bank.

**Profile number:** This is the number of transactions a bank can send in a 60-second window before the transactions need to be approved by the bank's central bank.

### 2.2.8.2 Buttons

### 2.2.8.3 Columns

**Period:** Number of the reconciliation period

**Start:** Start of the reconciliation period

**Finish:** End of the reconciliation period. The end timestamp of the last reconciliation period also marks the end of the E-Settlement day.

## 2.2.9 Reconciliation

Bank Name	Transaction Type	Count	Sum
CentralBank	Sent	0	0,00
CentralBank	Received	1	10000000,00
GigaBank	Sent	0	0,00
GigaBank	Received	19	3314260,00
MegaBank	Sent	20	2506275,00
MegaBank	Received	0	0,00

Figure 15. BADR Reconciliation screen

This screen shows the E-Settlement statistics for the completed reconciliation periods, and at EOD, the statistics for the completed E-Settlement day. The difference to the statistics screen (see 2.2.6) is that the reconciliation period statistics only change once per reconciliation period and show per-period statistics, whereas the statistics screen always shows cumulative statistics from the beginning of the day.

The text above the per-bank statistics table shows the reconciliation period, the statistics of which are currently being shown.

### 2.2.9.1 Fields

**Total received:** Total number and sum of transactions received during the period

**Total sent:** Total number and sum of transactions sent during the period

**Total and received:** Total number of transactions during the period

**Received:** Number and sum of transactions received from the given system participant during the period.

**Sent:** Number and sum of transactions sent to the given system participant during the period.

### 2.2.9.2 Buttons

### 2.2.9.3 Columns

## 2.3 Central Bank driver application

For description of the status bar symbols in the central bank application, see chapter 2.2. They are identical to those of the bank application. The only difference is that the central bank can (and usually will have) a negative liquidity balance.

### 2.3.1 Liquidity

Bank	Deposits	Credit	Total liquidity	Initial liquidity	Issued
TeraBank	8000000,00	5000000,00	13000000,00	10000000,00	10000000,00
GigaBank	7000000,00	4000000,00	11000000,00	8000000,00	8000000,00
MegaBank	6000000,00	3000000,00	9000000,00	6000000,00	6000000,00

Figure 16. CBDR Liquidity screen

This screen is used to manage and monitor the deposit, credit and liquidity related information for the banks under the central bank's control.

Note: initial liquidity transfers and all other transactions between central banks and banks affect the amount of liquidity being issued to the system. Liquidity transfers are normal settlement transactions from a central bank to bank.

In the Prototype the banks' final positions at EOD can be studied from the "Issued" column of this table. Negative number at EOD means that the bank is to receive liquidity from other banks.

#### 2.3.1.1 Fields

#### 2.3.1.2 Buttons

#### 2.3.1.3 Columns

**Bank:** The name of the bank.

**Deposits:** The amount of deposits stored by the bank to the central bank in EUR. This field is editable.

**Credit:** The amount of intraday credit available to the bank in EUR. This field is editable.

**Total liquidity:** Bank's deposits and credits in total in EUR. This number represents the maximum amount of liquidity that can issued to bank at BOD.

**Initial liquidity:** The amount of liquidity that will be transferred to the bank during beginning-of-day activities in EUR. Different banks can have different strategies with respect to the amount of liquidity they wish to use for E-Settlement purposes. This field is editable.



**Issued:** The amount of liquidity that has been issued to the bank during the day. All transactions conducted between the central bank and the bank change this number.

### 2.3.2 Bank status

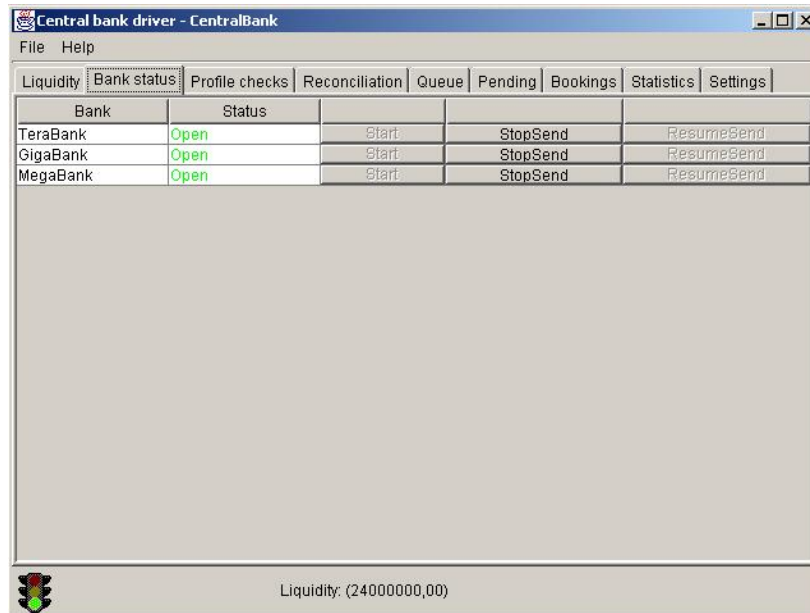


Figure 17. CBDR Bank status screen

This screen is similar to the SAS status screen (see 2.1.2), except that only the banks under the central bank's control can be managed from this screen.

### 2.3.3 Profile checks

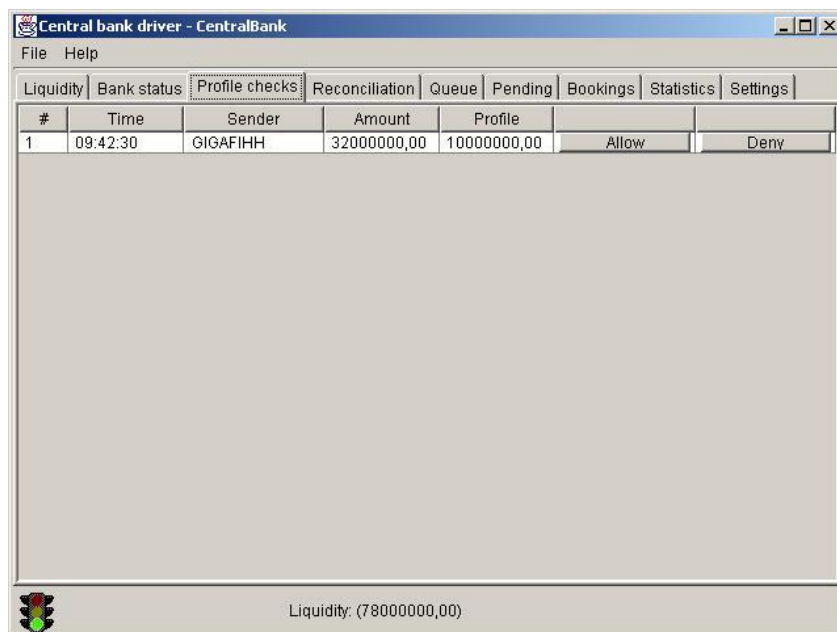


Figure 18. CBDR Profile checks screen

This screen is used for bank profile monitoring. Bank profile is a built-in security feature in E-Settlement system. The idea with the bank profile is to set, based on existing settlement statistics, limits to “normal” sum of and number of transactions that can be conducted by a bank in a period of time. If a bank tries to send too large a transaction, or too many transactions in a short period of time, the central bank is given a possibility to prevent these transactions from taking place. In the Prototype, each transaction needs to be approved or denied separately.

The bank profile consists of two elements:

- **Sum of transactions:** This is the sum transactions a bank can send in a 60-second window before the transactions need to approved by the bank’s central bank. For example, a single transaction of 11 MEUR, and the latter of two transactions valued 6 MEUR and 7 MEUR transactions sent in 10 second intervals, would need approval from bank’s central bank if the limit was 10 MEUR.
- **Number of transactions:** This is the number of transactions a bank can send in a 60-second window before the transactions need to approved by the bank’s central bank. For example, if the bank profile number is 100 and a bank tried to send 110 transactions in a minute, the last 10 transactions would be sent to the bank’s central bank for approval.

In the Prototype the bank profiles are maintained in SDI through the tools provided.

#### 2.3.3.1 Fields

#### 2.3.3.2 Buttons

**Allow:** Acknowledge to transaction, i.e. allow it to continue normally.

**Deny:** Do not allow the transaction to proceed.

#### 2.3.3.3 Columns

**#:** The number of the payment in the list of profile checks.

**Time:** The scheduled time of the payment.

**Sender:** The SWIFT BIC of the sender of the payment.

**Amount:** The sum of the payment in EUR.

**Profile:** The profile sum of the bank in question.

## 2.3.4 Reconciliation

Statistics for completed reconciliation period 09:42:00-09:48:00		
CentralBank		
Sent	3	24000000,00
Received	0	0,00
TeraBank		
Sent	20	2506275,00
Received	20	13314260,00
GigaBank		
Sent	19	3314260,00
Received	20	10368196,00
MegaBank		
Sent	19	2368196,00
Received	21	8506275,00

Liquidity: (24000000,00)

Figure 19. CBDR Reconciliation screen

This screen shows the statistics for completed reconciliation periods, and at EOD, the statistics for the completed E-Settlement day for the banks under the central bank's control. The difference to CBDR Statistics screen (see 2.3.8) is that whereas the CBDR statistics screen shows the central bank's perspective from the beginning of day, this screen shows the statistics for the whole central bank area.

The text above the per-bank statistics table shows the reconciliation period, the statistics of which are currently shown.

### 2.3.4.1 Fields

**Received:** Number and sum of transactions received by the given system participant during the period.

**Sent:** Number and sum of transactions sent by the given system participant during the period.

### 2.3.4.2 Buttons

### 2.3.4.3 Columns

## 2.3.5 Payment queue

This screen is identical to the corresponding BADR screen, see chapter 2.2.3. The only difference between BADR and CBDR queue is that no liquidity checks are performed for central banks, i.e. a central bank never runs out of liquidity in the Prototype.

## 2.3.6 Pending payments

This screen is identical to the corresponding BADR screen, see chapter 2.2.4.

## 2.3.7 Bookings

This screen is identical to the corresponding BADR screen, see chapter 2.2.5.

### **2.3.8 Statistics**

This screen is identical to the corresponding BADR screen, see chapter 2.2.6.

### **2.3.9 Settings**

This screen is identical to the corresponding BADR screen, see chapter 2.2.8.

## 3 USING THE PROTOTYPE

This chapter presents an overview of how to start up, configure and use the Prototype.

### 3.1 System start-up and configuration

The Prototype is somewhat complex distributed software environment. Therefore one must be careful in starting up and configuring the system.

#### 3.1.1 Booting up the system

In order to ensure proper system start-up, the servers in the Prototype Environment must be booted and services started in the proper order:

1. The domain controller must be started first and allowed to boot up properly
2. The SAS computer must be started next and allowed to boot up properly
3. The rest of the computers can be started in any order
4. After all servers have been started, log in to all servers
5. Check that all IPSec connections are up and running
6. Check that SDI is running properly at SAS server. Connect to SDI to check.
7. Start CA. This requires knowing the CA password and that a connection to SDI is made with proper credentials. Check that the CA is publishing CRLs to SDI on a regular basis.

Once these steps have been accomplished, the E-Settlement modules can be started up.

#### 3.1.2 Configuring system participants

More participants (both Central Banks and Banks) can be added to the Prototype as required. The following is needed for each participant:

- Identity information: unique bank name, plus unique E-Settlement IDs for the driver, module and module HSM, and an unique SWIFT code. These are stored in SDI.
- Hardware: one computer (Windows 2000 professional plus Service Packs) for each system participant, equipped with a smart card reader.
- Software: E-Settlement module and driver installation, HSM emulator installation
- Security issues: PKCS#12 key store for the module, Smart card created from the PKCS#12 key store of the HSM, module and HSM public certificates published to the system directory
- The configuration files presented in Appendix 2

In addition, the System Directory must be configured accordingly. SDI contains information regarding the physical addresses of the system participants, the hierarchy of central banks and banks and participants' configuration information such as bank names and bank profiles.

## 3.2 Planning a scenario

Different kinds of scenarios can be tried out using the Prototype. Running a specific demonstration or simulation scenario requires some planning beforehand. The following are the main steps in planning a scenario:

- Decide on the length of the E-Settlement day. The Prototype runs in real time, so the length of the day depends on the nature of the demonstration or simulation.
- Generate payment flows for the scenario. Simple, even payment flows can be generated with a tool provided with the E-Settlement system, see Appendix 3. More complex payment flows can be formed by combining simpler payment flows.
- Define BOD liquidity situation. Decide on how much liquidity each bank will have at their disposal, and available at the beginning of day.
- Decide the “problem situations”, if any, that should be demonstrated. For example, one bank might be set to “Stop sending” state for a certain period during the day.

## 3.3 Running a scenario

### 3.3.1 Starting

For each CEM and BEM taking part in the scenario, the module, driver and HSM emulator must be started. In addition, SAS must be started. The start-up order is insignificant. Once all required modules, drivers and HSMs have been started, the SAS Status screen should show all system participants in “Ready” state and the Central Banks ready to be started, as follows:

Bank	Status	Start	StopSend	ResumeSend
CentralBank1	Ready	Start	StopSend	ResumeSend
GigaBank	Ready	Start	StopSend	ResumeSend
MegaBank	Ready	Start	StopSend	ResumeSend
CentralBank2	Ready	Start	StopSend	ResumeSend
TeraBank	Ready	Start	StopSend	ResumeSend

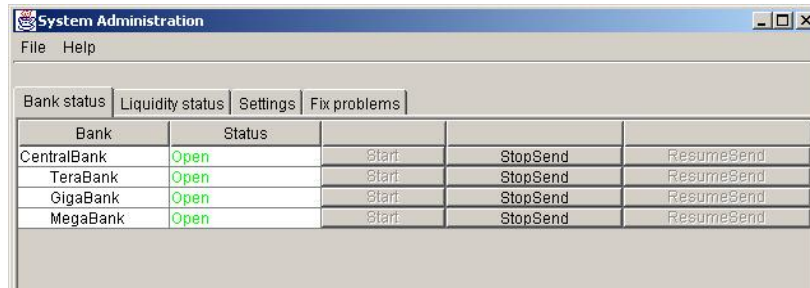
Figure 20. The prototype has been properly started.

### 3.3.2 Beginning-of-day activities

Beginning-of-day activities include the following:

1. Define the E-Settlement day in SAS Settings screen. Note that this cannot be changed once a module has been opened.
2. Configure banks' liquidity situation (deposits, credits, initial liquidity) in CBDRs' Liquidity screen
3. Open the bank modules and wait for initial liquidity upload to take place

Once the modules have been opened and the initial liquidity upload has taken place (this can take some time), SAS Status screen looks like this:



Bank	Status	Start	StopSend	ResumeSend
CentralBank	Open	Start	StopSend	ResumeSend
TeraBank	Open	Start	StopSend	ResumeSend
GigaBank	Open	Start	StopSend	ResumeSend
MegaBank	Open	Start	StopSend	ResumeSend

Figure 21. Modules have been successfully opened

Banks' liquidity situation screen should also show that the initial liquidity upload has taken place:

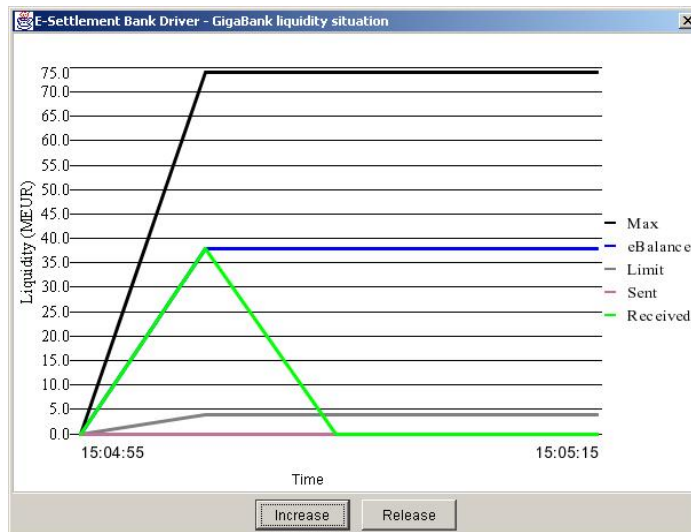


Figure 22. Initial liquidity upload has taken place

### 3.3.3 Running settlement scenarios

Generally speaking, one should follow the scenario one has planned. There are no steadfast rules as to how a settlement scenario, or a given E-Settlement day, should be run. There are two main “modes” in running the scenarios:

- Running transactions one by one. Transactions are input manually to banks' queues. This allows the audience to stay abreast of what is happening to a single transaction. Various liquidity situations and “demo error” situations can be best demonstrated in this mode.
- Importing pre-generated payment flows and following the behavior of the prototype “during the day”. Various liquidity scenarios and “Stop sending” situations can be perhaps better demonstrated in this way.

### 3.3.4 End-of-day activities

The end of E-Settlement day is an automatic operation that takes place according to the schedule set forth during beginning-of-day activities in SAS. During EOD activities, the banks de-issue their remaining liquidity and EOD statistics are calculated. After EOD activities have taken place, the banks' liquidity situation screen looks like this:

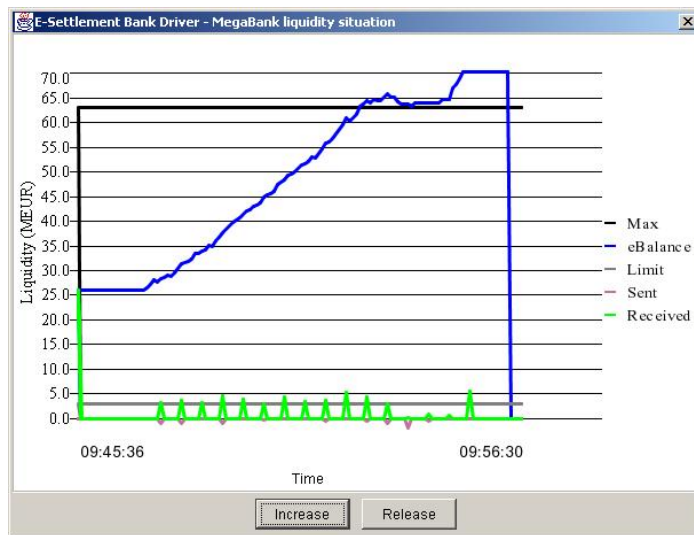


Figure 23. EOD liquidity de-issuing has taken place

CBDR liquidity screen after EOD shows the bank positions with regard to each other. For example, in the Figure below, TeraBank owes to GigaBank and MegaBank. In reality, the positions of the banks would have to be balanced, but the Prototype does not do this for clarity. Note also that the sum of liquidity in the system is zero.

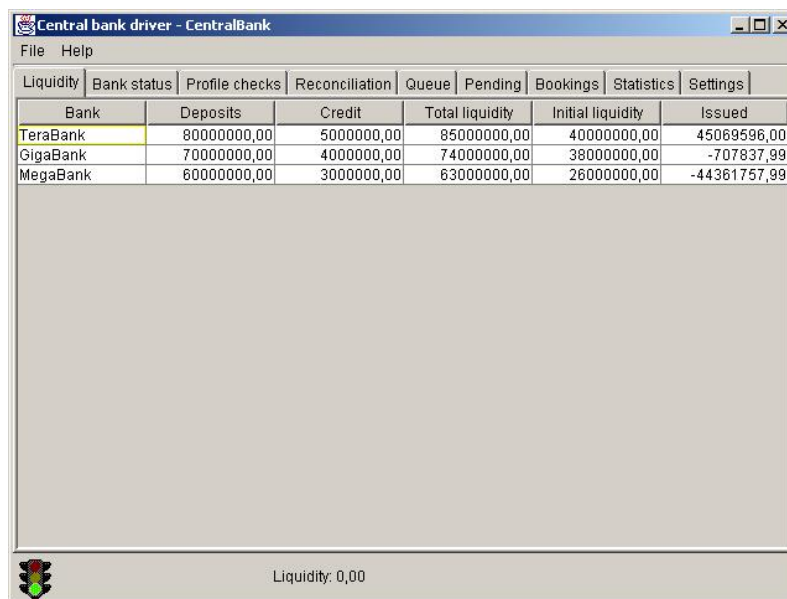


Figure 24. CBDR EOD liquidity situation



### 3.4 Troubleshooting

In a complex distributed software environment, many things can go wrong. What follows is a short checklist of troubleshooting instructions for the E-Settlement Prototype.

- Check that all required services up and running. IPSec, SDI and CA must all be running properly for the system to work.
- Check configuration. Usually if a service (module, driver, HSM) does not start properly, the configuration of the Prototype is incorrect. There can be a mistake in a start-up script, a configuration file or the SDI.
- Check log files. This is the best source of information for problems that take place in run time. Drivers, modules and HSM emulator all log activities to separate files. See Appendix 4 for the description of the log files.
- Check that HSM emulators are running and that smart cards have been inserted properly. This can be the cause of errors if module cannot be opened. Smart cards should not be removed from the readers during the E-Settlement day.
- Check that no processes are pending. If it seems that it is impossible to contact a module, a process might have been left pending from the previous run of the prototype. This can be checked from Task Manager.

## 4 PRELIMINARY EXPERIENCES

This chapter lists some preliminary experiences from the E-Settlement prototype implementation.

### 4.1 Implementation

The E-Settlement concept was specified and the Prototype implemented by two software companies under the supervision of Bank of Finland. Two software companies were used in the project to fulfill an “electronic four eyes principle”. This means that no single party has total knowledge of the security solution of the system. The project consisted of three phases:

- Specification phase: Nov 2001- Feb 2002. During this phase the core E-Settlement specifications were completed.
- Prototype implementation phase: March 2002 – May 2002. The actual implementation of the prototype took place in this phase. This phase was the most resource intensive of the phases.
- Integration and testing phase: Jun 2002- Aug 2002. The parts of the two software companies were integrated and preliminary experiences from prototype were gathered.

The table below summarizes the resource usage (in person days) of the various parties.

Phase	Bank of Finland	SW company 1 (E-Settlement system)	SW company 2 (HSM emulator, security solution)	Total
Specification	48	260	9	317
Prototype implementation	10	256	53	319
Integration and testing	14	44	15	73
Grand total	72	560	77	709

In addition, there have been some hardware (6 desktop computers) and software (6 operating system licenses plus server software license fees) costs.

The following numbers characterize the extent of the Prototype. It has to be born in mind, though, that the Prototype only has some of the error checking functionality of a real production environment. These numbers illustrate the “size” of the E-Settlement Prototype and at the same time give a picture of the possible production version of the E-Settlement core functionality.

- 420 Java classes, plus several off-the-shelf libraries
- 49000 lines of (commented) code

In all, the project was completed relatively well in schedule and budget. The most difficult parts during the Prototype implementation project proved to be integrating off-the-shelf servers and E-Settlement software to a functional whole, and managing the distributed system. On the other hand, several positive things about the Prototype implementation can be mentioned:

- The basic ideas of E-Settlement turned out to be quite easy to implement and worked as expected.
- The security architecture could be implemented as planned.
- The extent of E-Settlement core functionality is not that big.

## 4.2 Performance

The performance of the E-Settlement Prototype is quite much dependent on the hardware used to perform the computation intensive PKI encryption algorithms. There are three main possibilities here, two first of which have been tested during the Prototype work.

- CPU: the PKI operations are calculated in the servers' CPU. The security level in this solution is not as high as that of the smart card based solution, since the crucial HSM signing operation is done in CPU rather than HSM. This solution is much faster, though.
- Smart card: some PKI operations are calculated in HSM and some in CPU. This is secure but rather slow way of doing things, since the PKI operations in HSM, and HSM operations in general, tend to be quite slow. Technological development is making the smart cards much more effective in the future.
- Cryptographic accelerator card: PKI operations are calculated in a specially designed, tamper-resistant hardware accelerator card. This would be both secure and fast, but also more expensive.

The table below shows the “best case” performance achieved with the tried hardware configurations in trial run with 100 transactions from one bank to another. The “best case” performance here means that the first transaction has ended before the second one begins, i.e. no transaction queuing occurs. This is, of course, a highly unrealistic scenario, but does give an indication of the “raw” performance of the Prototype.

Hardware	Mean transaction time sec	Standard deviation sec
CPU	1.247	0.124
Smart card	13.23	1.52
Accelerator card <sup>1</sup>	0.15	0.05

<sup>1</sup> This is an estimate based on the commonly used rule of thumb: a cryptographic accelerator card brings tenfold increase in encryption operation speed.

## APPENDIX 1: APPLIED RJE FILE FORMAT

Payment batches can be input to the E-Settlement prototype using an applied RJE file format. Knowledge of the RJE file format as well as MT103 and MT202 messages is assumed in this description. An example RJE input file follows, with some excess formatting added to emphasize the RJE blocks and fields. In addition, the pieces of information crucial to the Prototype have been underlined.

```
{1:F01TERAFIHHXXX0024000001}
{2:I198CENTFIHRXXXXN}
{3:{108:SIMULATED}}
{4:103
  :20:ip7y0863e1k0q9hk5b1iacz6hrk5v4n41
  :32A:030405EUR3176
  :52A:/36934736476-861508
  :57A:/850433934-45021
  CENTFIHR
  :72:/REC/2,000010000
-}
$
{1:F01TERAFIHHXXX0024000001}
{2:I198MEGAFIHHXXXXN}
{3:{108:SIMULATED}}
{4:202
  :20:8t5ygwfw2tb0ciga0bz91eymebymbvgg21
  :32A:030512EUR249218
  :58A:MEGAFIHH
  :72:/REC/1,000020000
-}
```

Block 1 contains the SWIFT BIC of the sending bank. Block 2 contains the SWIFT BIC of the receiving bank. Payment batches can contain payments in either applied MT103 or MT202 format in RJE block 4. The type of the payment is immediately in the beginning of RJE block 4.

The following MT103 fields are utilized by the Prototype:

- 20: TRN ref of the transaction. Compulsory. This reference will not actually be used by the prototype, since a new reference will be created.
- 32A: date (six digits), currency (3 letters ISO code) and sum of transaction. Compulsory.
- 52A: beneficiary account. Optional.
- 57A: payer account (optional) and payer's BIC code (compulsory)
- 72: "/REC/", payment priority, ",", and the timestamp of the payment starting from midnight in milliseconds. When the payment batch is actually read into the Prototype, the current time of the day will be added to this. For example, if the batch is read in at 9:10:20 and the timestamp here says "000120000" (1 minute and 20 seconds), the resulting actual timestamp would be 9:11:40. Compulsory.

The following MT202 fields are utilized by the Prototype:

- 20: TRN ref of the transaction. Compulsory. This reference will not actually be used by the prototype, since a new reference will be created.
- 32A: date (six digits), currency (3 letters ISO code) and sum of transaction. Compulsory.
- 58A: beneficiary account (optional) and beneficiary bank's BIC code (compulsory)
- 72: "/REC/", payment priority, ",", and the timestamp of the payment starting from midnight. When the payment batch is actually read into the Prototype, the current time of the day will be added to this. Compulsory.

## APPENDIX 2: CONFIGURATION FILES

A number of configuration files are needed for each system participant. These can be easily copied from existing system participants' configuration files. The notation here is as follows:

- <ESET\_HOME> is the directory in which the E-Settlement software has been installed
- <HSM\_HOME> is the directory in which HSM emulator has been installed

The following files are needed by all elements. We use MegaBank as an example here.

- <ESET\_HOME>\bin\Mega\_badr\_start.bat – the driver start-up script
- <ESET\_HOME>\bin\start\_module\_Mega.bat – the module start-up script
- <ESET\_HOME>\conf\MegaBank.deploy - module deployment descriptor
- Module service definition files:
  - <ESET\_HOME>\conf\MegaBank\_ContainerService.xml
  - <ESET\_HOME>\conf\MegaBank\_HtmlAdaptorService.xml
  - <ESET\_HOME>\conf\MegaBank\_MessagingService.xml
  - <ESET\_HOME>\conf\MegaBank\_TimerService.xml
- HSM emulator configuration files
  - <HSM\_HOME>\config\HSMEmu.xml
  - <HSM\_HOME>\config\mPollux.xml

## APPENDIX 3. RJE GENERATION TOOL

A simple tool for generating RJE payment batch files is provided with the E-Settlement system. If <ESET\_HOME> is the directory in which the E-Settlement software has been installed, then the tool is “<ESET\_HOME>\bin\generate”.

Syntax of the command is as follows:

```
filename senderBIC receiverBIC number interval_sec initial_wait [type]
filename      = legal file name
senderBIC     = 8 letter BIC code of sending bank
receiverBIC   = 8 letter BIC code of receiving bank
number        = the number of payments to create, a decimal integer
interval_sec  = number of seconds between payments, a decimal integer
initial_wait  = number of seconds to wait before first payment, a decimal integer
type          = type of payments to generate, "MT103" (default) or "MT202"
```

As an example, the following command generates an input file named 10\_100\_4\_Mega\_Giga.rje from MegaBank to GigaBank, with 10 second wait before the first payment and then 100 payments in four second intervals. The payments will be of type MT103.

```
<ESET_HOME>\bin\generate <ESET_HOME>\data\10_100_4_Mega_Giga.rje MEGAFIHH GIGAFIHH 10 100 4
```

The division of generated payments is as follows 63% 0-50 KEUR and 37% 50 KEUR-1MEUR. 20% of the payments have an end user account in field 52A, 57A or 58A.

## APPENDIX 4. LOG FILES

The notation here is as follows:

- <ESET\_HOME> is the directory in which the E-Settlement software has been installed
- <HSM\_HOME> is the directory in which HSM emulator has been installed

BADR, CBDR, BEM, CEM and SAS store log files in <ESET\_HOME>\log directory. A log file name always contains the name of the element and a random number. In the list below, we use MegaBank and CentralBank as examples of bank and central bank names, respectively.

- BADR log files are named as follows: BADR\_MegaBank\_48746.log
- CBDR log files are named as follows: CBDR\_CentralBank\_56839.log
- SAS log files are named as follows: SAS\_32848.log
- BEM log files are named as follows: MegaBank\_23419.log
- CEM log files are named as follows: CentralBank\_34872.log

HSM emulator maintains two log files: one is “global” and on a more coarse level, and the other is a very detailed log file that is formed per each call of HSM emulator. The latter logs all access to the smart card, for example.

- <HSM\_HOME>\log\trace.txt is the global HSM emulator log
- <HSM\_HOME>\SC.log is the call-specific HSM emulator log

## APPENDIX 5. PROTOTYPE HARDWARE AND SYSTEM SOFTWARE

The PCs used are standard PCs with 1.5 GHz CPU, 1GB RAM and 40GB hard disk drive running Windows 2000 operating system and Java Runtime Environment version 1.3.1. The PCs are also equipped with a smart card reader.

The standard prototype configuration consists of three PCs representing banks systems and the bank e-settlement module (BEM), which are sending payments to each other (see picture 1). The payment messages formats in the prototype are MT103 and MT202.



*Picture 1: The bank system simulators and BEM PCs*

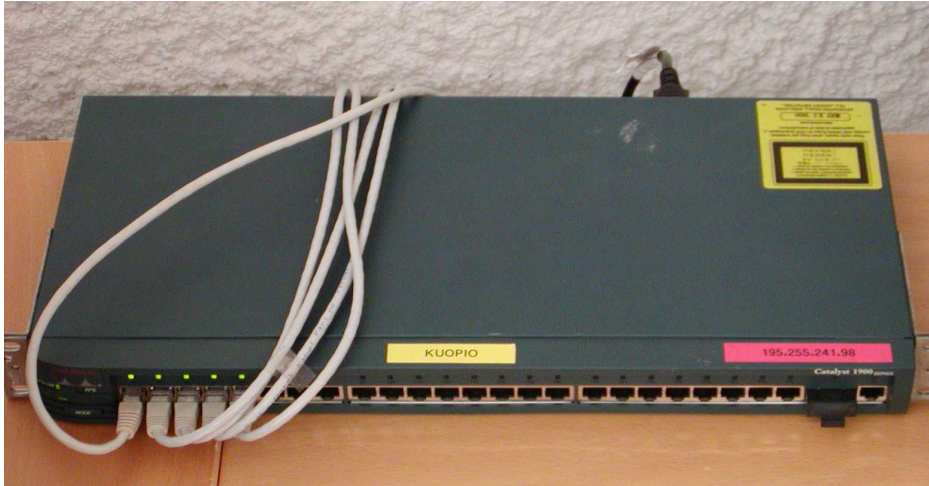
The system is controlled by the system administration site (SAS) and by the central bank using its CEM module (see picture 2). SAS includes off-the-shelf CA and directory products.



*Picture 2: The SAS and CEM modules*



For the TCP/IP network between the PCs and modules a standard 10 Mbps Ethernet switch was used (see picture 3).



*Picture 3: The TCP/IP switch*

The hardware security modules (HSM) are based on off-the-shelf PKI chip cards (see picture 4).



*Picture 4: The prototype HSM card*

## References

ECB issues paper (2002) **E-payments in Europe – the Eurosystem perspective**.  
September.

Leinonen, H. (2000) **Re-engineering payment systems for the e-world**.  
Discussion Proposal 17/2000, Bank of Finland.

Schneier, B. (1996) **Applied Cryptography Second Edition: protocols, algorithms, and source code**. In C. John Wiley & Sons, Inc. Canada.

### Internet sites:

<http://europa.eu.int/>

The European Union's server. The Parliament, the Council, the Commission, the Court of Justice, the Court of Auditors and other bodies of the European Union (EU).

[www.abe.org](http://www.abe.org)

The Euro Banking Association (EBA).

[www.bis.org](http://www.bis.org)

The Bank for International Settlements.

[www.bof.fi](http://www.bof.fi)

The Bank of Finland.

[www.chips.org](http://www.chips.org)

The Clearing House Interbank Payment System (CHIPS).

[www.ecb.int](http://www.ecb.int)

The European Central Bank.

[www.ecbs.org](http://www.ecbs.org)

The European Committee for Banking Standards (ECBS).

[www.ofx.net](http://www.ofx.net)

Open Financial Exchange.

[www.pankkiyhdistys.fi](http://www.pankkiyhdistys.fi)  
The Finnish Banking association.

[www.swift.com](http://www.swift.com)  
S.W.I.F.T.

## **BANK OF FINLAND DISCUSSION PAPERS**

ISSN 0785-3572, print; ISSN 1456-6184, online

- 1/2002 Tuomas Välimäki **Bidding in fixed rate tenders: theory and experience with the ECB tenders.** 2002. 45 p. ISBN 951-686-763-4, print; ISBN 951-686-764-2, online. (TU)
- 2/2002 Juha Juntila **Forecasting the macroeconomy with current financial market information: Europe and the United States.** 2002. 70 p. ISBN 951-686-765-0, print; ISBN 951-686-766-9, online. (TU)
- 3/2002 Seppo Honkapohja – Kaushik Mitra **Performance of monetary policy with internal central bank forecasting.** 2002. 44 p. ISBN 951-686-767-7, print; ISBN 951-686-768-5, online. (TU)
- 4/2002 Iftekhar Hasan – Markku Malkamäki – Heiko Schmiedel **Technology, automation, and productivity of stock exchanges: International evidence.** 2002. 41 p. ISBN 951-686-769-3, print; ISBN 951-686-770-7, online. (TU)
- 5/2002 Vesa Kannianen – Mikko Leppämäki **Financial institutions and the allocation of talent.** 2002. 23 p. ISBN 951-686-771-5, print; ISBN 951-686-772-3, online. (TU)
- 6/2002 Anne Brunila – Marco Buti – Jan in't Veld **Cyclical stabilisation under the Stability and Growth Pact: How effective are automatic stabilisers?** 2002. 33 p. ISBN 951-686-773-1 print; ISBN 951-686-774-X, online. (KT)
- 7/2002 Anne Brunila **Fiscal policy: Coordination, discipline and stabilisation.** 2002. 24 p. ISBN 951-686-775-8 print; ISBN 951-686-776-6, online. (KT)
- 8/2002 Ari Hyytinen – Iikka Kuosa – Tuomas Takalo **Law or finance: Evidence from Finland.** 2002. 57 p. ISBN 951-686-777-4, print; ISBN 951-686-778-2, online. (TU)
- 9/2002 Bill F. Francis – Iftekhar Hasan – Delroy M. Hunter **Returns and volatility linkages in the international equity and currency markets.** 2002. 36 p. ISBN 951-686-779-0, print; ISBN 951-686-780-4, online. (TU)
- 10/2002 Heli Paunonen – Hanna Jyrkönen **Cash usage in Finland – How much can be explained?** 2002. 33 p. ISBN 951-686-781-2, print; ISBN 951-686-782-0, online. (RM)

- 11/2002 Heiko Schmiedel **Total factor productivity growth in European stock exchanges: A non-parametric frontier approach.** 2002. 37 p. ISBN 951-686-783-9, print; ISBN 951-686-784-7, online. (TU)
- 12/2002 Leena Mörntinen **Banking sector output and labour productivity in six European countries.** 2002. 73 p. ISBN 951-686-785-5, print; ISBN 951-686-786-3, online. (TU)
- 13/2002 Pertti Pylkkönen **Riskirahastot.** 2002. 44 p. ISBN 951-686-787-1, print; ISBN 951-686-788-X, online. (RM)
- 14/2002 Tapio Pohjola **Effects of fiscal policy on the durability of low inflation regimes.** 2002. 30 p. ISBN 951-686-789-8, print; ISBN 951-686-790-1, online. (TU)
- 15/2002 Iftekhar Hasan – Sudipto Sarkar **Banks' option to lend, interest rate sensitivity, and credit availability.** 2002. 54 p. ISBN 951-686-791-X, print; ISBN 951-686-792-8, online. (TU)
- 16/2002 Matti Keloharju – Markku Malkamäki – Kjell G. Nyborg – Kristian Rydqvist **A descriptive analysis of the Finnish treasury bond market.** 2002. 26 p. ISBN 951-686-793-6, print; ISBN 951-686-794-4, online. (TU)
- 17/2002 Mikko Niskanen **Lender of last resort and the moral hazard problem.** 2002. 32 p. ISBN 951-686-795-2, print; ISBN 951-686-796-0, online. (TU)
- 18/2002 George W. Evans – Seppo Honkapohja **Policy interaction, learning and the fiscal theory of prices.** 2002. 28 p. ISBN 951-686-797-9, print; ISBN 951-686-798-7, online. (TU)
- 19/2002 Kenneth Leong **Reconciling the New Keynesian model with observed persistence.** 2002. 29 p. ISBN 951-686-799-5, print; ISBN 952-462-000-6, online. (TU)
- 20/2002 Maritta Paloviita **Inflation dynamics in the euro area and the role of expectations.** 2002. 39 p. ISBN 952-462-003-0, print; ISBN 952-462-004-9, online. (TU)
- 21/2002 Markku Lanne **Nonlinear dynamics of interest rate and inflation.** 2002. 27 p. ISBN 952-462-005-7, print; ISBN 952-462-006-5, online. (TU)

22/2002 Jouko Vilmunen **Dynamics of investment behaviour in Finland: aggregate and firm level evidence.** 2002. 30 p. 952-462-007-3, print; ISBN 952-462-008-1, online. (TU)

23/2002 Harry Leinonen **Settlement in modern network-based payment infrastructures – description and prototype of the E-Settlement model.** 2002. 168 p. 952-462-009-X, print; ISBN 952-462-010-3, online. (RM)