

Future management of mobile phones

DO VAN THANH, ANNE MARTE HJEMÅS, ANDERS BJURGÅRD AND ØYSTEIN CHRISTIAN LØKEN



Do van Thanh is Senior Research Scientist at Telenor R&D



Anne Marte Hjemås is Research Scientist at Telenor R&D



Anders Bjurgård is Head of Business Logic & Terminals in Telenor Nordic



Øystein Chr. Løken is Product Manager for Device Management in Telenor Nordic

The increasing diversity and complexity of mobile phones call for an efficient Device Management System. Unfortunately, although there are currently quite a lot of effort and activities around device management, there is not yet a satisfactory one. This paper presents a vision of a future device management system elaborated by Telenor Nordic Mobile. It starts with an overview of the standards related to device management. Before describing the future Device Management System, the limitations of current systems are presented. Last but not least, the architecture of a future Device Management System is introduced and explained.

1 Introduction

Today's mobile phones are far more advanced than only a few years ago. The services that are available are not just plain old telephony anymore but rather complex and diverse. The user is faced with a broad range of devices and settings, and as the mobile phone converges to become a mini PC, the user will surely be overwhelmed by options, application settings, configurations, and services. He/she will be reluctant to install new services when there were problems with the first one. More advanced and open devices often mean more bugs, and an increasing risk of attacks by malicious viruses. However, the user is not the only player having problems. The mobile operator is also affected. Indeed, more problems also mean more pressure on the mobile operator's customer service. Due to the variety of devices and applications, the training of call centre agents could be a problem. On the other hand, the agents cannot expect the user to know what version of the operating system they have on their device. Some users may not even know what their device manufacturer or model is. Trying to figure this out before starting to solve the problem takes time and frustration. Many challenges are coming, and being able to guide the users through them will be an enormous added value.

The ideal solution to all the problems described above is to have a system that can perform remote and automatic configuration and management of the mobile phones with insignificant user's effort and minimal intervention from customer service. Such a system is called Device Management System in the mobile communication jargon. Unfortunately, although there are currently quite a lot of effort and activities around device management, there is not yet a satisfactory one. This paper presents a vision of a future device management system elaborated by Telenor Nordic Mobile [13]. It starts with an overview of the standards related to device management. Before describing the future Device Management System, the limitations of current systems are pre-

sented. Last but not least, the architecture of a future Device Management System is introduced and explained.

2 Standards for device management

There are many different protocols and technology for device management that support different functionality, and there will probably always be more than one protocol, but most standardization bodies and vendors look at SyncML DM as the most promising protocol that all other protocols merge into. OMA's SyncML DM supports all the functionality needed within 3-5 years, and is flexible for change. SyncML will be considered in more details in the following sections.

2.1 SyncML

"SyncML is a new industry initiative to develop and promote a single, common data synchronization protocol that can be used industry-wide. Driving the initiative are Ericsson, IBM, Lotus, Motorola, Nokia, Palm Inc., Psion and Starfish Software. The SyncML initiative is supported by several hundred leading companies in the industry, and additional companies are welcome to join the open initiative and participate." [1]. SyncML joined the Open Mobile Alliance (OMA) [2] in 2002 and the SyncML protocol became hence an OMA standard.

Figure 1 gives an overview of the SyncML Protocol Architecture. SyncML provides a common interface for synchronization that supports several different transport protocols and physical media. This will help achieve the goal that any mobile device should be able to synchronize with any data repository. The SyncML Protocol does not dictate the internal representation of data, but rather how to exchange data in an interoperable way. The protocol is designed to be extensible, allowing developers the freedom to define new data formats or protocol primitives.

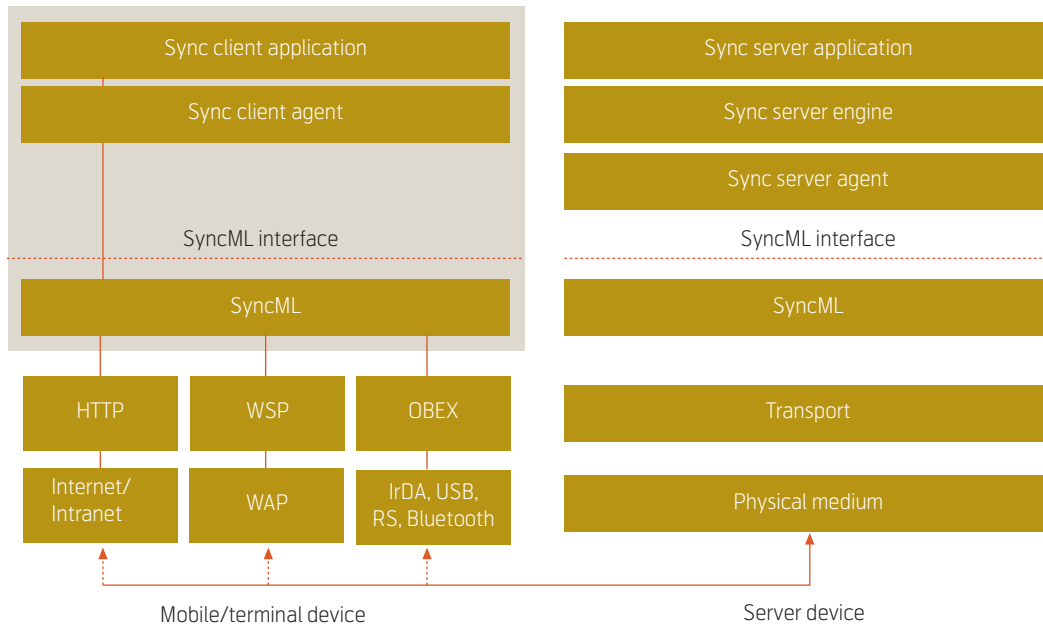


Figure 1 SyncML Protocol Architecture [2]

SyncML is based on XML and provides an XML DTD allowing representation of the information required for device management and data synchronization. This provides the means to represent commands, data and also metadata. The Device Management and Data Synchronization protocols describe how to exchange messages conforming to the DTD.

Being tailored for mobile devices with potentially low bandwidth, a binary encoding is defined. This encoding is based on work by the WAP Forum; WBXML (WAP Binary XML).

The SyncML specification contains the following main components:

- An XML-based representation protocol;
- A synchronization protocol (SyncML DS) and a device management protocol (SyncML DM);
- Transport bindings for the protocol;
- A device description framework for device management. [6]

2.2 SyncML DM

The Device Management Working Group continues the work done by the SyncML initiative and WAP Forum concerning Device Management (DM). Of most importance here is the SyncML Device Management Protocol.

The SyncML Device Management Protocol [3] is a management protocol that uses the SyncML Repre-

sentation Protocol. Uniquely addressable management objects are used to reflect the various aspects of device settings and run-time environment. Device settings may be read or written using commands that operate on the management objects.

Figure 2 gives an overview of the two phases of the SyncML DM Protocol – authentication and management. Authentication may be initiated by the user himself, or possibly by some form of notification sent from the server. The SyncML DM Protocol specifies several notification methods. Either way, the authen-

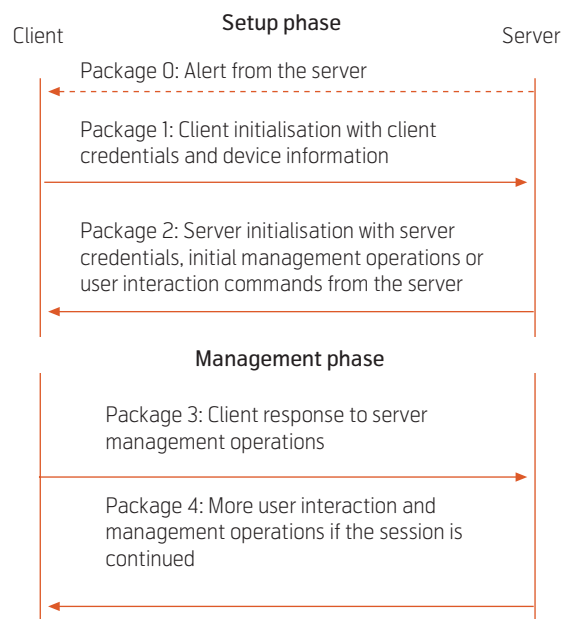


Figure 2 SyncML DM Protocol Parts [4]

tication phase ensures that the client and server can establish a trusted relationship. Also, the authentication reply, Package 2, from the server may contain management operations to be carried out by the client and may end the device management session if no further communication is needed. One or more iterations of the management phase then follow. [4]

SyncML Device Management Bootstrap is the initial process in which an un-provisioned device is provided with the necessary settings to start the ordinary device management process described above. SyncML DM Bootstrap is not intended to be used for other purposes.

Bootstrapping can be done either as a customized bootstrap or a server initiated bootstrap. Figure 3 gives an overview of customized bootstrap. The device is provided at the factory with the necessary information required to engage in a management session. This is not very flexible, and requires that the manufacturer knows details about the operator's network.

Figure 4 shows a server-initiated bootstrap. By way of automatic detection, the server initiates the management session.

SyncML DM promises to simplify device management by providing a common protocol across different devices. Of course, settings for each device will still be determined by its capabilities. Mobile phone manufacturers can expose any type of device settings, including runtime environment. This will provide the capability to upgrade software or firmware and provide bug fixes.

SyncML DM v1.1.2 became an approved enabler release in January 2004. An approved enabler release has passed the candidate status and has associated interoperability test cases generated by OMA.

An example from [6] provides a more hands-on feeling of how the protocol appears (Figure 5 and Figure 6).

The protocol contains necessary authentication support, alerts, error codes, etc. For those interested, [6] is a recommended document.

2.3 SyncML DS

The Data Synchronization Working Group continues the work done by the SyncML Initiative regarding Data Synchronization, DS. The SyncML Data Synchronization protocol specifies how the SyncML Representation Protocol is to be used for synchronization of data.

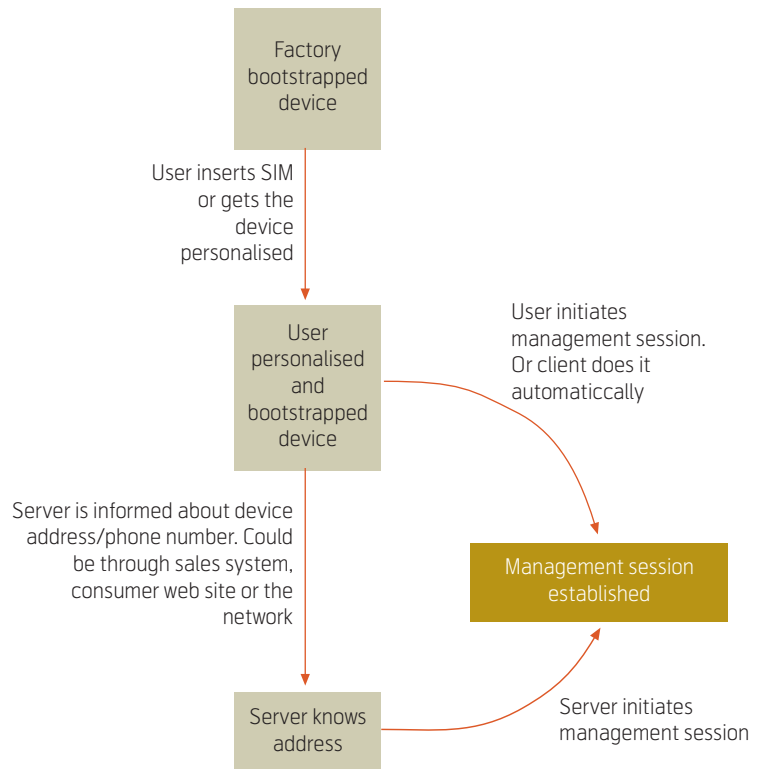


Figure 3 Customized bootstrap [5]

Synchronization of data between two devices requires that a change log be kept. This allows the server and client to figure out which changes to the data are most recent. SyncML DS does not pose any restric-

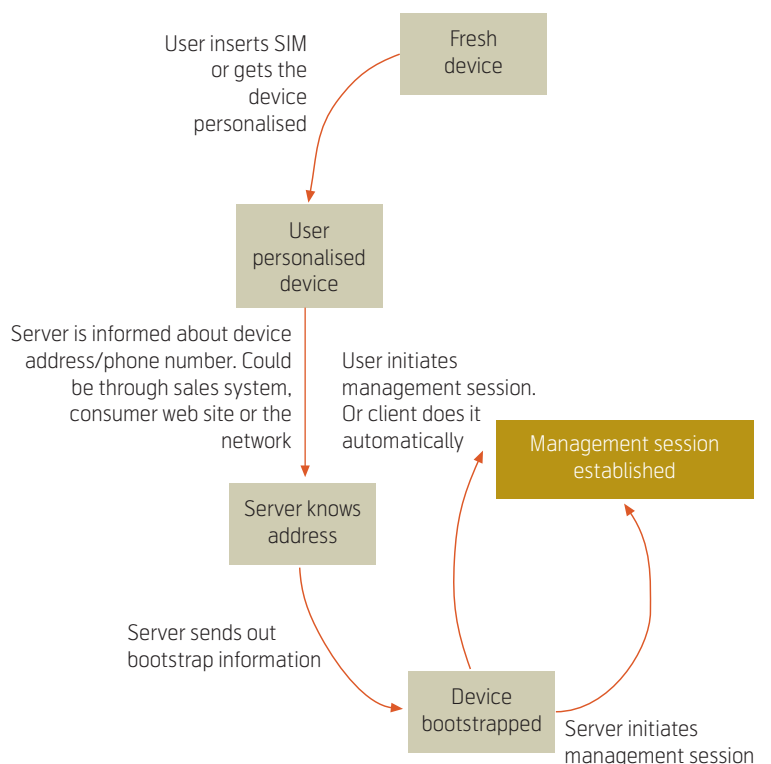


Figure 4 Server initiated bootstrap [5]

```

<SyncML xmlns='SYNML:SYNML1.1'>
  <SyncHdr>
    ...blah, blah...
  </SyncHdr>
  <SyncBody>
    <Status>
      <MsgRef>1</MsgRef>
      <CmdID>1</CmdID>
      <CmdRef>0</CmdRef>
      <Cmd>SyncHdr</Cmd>
      <Data>200</Data>
    </Status>
    <Alert>
      <CmdID>2</CmdID>
      <Data>1100</Data> <!-- User displayable notification -->
      <Item></Item>
      <Item>
        <Data>Your antivirus software is being updated.</Data>
      </Item>
    </Alert>
    <Get>
      <CmdID>3</CmdID>
      <Item>
        <Target>
          <LocURI>./antivirus_data/version</LocURI>
        </Target>
      </Item>
    </Get>
    <Final/>
  </SyncBody>
</SyncML>

```

Figure 5 An example of initiation of an anti-virus update

```

<Add>
  <CmdID>2</CmdID>
  <Meta>
    <Format xmlns="syncml:metinf">b64</Format>
    <Type xmlns="syncml:metinf">
      application/antivirus-inc.virusdef
    </Type>
  </Meta>
  <Item>
    <Meta>
      <Size xmlns='syncml:metinf'>37214</Size>
    </Meta>
    <Target><LocURI>./antivirus data</LocURI></Target>
    <Data>
      <!--Base64-coded antivirus file -->
    </Data>
  </Item>
</Add>

```

Figure 6 An example of adding an anti-virus software update

tions on how this change log is implemented, as long as the devices are able to specify which data items have changed. Figure 7 shows the basics of data synchronization. The device, in this case a mobile phone, will send a record of all changes since the previous synchronization took place. The server will then check this record against its own data and decide which data to keep, change or delete. The server will

then send a reply to the client with its own set of modified data. [7][8]

The advantage of SyncML DS is that it allows synchronization of any data that the device manufacturer decides to expose, using a common framework. Also the most important mobile phone manufacturers are supporting and implementing this protocol.

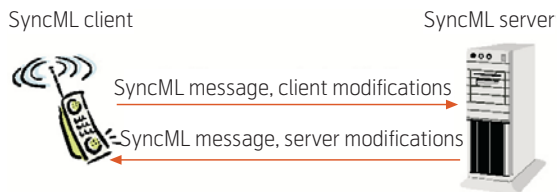


Figure 7 Synchronizing mobile client with server [8]

3 Current Device Management Systems

Current Device Management Systems are still rather primitive and limited. Some operators have a deal with some distributors to pre-configure devices with MMS, WAP and Internet. The distributors get a provision for a successful configuration. The end users can obtain the settings themselves via WEB, SMS, WAP or IVR solutions. In most cases the users need to specify their phone type and also their phone number in some cases. The SMS interface is used a lot, and the Web interface has even higher and more increasing usage than the SMS interface. They can also call customer service, but the process is no different except that the call centre agent performs the same operation. In some cases the agents have to ask the customers what manufacturer and device model they have. They will then receive an Over-The-Air (OTA) message that sets up the device. For technical reasons, they have to do this for each setting: MMS, WAP and Internet.

The OTA management functions offered today are quite often limited setups of CSD, GPRS, MMS, ISP, Bookmarks and email settings, and SyncML DS.

OTA can be requested in the following ways:

- SMS sent by customer
- Mobile operator WEB-sites: for example telenormobil.no, djuice.no, djuice.se, telenormobile.se
- Mobile operator WAP-site: for example wap.telenormobil.no

4 Future Device Management System

4.1 Players of the Future Device Management System

In addition to the customer who is the end-user, a future Device Management System will interact with several other important players that have different needs and expectations. The identified players for the future DMS are as follows:

- *Customer/device user:* The customer is the service subscriber and device user. This should be one of the most important users to consider. A future DMS will satisfy many customer needs. The customer wants:
 - Services to perform flawlessly from day one with minimum interaction
 - Backup and synchronization of personal data
 - Device independent operation. The services follow the user, not the device
 - A faultless and safe device
 - Customized and personalized devices and services
- *Corporation:* A corporation is also a very important customer. A corporation will gain in addition to the customer:
 - Improved management of mobile devices
 - Bulk configuration
 - Presence information
 - Branding and corporation menus
- *Operator:* The operator provides the network and its resources. The operator can also host and administer the solution. Example: Telenor Nordic Mobile and NetCom. The operator wants:
 - One solution for all devices
 - Enhanced service level and several administration levels adapted to customer service, net division, DMS administrator and corporate users
 - Increased service uptake
 - Decreased customer care costs
- *Service Provider:* The Service Provider develops and provides value added services, and is responsible for the provision and continuous availability of subscribed services. Responsibilities are management, administration and operation of services. The service provider relies on the operator by means of the physical network and resources. The needs are much the same as those of the Operator.
- *Content Provider:* Also called third parties. The content provider in this case delivers services through the service provider. It could be Java games or third party applications. Content providers rely on the service provider for provisioning and billing. Benefits are:
 - A standardized tool for application configuration
 - Faster content delivery
 - Improved content adaptation
- *Device Manufacturer:* The manufacturer produces the actual devices and has knowledge of terminal capabilities, known bugs, updates, etc. The manufacturers have many benefits of a standardized Device Management.

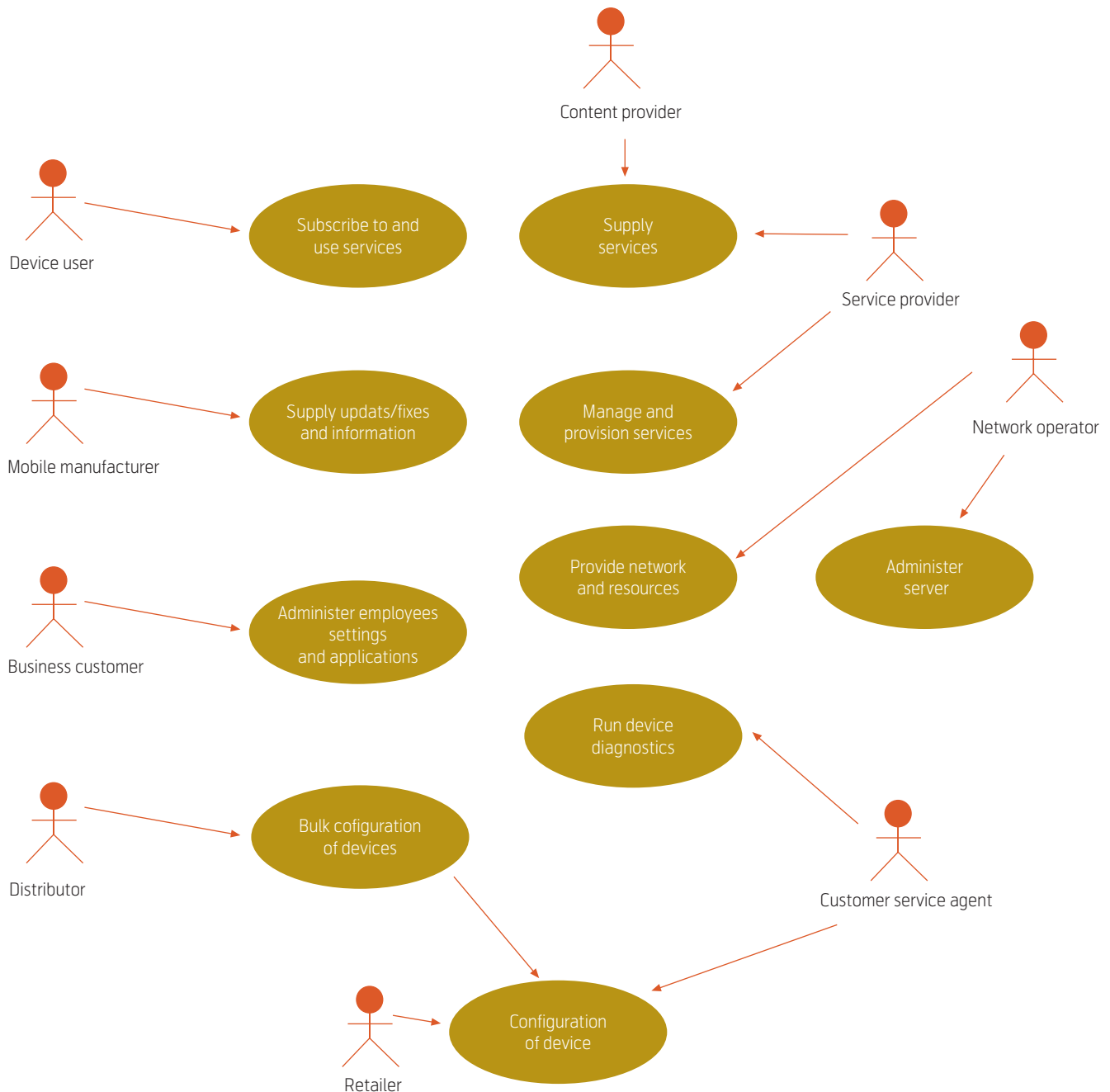


Figure 8 Players and business use cases

- Only one solution to support in the devices
- No need to call back devices with firm/software errors, can perform remote updates
- **Customer Service Agent/Operator's Customer Service:** A customer Service Agent talks to the device users, provides information and helps them solve their problems. Benefits are:
 - Instant knowledge of device characteristics
 - Quicker service
 - Better tools to aid customer
- **Retailer:** The retailer sells the devices to the end user. They can pre-configure the phone for the user and perform repairs etc. Benefits:
 - Easier configuration
- **Distributor:** The distributor can have a deal with a Network Operator to pre-configure a volume of devices. Benefits:
 - Bulk configuration

All the players need a user interface to the system. The different players have different access. Some players have access to similar functionality, but access to a subset of information. We prioritize cus-

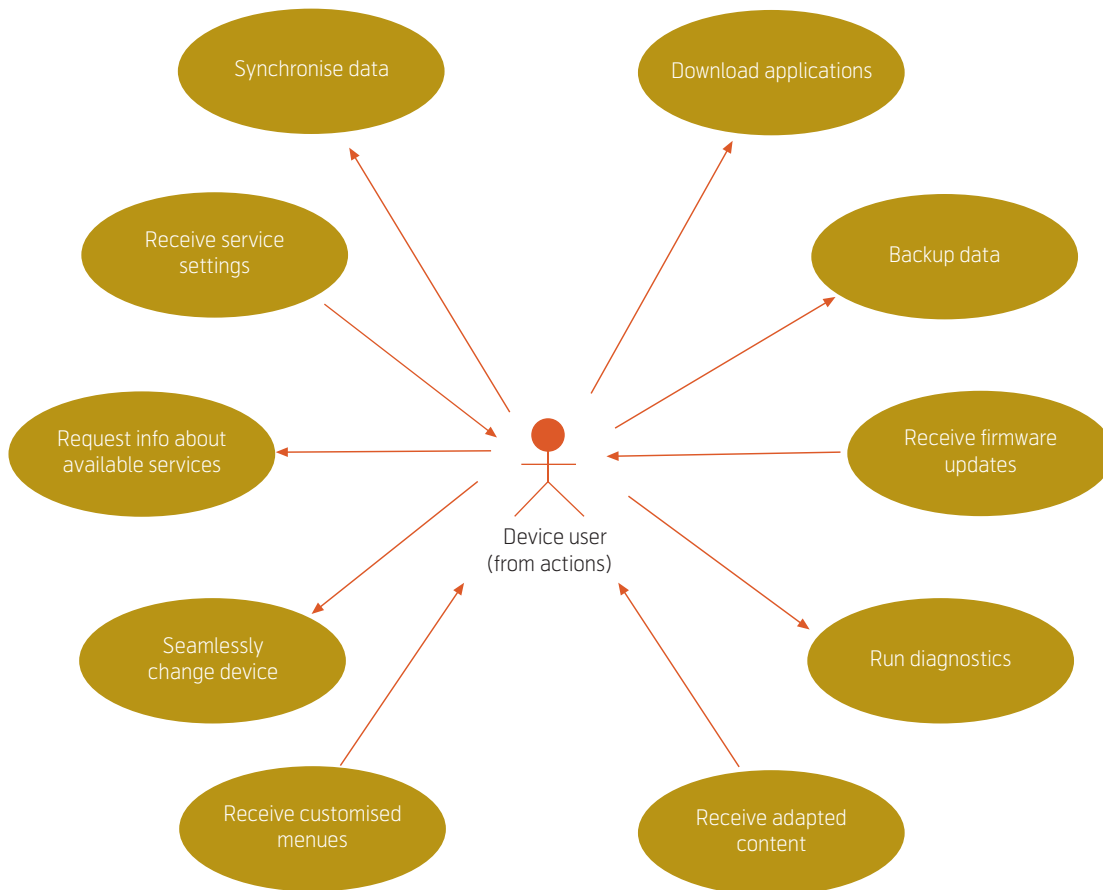


Figure 9 Use cases for device user

tomers angle, but the other players are involved in many of the use cases.

Figure 9 shows some use cases for the device user. “Seamlessly change device” means that the data and services follow the user, not the device.

4.2 Features of the Device Management System

There are four features that we will probably see in most Device Management Systems and devices within three years. The features present in the Device Management System are:

- Remote Configuration
- Software and Firmware updates
- Mobile Diagnostics
- User Data Management
- Personalisation

4.2.1 Remote Configuration

Remote Configuration includes provisioning of key parameters to configure the handset and activate the service remotely, setting initial configuration information in devices, and modifying or reading operator parameters. The interaction with the user can be fully automatic, with user input, or through a session. How

automatic the configuration is, should be based on the nature of the data. If it is configuration for basic services like MMS, it should be automatic. For services that have personal configuration parameters, there may be a need for some kind of user interaction.

Bulk configuration must also be enabled and is a useful operation that can customize groups of devices, for example updating the electronic payment software package on the SIM card of all prepaid customers. An administrator of the Device Management system can manage branded services, menus and other UI customization remotely. If the service requires certain software or firmware updates, we rely on the functionality in the next section. By providing automatic download of service menu for specific services to the phone, the usage and user experience of the services will most likely increase. Users with advanced WPABX solutions can then easily use functionality like call transfer, add/remove themselves from hunting groups etc. without having to remember cryptic key codes.

4.2.2 Software and firmware updates

Device management can also perform software operations, i.e. upgrading software packages or installing new applications on the terminal to provide for new

services. This includes both native and Java applications. Downloading software will be more attractive when the future high-speed connections are available. There will also be remote software management where the functionality for the Administrator will be [12]:

- Application and component inventory
- Install, delete, update, check, stop, kill, start application and/or component
- Remote reboot.

All with the proper user consent and authorization, of course.

Software operations also include installation of software patches/bug fixes for operational improvement, i.e. firmware. Firmware is programming that is inserted into programmable read-only memory, thus becoming a permanent part of a computing device. Firmware is created and tested like software. When ready, it can be distributed like other software and, using a special user interface, installed in the programmable read-only memory by the user. Firmware is sometimes distributed for printers, modems, and other computer devices [9]. In short: computer programme instructions incorporated in a hardware device that cannot be changed under computer program control are defined as firmware (e.g. the operating system of a device).

Updating firmware can improve performance, fix bugs or add functionality. Many still believe that the mobile phone is secure from viruses and worms, but there are already security holes in some devices where worms can spread via Bluetooth, and SMS can be sent without the user noticing in Java applications [12]. Firmware updates are delivered as binary packages replacing the malfunctioning part of the software image. The mobile manufacturer delivers the firmware update packages to the operator, which distributes the updates to the relevant handsets. The updates include a security voucher from the mobile manufacturer ensuring secure handling.

4.2.3 Mobile diagnostics

It is important to be able to read as much information as possible from a device, e.g. service settings, device model, a list of installed or running software, version information, hardware configurations, etc. A service can then be adapted to device capabilities and user preferences. The system can also listen for alerts sent from a device, and invoke local diagnostics on it. The user will be notified of the diagnostic session and can give permanent or temporary rights for certain administration actions. The user can also revoke the rights. The possibility of automatic detection is also sorted under

this functionality. Automatic detection means that the network detects the (new) combination of device and SIM-card as soon as the terminal is turned on or comes within range. This feature is very important for many of the user scenarios that are discussed later.

4.2.4 User data management

User data management includes backup and synchronization of user data, system and service settings etc. Standards like SyncML DS, for synchronization of mobile terminals, has been in the works for many years, and is already included in the new, advanced devices. The user can now synchronize data anywhere, anytime. The backup could be either content aware or content unaware. No content awareness means that a raw, binary image is copied to a server. The data cannot be restored to another device model. Content aware backup means that the user can select what will be backed up and the data can be restored to another device model. Data like calendar, contacts etc. could be synchronized to a USIM card so that the possible bad consequences of borrowing a phone are reduced. The customer only needs access and the correct settings to a reliable server.

4.2.5 Personalization

If the laws and regulations allow it, the service provider could gather information about the user and use personalization to offer services that the user might be interested in. The service offer could be an extra item in the operator specific menu; either a concrete offer or a menu option that fetches new service offerings. Personalization uses information that is either obtained from or provided by the user. A good example is the online book store Amazon.com which now has expanded to selling DVDs, toys and much more. The site gathers information about the user based on what the user buys, views or searches for. Based on that information and what other users with similar preferences have bought, the user will be presented with offers that the personalization engine has predicted to be interesting. The recommendations, based on a complex set of rules, are often quite accurate.

This could also apply to telecom services. If Nick has ticked that he is interested in subject A, or has downloaded Application A, he could receive recommendations about similar applications. Or maybe many users that have downloaded Application A also have downloaded Application B. The system could then recommend Application B to Nick.

4.3 Architecture of a future Device Management System

The architecture of a future Device Management System is shown in Figure 10.

DMS server

The DMS server contains the actual business logic for Device Management. The DMS server's main responsibility is to keep track of existing terminals and their general attributes, OTA standards (proprietary and SyncML DM), handle requests and produce statistics. All non-OTA interfaces to external systems should be standardized and controlled/owned by the operator. Device Management is developing fast, and it is vital that it is possible to change vendors or at least keep them on their toes. When detecting a new IMEI-MSISDN combination the DMS should add or update an entry in the Subscriber-Terminal DB, perform a mobile diagnostic and register and validate settings etc. on the device. If any updates are necessary an update session will be initiated towards the device with or without user interaction.

There are also advantages of having the terminal database outside the DMS server. The operator has more control, but the vendor should have an interface to the database in order to keep it updated. This would probably also mean more server customization for the vendors if the operator wishes to use another server.

Having all device related information and operations in one place fits the CSF-philosophy. Using the DMS for all device operations fits the LEGO principle. The question is how many LEGO parts the system should be made of? The terminal database could be one separate self-serviced LEGO-part, but it is an integral part of the DMS and the data is not operator specific. Regardless, the other systems should only use the DMS interface to obtain device information without having to know whether the database is in the server or in a separate LEGO part used by the server.

Storage of IMEI in the HLR / automatic detection

An infrastructure in the network for detecting SIM-IMEI combinations must be present. Several vendors offer a solution inspired by Vodafone's 3GPP-proposal to store the IMEI in the HLR. This is a simple and elegant solution for simple applications, but increasing the subscriber profile size too much would decrease the number of subscribers to be served by an HLR. HLRs are expensive and difficult to update. They are system critical components, and changing their interfaces is a long process through standardization forums. However, storing only an IMEI number, or even just the TAC (Type Allocation Code) is suffi-

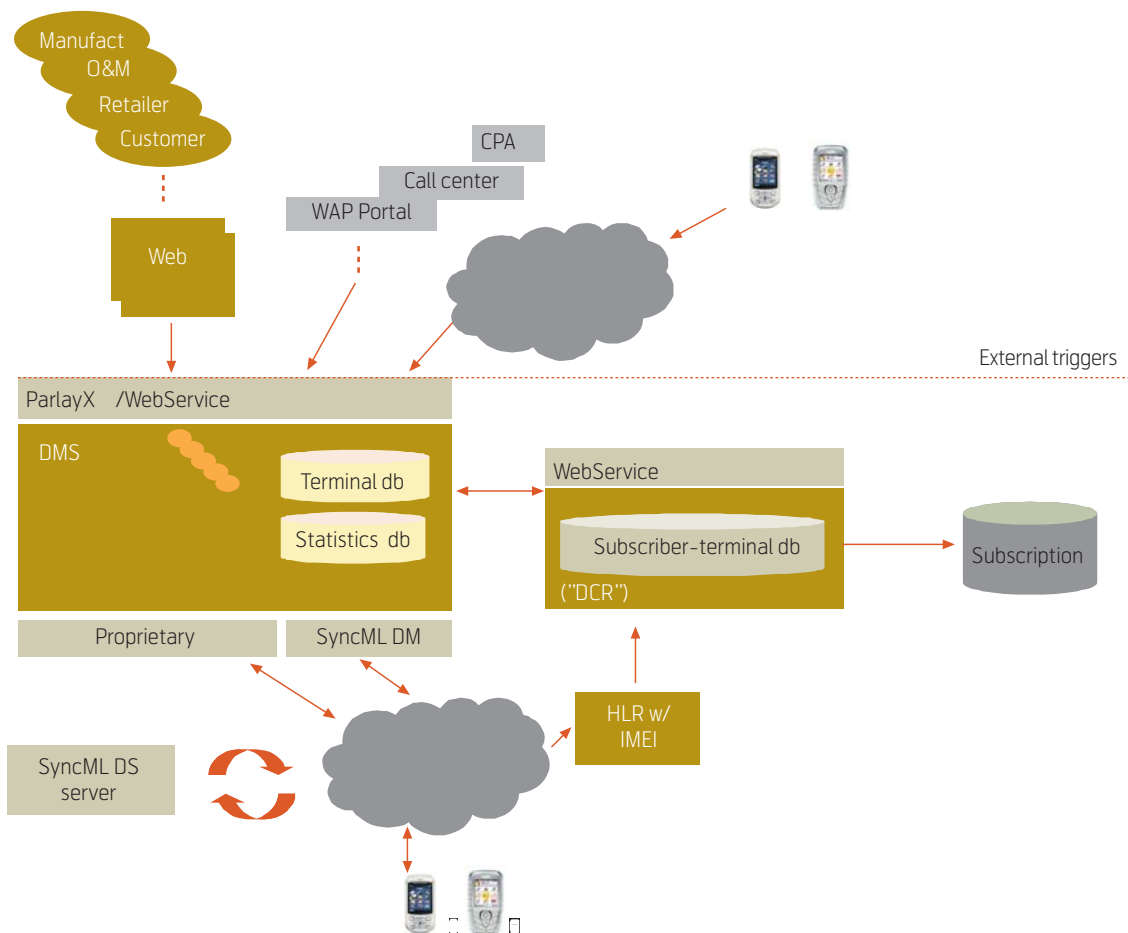


Figure 10 Architecture overview

cient for most of the basic usages like MMS or very performance critical applications. It is also necessary to detect new or unregistered combinations of IMEI-MSISDN, and the HLR is a good candidate. There must also be a trigger that notifies the DMS or the subscriber-terminal DB (depending on architecture) of new IMEI-MSISDN combinations.

Subscriber-terminal

A customer can have more than one subscription. We assume that the services follow the subscription, and hence the device data will be coupled with the phone number (MSISDN). The subscriber-terminal relation is perhaps the most difficult issue. Where should the relation be stored and what should be stored? A simple subscriber-IMEI relation might be present in the HLR within 2005, and the DMS already keeps a database of information about the terminals. An application could look up the subscriber's device type in the HLR, and then find more information in the terminal database. However, this is not extendible if in the future one should wish to extend the subscriber-terminal relationship with more information, e.g. what version of the virus software the customer has. This information could be requested direct from the device when needed, but the most flexible option would be to have a separate, extendible database to store detailed subscriber-device specific information. This is important for applications where the action is not triggered from the subscriber; for example, if one wants to automatically send virus software updates to all customers who have a specific device-operating system combination. It is very likely that the combination of device, firmware and software is not as obvious in the future as it is today. Today, device management is in most cases not as real-time critical as call handling, which also uses the HLR. Updating the HLR is an expensive task, and having a separate database for more high-level user data will provide for more flexibility and will be more future-proof. It is impossible to predict 100 % what type of information that is useful to store about a subscriber in 3–5 years. This database is most important for network-triggered actions and statistics. By separating this database from the DMS, the system will be more flexible and maintainable. The content of the database could be very Telenor specific, and integrating too much with the DMS would tie Telenor to one DMS vendor.

The subscriber entries could be populated either by a simple mirroring process, or by an event-based method that uses automatic detection to detect valid subscriptions in the network that do not already exist in the database. If it is a Nordic subscription, the DMS-db could be updated with the new subscription. With the event-based method there should be a clean-

up process making sure that devices that have not been registered in the network for n days, should be deleted. Regulations might even require such a process. The database should be handled by the DMS, but should be a separate, DMS-vendor-independent component. It could function as an extended HLR.

The subscriber-terminal data can be stored in a Lightweight Directory Access Protocol (LDAP) structure. LDAP defines how data should be accessed, but not how it is stored. Descriptive information is stored in the attributes of an entry, where each attribute describes a specific type of information. An example could be:

```
msisdn: 555 555 555
name: John Doe
imei: 347398473
os: Symbian v 7.0
```

Subscription database

Another difficult issue is how to integrate DMS with the subscription information. It is important to know if the subscriber is eligible for GPRS, which is a relevant problem at several operators. There is one of two statements that must be broken: No dependencies or no double storage. If we strive for minimum dependencies, we mirror the subscription database, or parts of it, to the Subscriber-Terminal Database (STdb). This would mean significantly more data in the STdb, also it will be more prone to error because of the delay between the mirror update. Event-based updates would solve this problem, but demands that the subscription database is designed to send events to listeners. Either way, duplicate data will be stored in two different databases. Another solution would be to have DMS or a process in the Subscriber-terminal to query the subscription database directly. Since the Subscriber-Terminal db probably will be more permanent and stable than the DMS, the queries could be handled from the db, thus the DMS would not have to know where the subscription data is. Today there is much stress on the subscription database, and this connection is not always stable. Today there is no integration with subscription data in the Norwegian DMS solution, but there is a clear need for it in the future. This integration must be taken into account when designing the subscriber-terminal database.

Web

Today the Web solution is tightly integrated with the DMS. The ideal would be to separate web from server so that the web solution uses the same interface as the other systems. The web interface will get more complex as more roles with more options get involved. Examples of this are web interfaces for web administrators in a company, operator administrators,

company employees and end-users. Skilled web developers is a must-have, and there are many qualified third parties that are cheaper than the DMS vendors.

J2EE

It is interesting to see that there is some J2EE-standardization work in progress regarding Device Management. Although this is too immature to be considered yet, one should be open to the possibility of great savings by integrating certain DMS functions into existing J2EE-infrastructure.

SyncML DS

Synchronization of PIM data is a very simple and cheap solution to put into operation. The only thing you need is a server that runs, e.g. JBoss with sync4J. However there are costs related to storage space, backup systems and other O&M tasks. We have put the SyncML DS server in the picture, because it should be readily available to Telenor's customers. If Telenor does not host it, there should be an agreement with a third party that Telenor offers a default set-up for.

5 Conclusion

The increasing complexity and diversity of the mobile phones are the fast growing challenges to mobile operators, and an efficient Device Management System is the best tool to solve it. The vision for a future Device Management System (DMS) is to lower the service threshold for the customers and make sure that the average user only has to think content, not technology. The most important functionalities like Remote Configuration and Mobile Diagnostics, including automatic detection, must be implemented.

Reference

- 1 *OMA Homepage*. (2004-04-15) [online] – URL: <http://www.openmobilealliance.com>
- 2 *Building an Industry-Wide Mobile Data Synchronization Protocol – SyncML White Paper, Version 1.0*. (2004-04-16) [online] – URL : <http://www.openmobilealliance.org/tech/affiliates/syncml/whitepaper.pdf>
- 3 *SyncML DM: A SyncML Protocol for Device Management*. (2004-04-16) [online] – URL: http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_devman_business_cases_whppr.pdf
- 4 *SyncML Device Management Protocol, Version 1.1*. (2004-04-16) [online] – URL: http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_dm_protocol_v11_20020215.pdf
- 5 *SyncML Device Management Bootstrap*. (2004-04-16) [online] – URL : http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_dm_boot_v11_20020215.pdf
- 6 *SyncML Representation Protocol, Device Management Usage, version 1.1*. (2004-04-16) [online] – URL: http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_dm_represent_v11_20020215.pdf
- 7 *SyncML Representation Protocol, Data Synchronization Usage, version 1.2, Draft Version 14-Jan-2004*. (2004-04-16) [online] – URL: http://member.openmobilealliance.org/ftp/Public_documents/DS/2003/OMA-DS-2003-0548R02-SyncRepresentDataSyncUsage.zip
- 8 *SyncML Sync Protocol, version 1.1*. (2004-04-16) [online] – URL: http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_sync_protocol_v11_20020215.pdf
- 9 *What is*. (2004-03-29) [online] – URL: http://whatis.techtarget.com/definition/0,,sid9_gci212127,00.html
- 10 Toroi, Teemu. *The SyncML Road Ahead – Application Development and Device Management*. SyncML Initiative, 2002-01-30.
- 11 Nokia. *Smart Messaging Specification, Revision 3.0.0*. 2000-12-18.
- 12 *ALD homepage*. (2004-06-14) [online] – URL: <http://www.thebunker.net/release-bluestumbler.htm>
- 13 Hjemås, A M, Brecke, H, Rønning, M, Eldhuset, K. *Mobile Device Management In The Future*. Fornebu, Telenor R&D, 2004. (Telenor R&D report R 20/2004)

For a presentation of Do van Thanh, please turn to page 2.

Anne Marte Hjemås has been working as researcher in the Service Platform unit of Telenor R&D, Trondheim since 2000. She holds an MSc degree in Technical Cybernetics from the Norwegian University of Science and Technology (NTNU) and has worked with system architecture and development for eight years. She has been involved in several commercial projects as well as research projects. Her current research interests are system architecture, service platforms for telecoms, device management, IM and Presence.

email: anne-marte.hjemas@telenor.com

Anders Bjurgård is Head of Business Logic and Terminals, and has been working in Telenor Mobile's unit for Product Development since 1997. He has been technical responsible for several product launches within the Consumer and Corporate area for mobile products. From 1995 to 1997 he was Technical Project Manager for Telenor/Digifone in Ireland with the responsibility for Service Platform and Core Network from bid to launch of the GSM net. His main interest today is the evolution of mobile services into all-IP services.

email: anders.bjurgard@telenor.com

Øystein Christian Løken has been working with terminals and device management related areas in Telenor Mobile Nordic since 2001. He holds a Bachelor degree in Cybernetics from Buskerud University College, Norway, a Bachelor degree in Microelectronic Engineering from Griffith University, Australia, and a Master degree in Information Technology from Bond University, Australia. He has been involved in several commercial projects in Telenor Nordic, such as the commercial launch of MMS and 3G, to more device management projects, such as the Telenor Entry project. He is currently product manager for the Telenor Nordic DMS platform and responsible for Telenor Nordic variant implementation.

email: Oystein-Christian.Loken@telenor.com