Master seminar

A generic database plugin for the c'man synchronization platform

Michael Kanonik

Supervisor: Dr. Ralf Schenkel

in cooperation with Consistec GmbH

31 October 2008

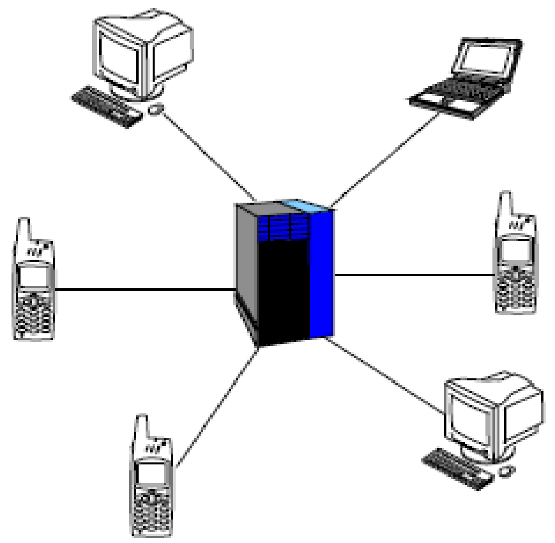
Outline

- Introduction
- SyncML Protocol
- Generic database plugin
- Conclusion

Introduction

•Data Synchronisation
Process of establishing consistency among data on remote sources and harmonization of the data over time

•Reason for standard
Existing synchronization solutions: vendor-, application- or operating system **specific**



Introduction

•SyncML: open platform-independent standard for data synchronization and device management.

•SyncML Initiative - a non-profit corporation formed by a group of major IT companies

•c`man (consistec mobile access node)
synchronisation platform based on SyncML

SyncML Protocol

Synchronisation scenario (types) in **c'man:**

1)Two-way sync Client and the Server exchange information about modified data in these devices

2)One-way sync from Server only

Only the Client gets all modifications from the Server

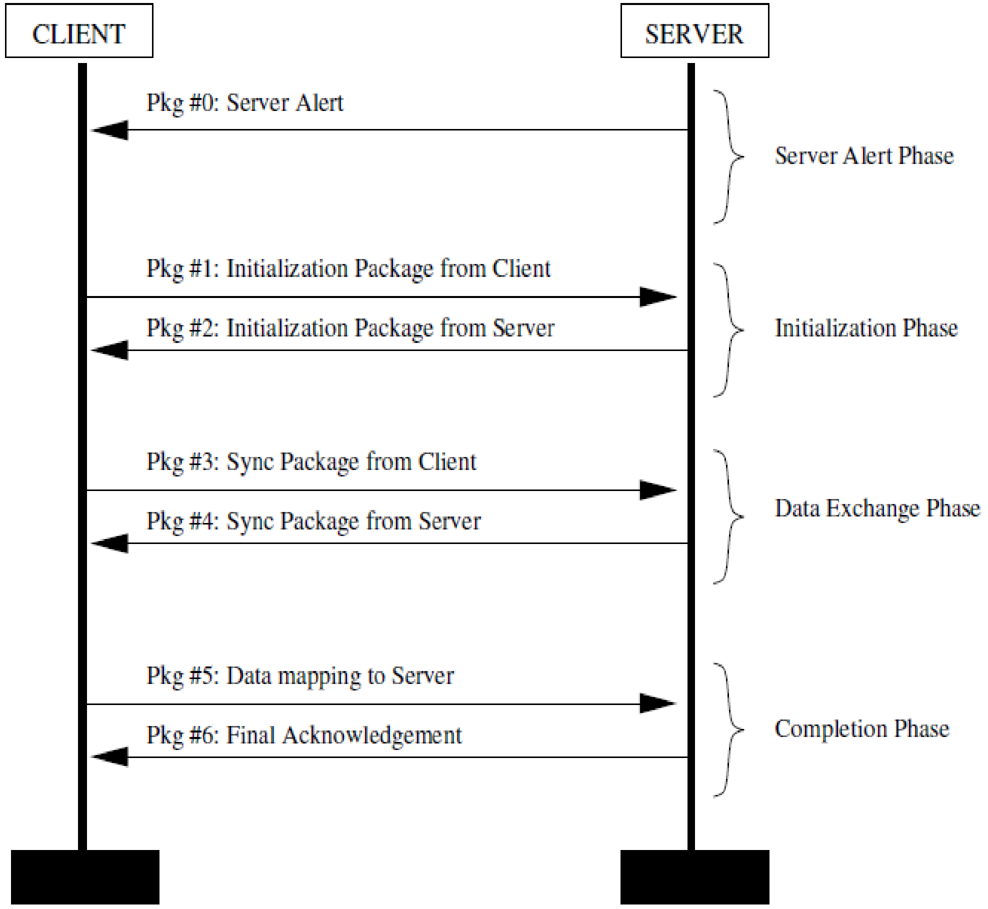
SyncML have additionally:

3) Slow sync

All items are compared with each other

SyncML Protocol - Phases

- Initialization
- Data Exchange
  Interaction for transferring all modifications since a previous synchronization
- Completion
  End a session properly
  Confirms that the synchronizing entities have received all the information


CLIENT
SERVER
Pkg #0: Server Alert
Server Alert Phase
Pkg #1: Initialization Package from Client
Pkg #2: Initialization Package from Server
Initialization Phase
Pkg #3: Sync Package from Client
Pkg #4: Sync Package from Server
Data Exchange Phase
Pkg #5: Data mapping to Server
Pkg #6: Final Acknowledgement
Completion Phase

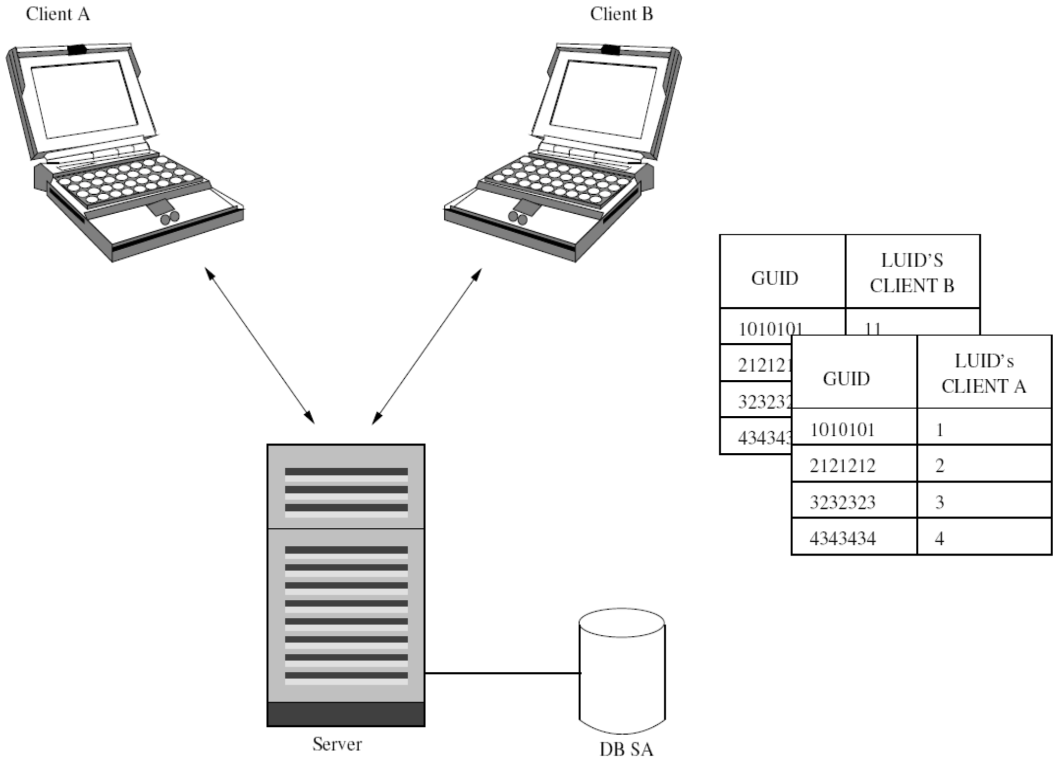
SyncML Protocol - Identifiers Mapping

Client devices:smaller capabilities

IDs to address data items:

• **Server**: globally unique identifiers GUID
(lengths - typically in the range of 64–128 bytes)

• **Clients**: local unique identifier LUID (lengths <=16 bytes)



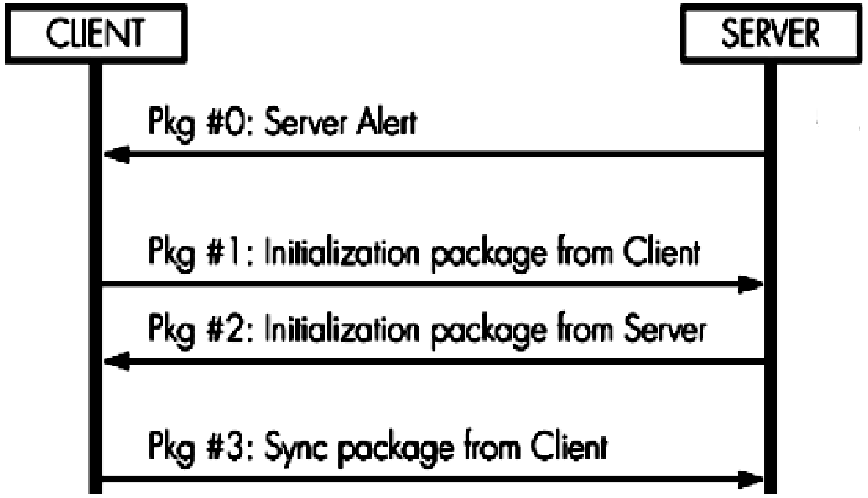
GUID	LUID'S CLIENT B
1010101	11
212121	
323232	
434343	

GUID	LUID's CLIENT A
1010101	1
2121212	2
3232323	3
4343434	4

SyncML Protocol - ID Mapping

LUID	DS obj

Client A


CLIENT
SERVER
Pkg #0: Server Alert
Pkg #1: Initialization package from Client
Pkg #2: Initialization package from Server
Pkg #3: Sync package from Client

GUID	DS obj
10001	d o 1
10002	d o 2

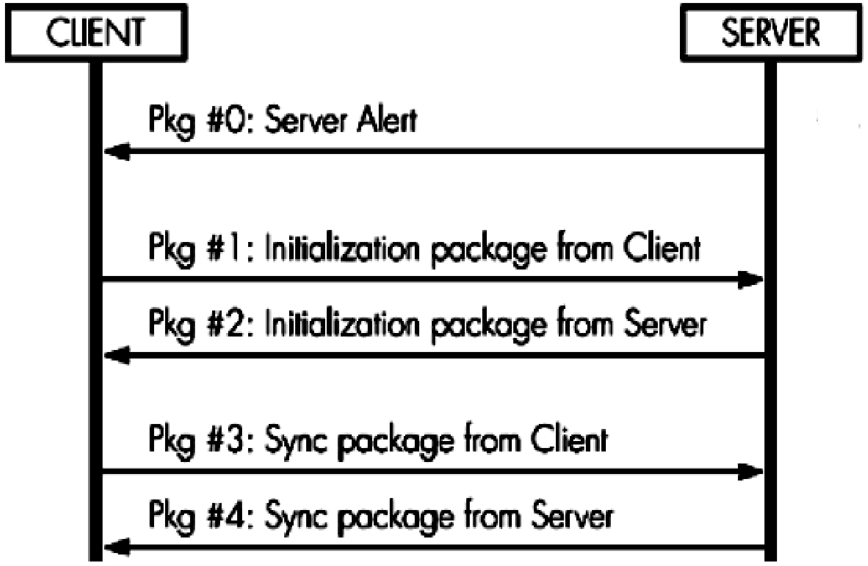
Server

GUID	LUID of A

SyncML Protocol - ID Mapping

LUID	DS obj
1	d o 1
2	d o 2

Client A


CLIENT
SERVER
Pkg #0: Server Alert
Pkg #1: Initialization package from Client
Pkg #2: Initialization package from Server
Pkg #3: Sync package from Client
Pkg #4: Sync package from Server

Add do1,do2

GUID	DS obj
10001	d o 1
10002	d o 2

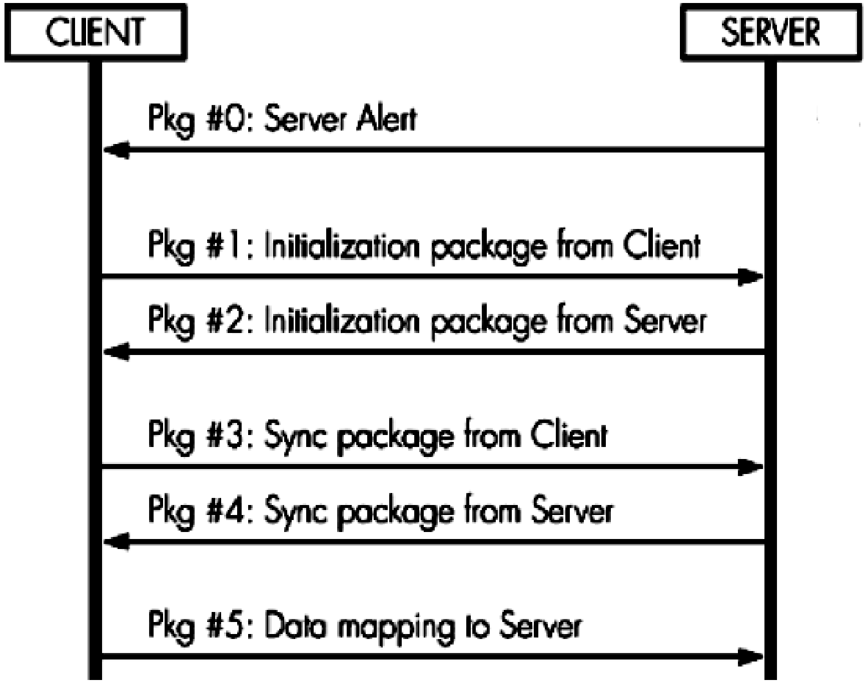
Server

GUID	LUID of A

SyncML Protocol - ID Mapping

LUID	DS obj
1	d o 1
2	d o 2

Client A


CLIENT
SERVER
Pkg #0: Server Alert
Pkg #1: Initialization package from Client
Pkg #2: Initialization package from Server
Pkg #3: Sync package from Client
Pkg #4: Sync package from Server
Pkg #5: Data mapping to Server

Status Add:OK

MAP(LUID,GUID)

GUID	DS obj
10001	d o 1
10002	d o 2

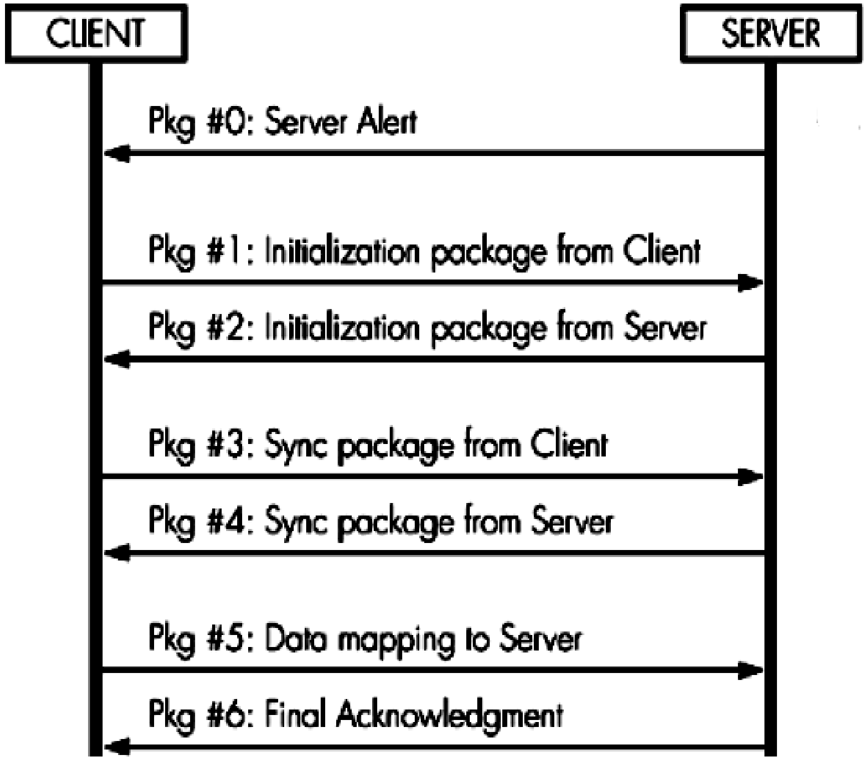
Server

GUID	LUID of A
10001	1
10002	2

SyncML Protocol - ID Mapping

LUID	DS obj
1	d o 1
2	d o 2

Client A


CLIENT
SERVER
Pkg #0: Server Alert
Pkg #1: Initialization package from Client
Pkg #2: Initialization package from Server
Pkg #3: Sync package from Client
Pkg #4: Sync package from Server
Pkg #5: Data mapping to Server
Pkg #6: Final Acknowledgment

GUID	DS obj
10001	d o 1
10002	d o 2

Server

GUID	LUID of A
10001	1
10002	2

Status MAP:OK

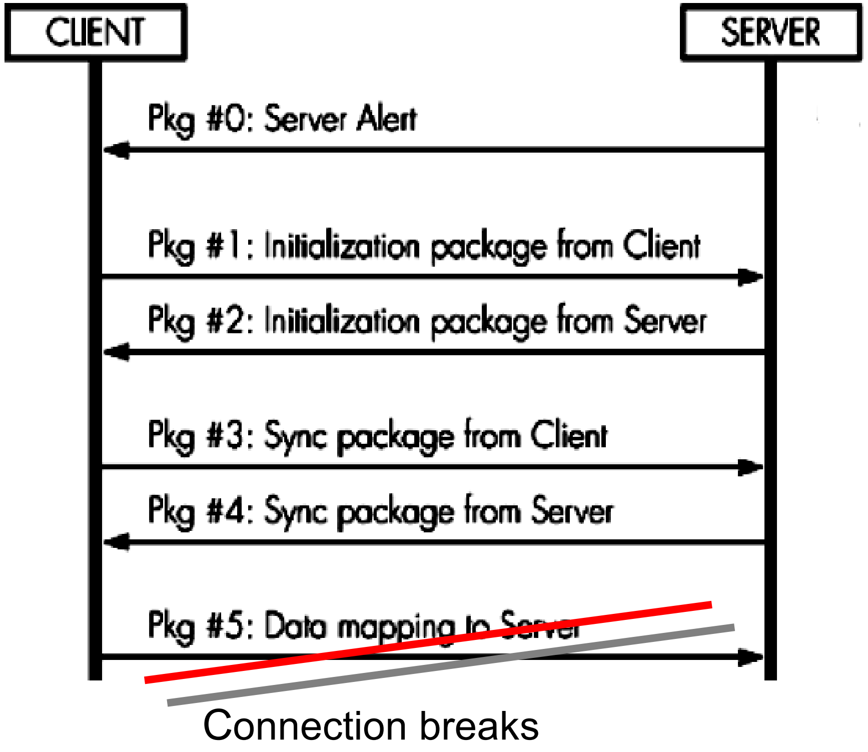
Goal of the thesis

- Extent of c'man synchronisation framework
- Compensate weakness of the SyncML protocol (inconsistent of DB)
- Implement configurable generic DB plugin

SyncML Protocol - ID Mapping Inconsistent

LUID	DS obj
1	d o 1
2	d o 2

Client A


CLIENT
SERVER
Pkg #0: Server Alert
Pkg #1: Initialization package from Client
Pkg #2: Initialization package from Server
Pkg #3: Sync package from Client
Pkg #4: Sync package from Server
Pkg #5: Data mapping to Server
Connection breaks

GUID	DS obj
10001	d o 1
10002	d o 2

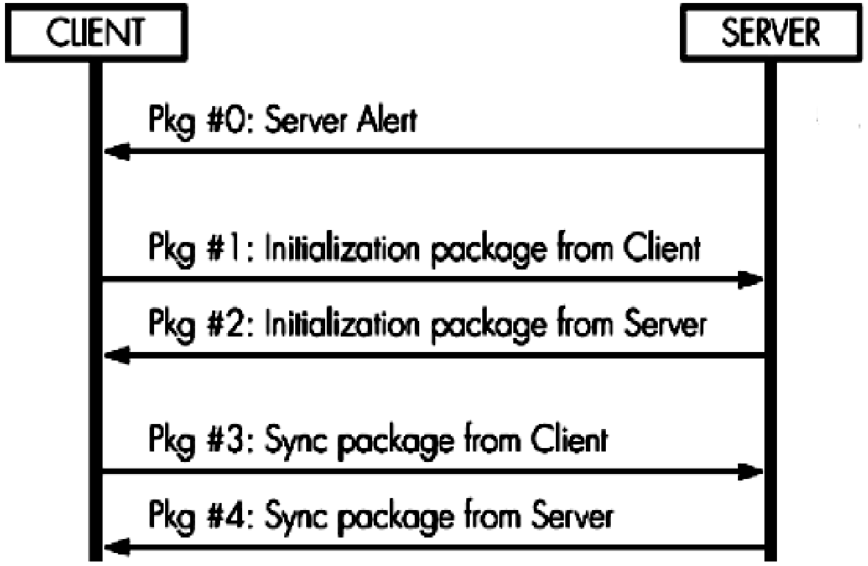
Server

GUID	LUID of A
~~10001~~	~~1~~
~~10002~~	~~2~~

SyncML Protocol - ID Mapping Inconsistent

LUID	DS obj
1	d o 1
2	d o 2
3	d o 1
4	d o 2

Client A


CLIENT
SERVER
Pkg #0: Server Alert
Pkg #1: Initialization package from Client
Pkg #2: Initialization package from Server
Pkg #3: Sync package from Client
Pkg #4: Sync package from Server

GUID	DS obj
10001	d o 1
10002	d o 2

Server

GUID	LUID of A

By next session

Client assigns new LUIDs

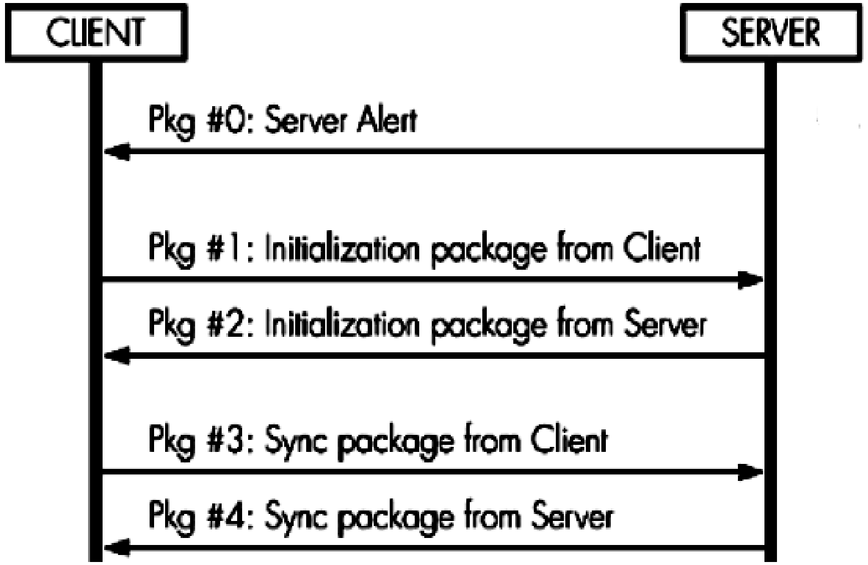
Add do1,do2

<u>Problem:</u> <u>duplicates</u> on the client

SyncML Protocol - ID Mapping Inconsistent

LUID	DS obj
1	d o 1
2	d o 2
3	d o 1
4	d o 2

Client A


CLIENT
SERVER
Pkg #0: Server Alert
Pkg #1: Initialization package from Client
Pkg #2: Initialization package from Server
Pkg #3: Sync package from Client
Pkg #4: Sync package from Server

GUID	DS obj
10001	d o 1
10002	d o 2

Server

GUID	LUID of A

Problem: duplicates on the client

Solution: 1)Roll-back when no confirmation to Server

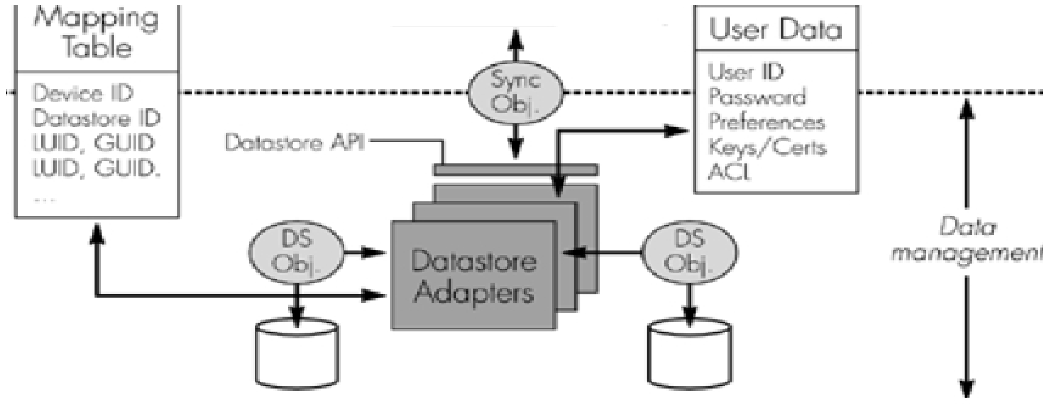
2)Use slow sync(is not implemented in c'man)

3) Drop Client DB and copy from Server

Outline

- Introduction
- SyncML Protocol
- **Generic database plugin**
- Conclusion

Generic Database plugin



<u>Data Management part</u>

accessing and updating the actual DSs that are being synchronized with the Client.

<u>Datastore (DS) Adapter:</u>

1) converts a generic <u>sync object</u> into <u>datastore object</u> and vice versa

2) assists in ID Mapping

Generic Database plugin - Adaptor

Now:

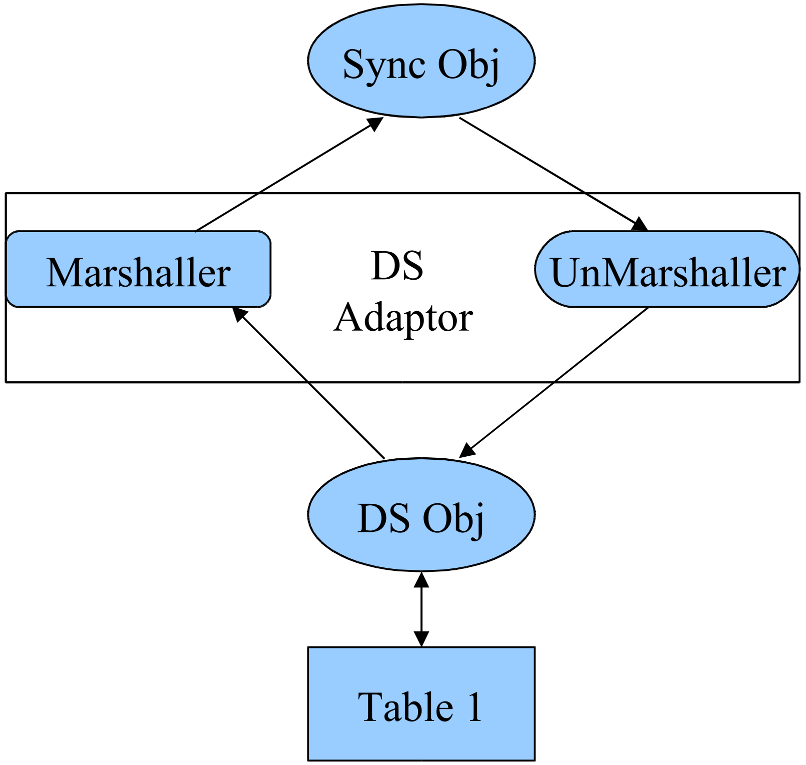
Server adaptors are implemented manually for every application/DS

Problem:

For every DB scheme - need to add/change code manually to DS API,Marshallers and UnMarshaller

•a lot of code

•errors occur often



Generic Database plugin - Adaptor

Main class of the API :

abstract class ADatastore -encapsulates Datastore

•**modifyContent**(AContentModification)

add ,replace or delete the content according to Modification type

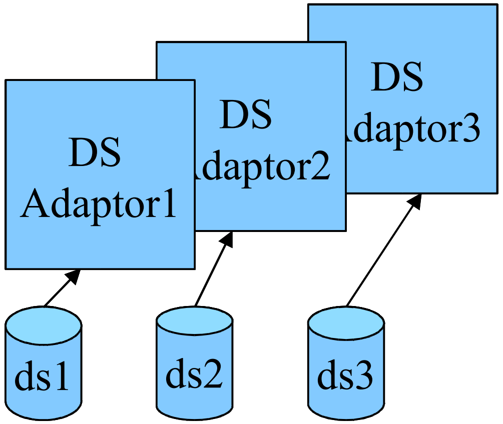
• AContentModification **getContentModification**(String UNID)
Returns REPLACE modification object if there is an entry with the specified UNID; DELETE - otherwise

• AcontentModification[ ] **getContentModificationsSince(Date date)**
Returns all Items,that was modified since certain date

Generic Database plugin - Adaptor

Goal:

To generate a scheme independent adaptor per Datastore type



Solution:

Represent tables of relational DB in XML format

Conclusion

Problems:

•Weakness of the SyncML protocol (inconsistent of DB)

•Application-specific adapter

Goals of the thesis:

•Extension of c'man synchronisation framework

•Improving SyncML protocol (secure transaction)

•Configurable through XML data server plugin

End

Thank You!

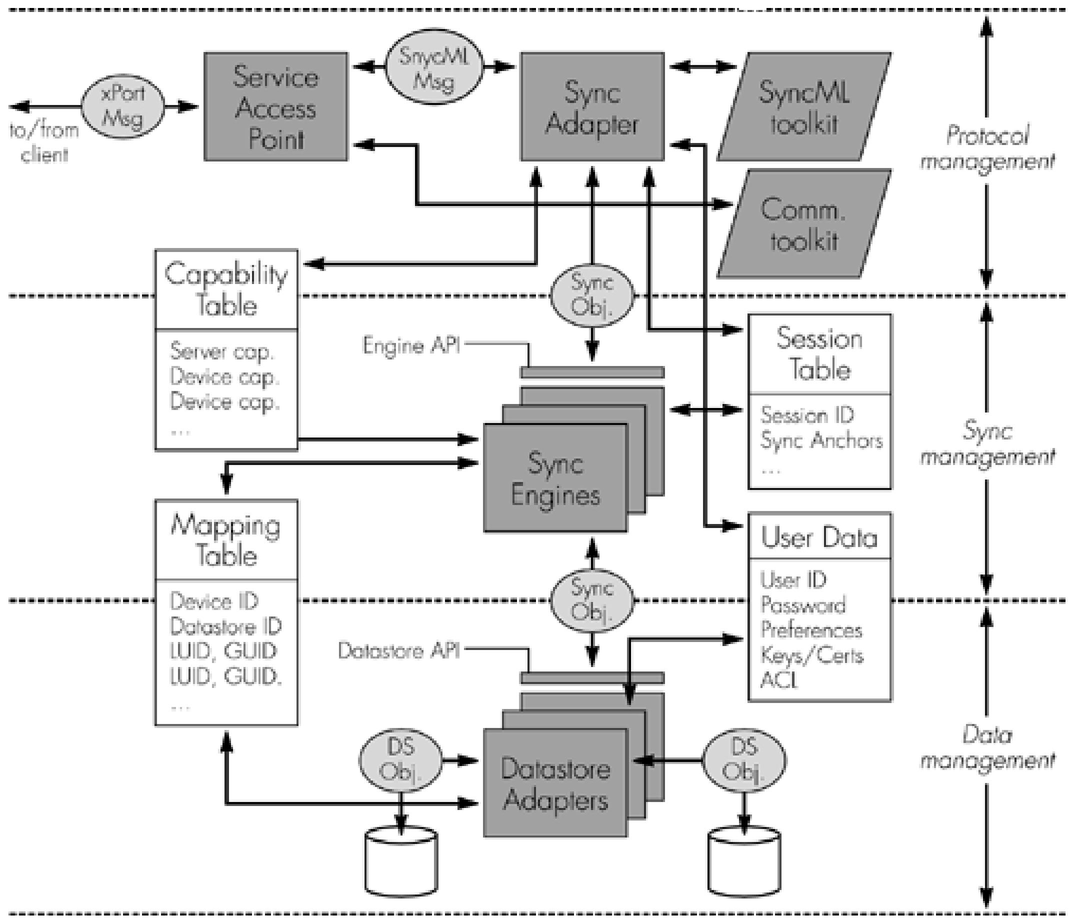
Questions, remarks?

Generic Database plugin - SyncML Server

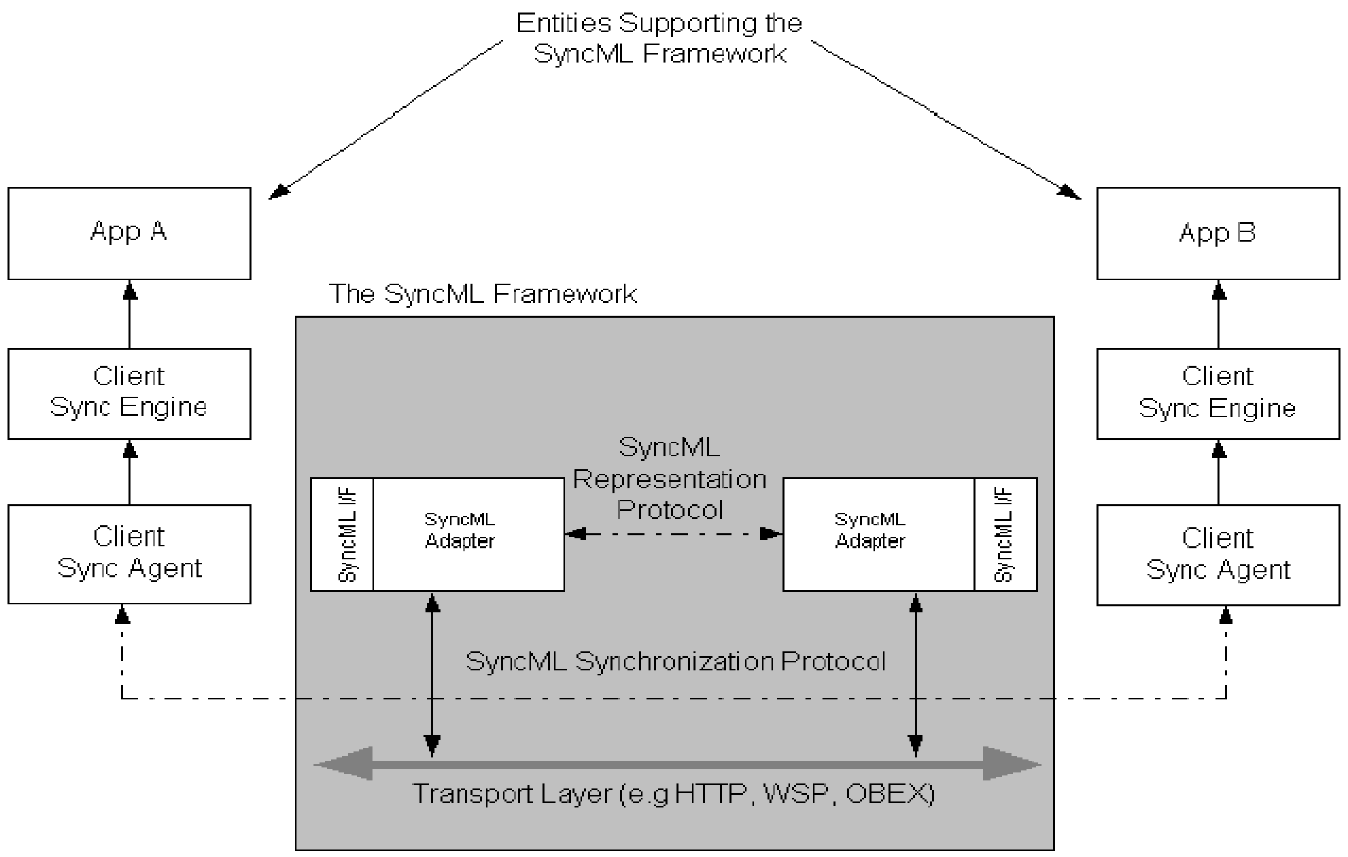
Generic SyncML Server implements

SyncML protocols:

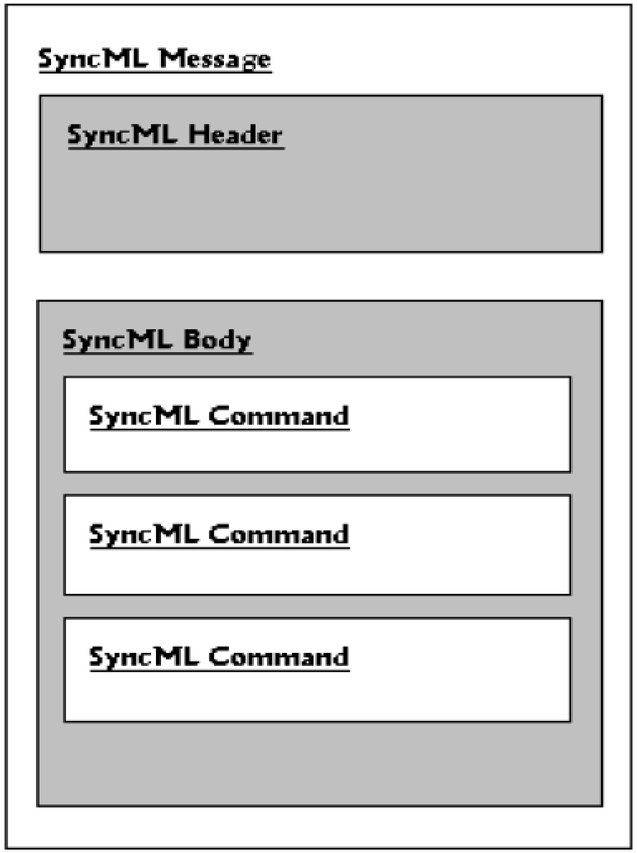
- Representation
- Synchronisation



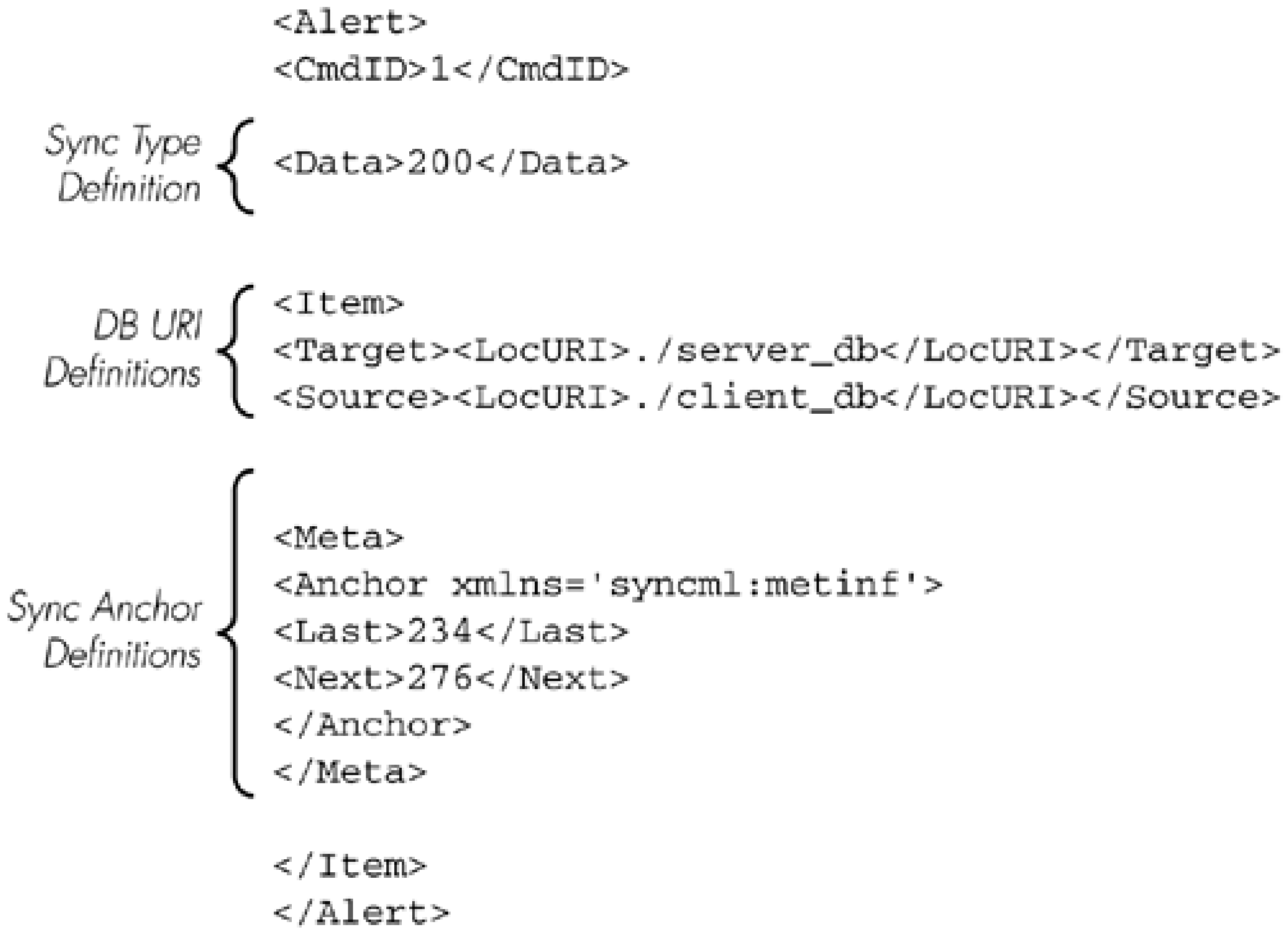
Appendix – SyncML Protocol



Appendix – SyncML Message

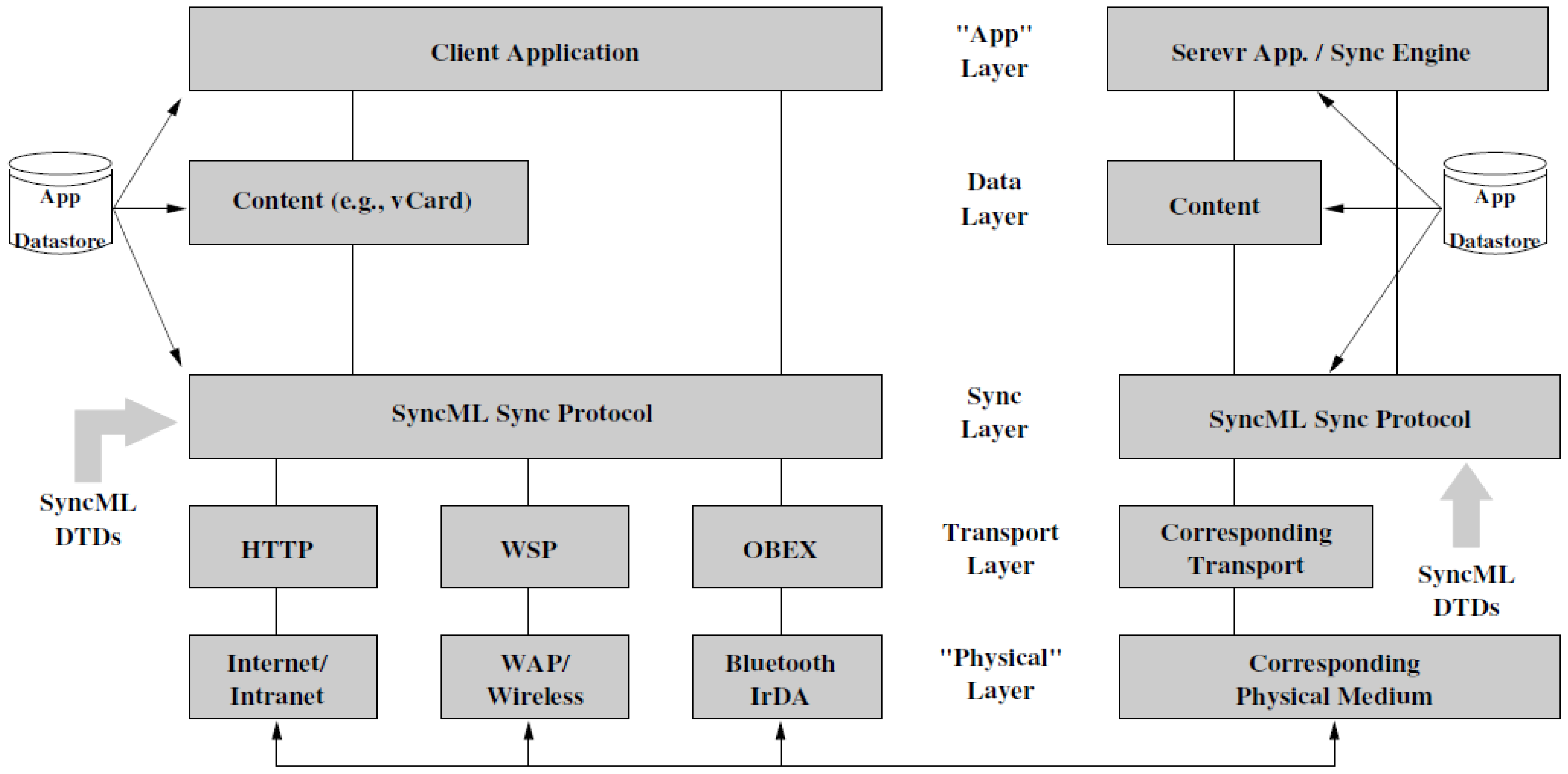


Appendix – Alert command in Initialization


Sync Type Definition
DB URI Definitions
Sync Anchor Definitions

```
<Alert>
<CmdID>1</CmdID>

<Data>200</Data>

<Item>
<Target><LocURI>./server_db</LocURI></Target>
<Source><LocURI>./client_db</LocURI></Source>

<Meta>
<Anchor xmlns='syncml:metinf'>
<Last>234</Last>
<Next>276</Next>
</Anchor>
</Meta>

</Item>
</Alert>
```

Appendix – Levels of architecture


Client Application
"App" Layer
Serevr App. / Sync Engine
App Datastore
Content (e.g., vCard)
Data Layer
Content
App Datastore
SyncML Sync Protocol
Sync Layer
SyncML Sync Protocol
SyncML DTDs
HTTP
WSP
OBEX
Transport Layer
Corresponding Transport
SyncML DTDs
Internet/ Intranet
WAP/ Wireless
Bluetooth IrDA
"Physical" Layer
Corresponding Physical Medium