SEMINAR ON

MeeGo

By

GURUPRASAD T V

8th SEM Computer Science & Engg

MIT Kundapura

Udupi

Mobile: +919900652616

Mail id:guruprasadtv@gmail.com

INDEX

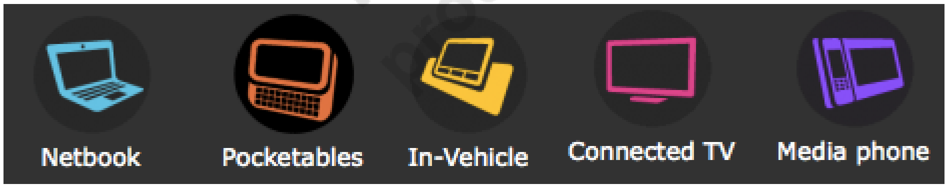
ABSTRACT

MeeGo is an open source operating system and many leading cell phone manufacturers, network service providers and third party software companies are waiting for its release. It is a Linux-based OS and is believed to be compatible with wide range of devices such as desktop computers, notebooks, cell phones, tablet PCs and even television sets. MeeGo powered device will allow the users to go online and access entertainment whether the user it at home or traveling. It has been revealed that the MeeGo OS is a merger between Nokia's Maemo and Intel's Moblin. Since MeeGo is backed up with powerful names like Nokia and Intel, many cell phone companies are waiting for its release so that they can make use of this OS. Since MeeGo is an open source project, developers will be able to modify MeeGo's code so that they can test and develop new software and enhance the performance of the existing apps. Developers can access the Moblin and Maemo communities to discuss about the MeeGo OS. The goal of developing the MeeGo OS is to provide users with an easy to use operating system and also to allow them to make use of existing software on compatible devices that were considered to be incompatible for running applications. Developers will be able to develop new applications that will be run on MeeGo supported devices. MeeGo is a platform where new ideas and new possibilities can be brought to the user. MeeGo is also believed to be an OS that will provide the user with refreshingly new platform for communicating, playing multimedia files and for using office based applications. MeeGo will also bring some rich features to mobile devices such as computation technology, graphic based apps, etc. It is expected to provide the user with a very simple user interface that is not only easy to use but will also let the user to customize it to the fullest.

INTRODUCTION

MeeGo is a Linux-based platform that is capable of running on multiple computing devices, including handsets, netbooks, tablets, connected TVs and in-vehicle infotainment systems. The primary goal of the merger of the Maemo and Moblin projects was to unify the efforts of the **Moblin** and **Maemo** communities and to enable a next generation open source Linux-platform suited for a variety of client devices. Most importantly, MeeGo will be doing so while: 1. Maintaining freedom for innovation 2. Continuing the tradition of community involvement (inherited from Maemo and Moblin) 3. Accelerating time-to-market for a new set of applications, services and user experiences With the merger, the MeeGo project has now the opportunity to significantly expand the market opportunities on a wide range of devices and support multiple chip architectures (ARM and x86) from the get go. MeeGo also provide a rich cross-platform development environment so applications can span multiple platforms and will unify developers providing a wealth of applications and services. Such opportunities for instances were out of reach for Maemo and Moblin individually. Furthermore, MeeGo is committed to work in the upstream projects so that everyone using those upstream projects can benefit from MeeGo's contributions to upstream projects.

Intended to run on a variety of hardware such as mobile phones, net books ,in-vehicle, smart TV and tablets each device share the same MeeGo internals just pasted with a different user interface on top.



With this in mind it has been demonstrated by Intel the power of MeeGo and how any device running the OS can connect to each other. For example you’re playing a movie on your phone and want to watch it on your TV, the two devices connect together via a cloud network and your movie will play on your TV and phone. It’s extremely versatile.

As the OS is open source there has a huge community following which means anyone that wants to get involved with the project can and should. One of the main projects that continues to grow everyday is the N900 adaptation team. These guys are responsible for getting MeeGo to run on the Nokia N900, currently the only commercial mobile phone that will run MeeGo, but let’s just say we wouldn't use it as our primary OS just yet as it is still in early stages of development.

MeeGo Includes:

- Performance optimizations and features which enable rich computational and graphically oriented applications and connected services development.
- No-compromise internet standards support delivering the best web experiences.
- Easy to use, flexible and powerful UI/app development environment based on Qt[(pronounced "cute") is a cross-platform application development framework widely used for the development of GUI programs (in which case it is known as a widget toolkit), and also used for developing non-GUI programs such as console tools and servers].
- Open source project organization managed by the Linux Foundation.
- State of the Art Linux stack optimized for the size and capabilities of small footprint platforms and mobile devices, but delivering broad Linux software application compatibility.

MAEMO AND MOBLIN BACKGROUND

Maemo Background

The Maemo project, initially created by Nokia *(http://www.maemo.org)*, provided a Linux-based software stack that runs on mobile devices. The Maemo platform is built in large parts of open source components and its SDK provides an open development environment for applications on top of the Maemo platform. A series of Nokia Internet Tablets with touch screen have been built with the Maemo platform. The latest Maemo device was the Nokia N900, powered by Maemo 5, that introduced a completely redesigned finger-touch UI, cellular phone feature, and live multicasting on the Maemo dashboard.

Moblin Background

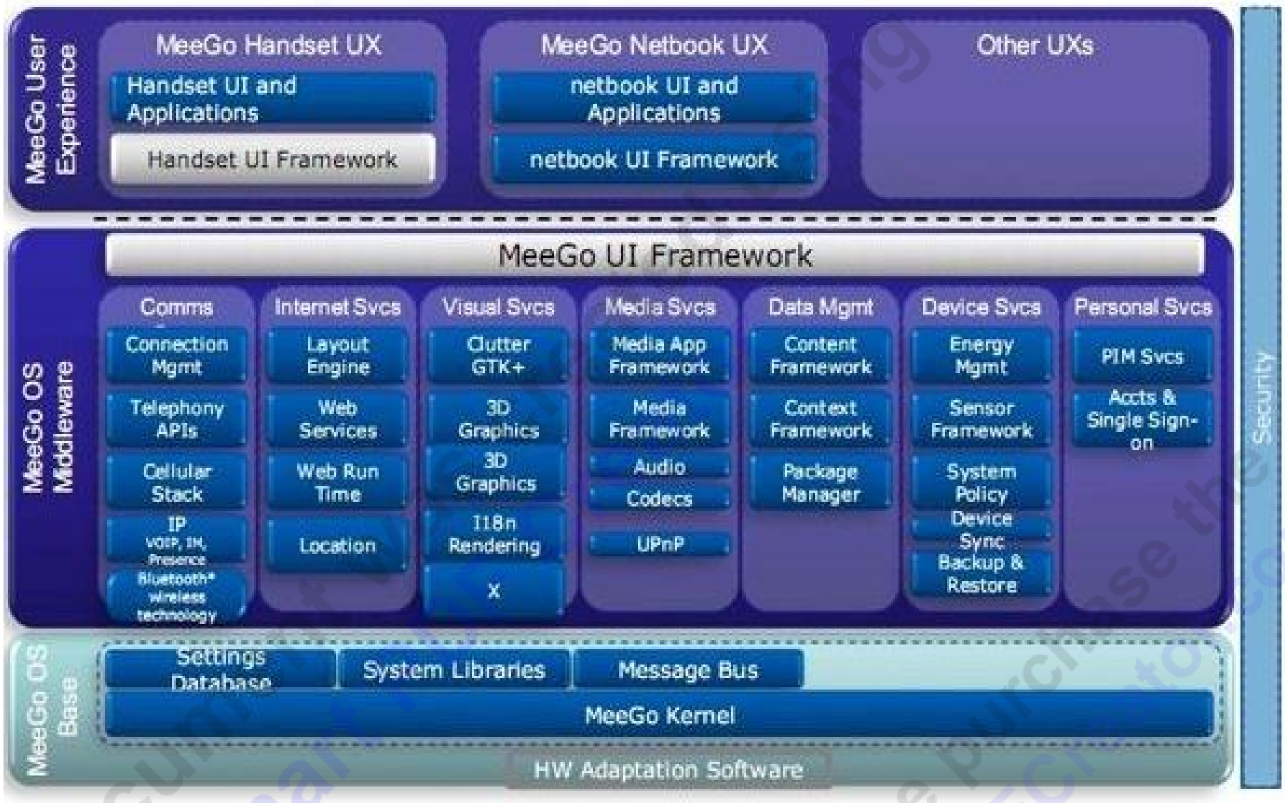
The Moblin project, short for Mobile Linux, is Intel's open source initiative *(http://www.moblin.org)* created to develop software for smartphones, netbooks, mobile internet devices (MIDs), and in-vehicle infotainment (IVI) systems, and other mobile devices. It is an optimized Linux-based platform for small computing devices. It runs on Intel Atom, an inexpensive chip with low power requirements. A unique characteristic to devices running Moblin is that they can boot up quickly and can be online within a few seconds.

FACTS WE MUST KNOW ABOUT MEEGO

- Full open source project governed according to best practices of open source development: Open discussion forums, open mailing lists, open technical steering committee meetings, peer review, open bugzilla, etc.
- Hosted under the auspices of the Linux Foundation

- Offers a complete software stack including reference user experience implementations.
- Aligned closely with upstream projects – MeeGo requires that submitted patches also be submitted to the appropriate upstream projects and be on a path for acceptance
- Offers a compliance program to ensure API and ABI compatibility (Compliance program discussed in a later section)
- Enables all players of the industry to participate in the evolution of the software platform and to build their own assets on MeeGo
- Lowers complexity for targeting multiple device segments
- Offers differentiation abilities through user experience customization
- Provides a rich cross-platform development environment and tools
- Offers a compliance program to certify software stacks and application portability
- Supports multiple hardware architectures
- Supports multiple app stores
- Has no contributors agreements to sign; instead it follows the same “signed-off-by” language and process as the Linux Kernel
- MeeGo 1.0 Netbook release supports the following languages: Japanese, Korean, Chinese Simplified, Chinese Traditional, Swedish, Polish, Finnish, Italian, Brazilian Portuguese, French, German, Spanish, Russian, Dutch, English, and British English.
- MeeGo incorporates Qt a software language used by Nokia to create applications for its mobile phones, which means that any existing apps that have been programmed in Qt will be able to run in MeeGo making apps work cross platform.

MEEGO ARCHITECTURE



MeeGo Software Architecture Overview

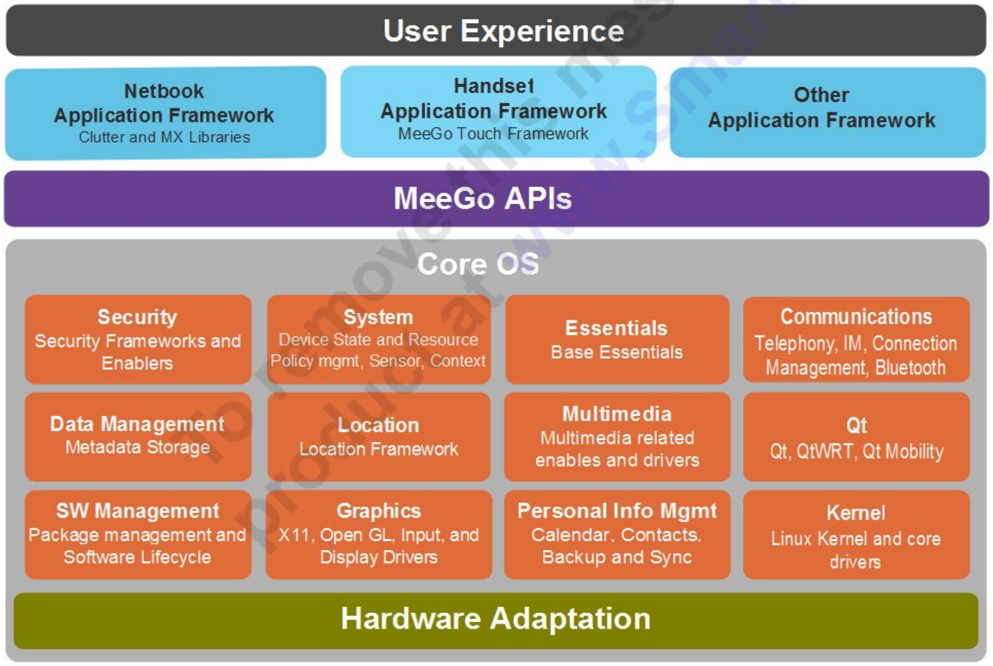
The MeeGo platform architecture can be viewed in three different ways:

- **Layer view**
- **Domain view**
- **API view**

Layer View

It shows the separation of different layers and user experience (UX) verticals. The Layer view consists of three layers: User Experience, Application API, and Core OS layers.

- **The user experience layer** contains the UX verticals. It provides reference user experiences for multiple platform segments. MeeGo 1.1 contains reference user experiences for handhelds and netbooks. Additional platform segments will be supported over time. The User experience layer provides the Application Framework for each device profile.
- **Application API layer** contains the MeeGo API. It provides the interface for application development. The current release includes Qt 4.7, Qt mobility 1.0, Open GL ES 1.1, and Open GL ES 2.0. Additional features are under development.
- **Core OS layer** contains all the middleware/OS service domains and the hardware adaptation services. It includes the Linux kernel and all the middleware needed to define hardware and usage model independent API for building both native applications and web run time applications. The Hardware Adaptation API is for adapting MeeGo to support various hardware architectures.



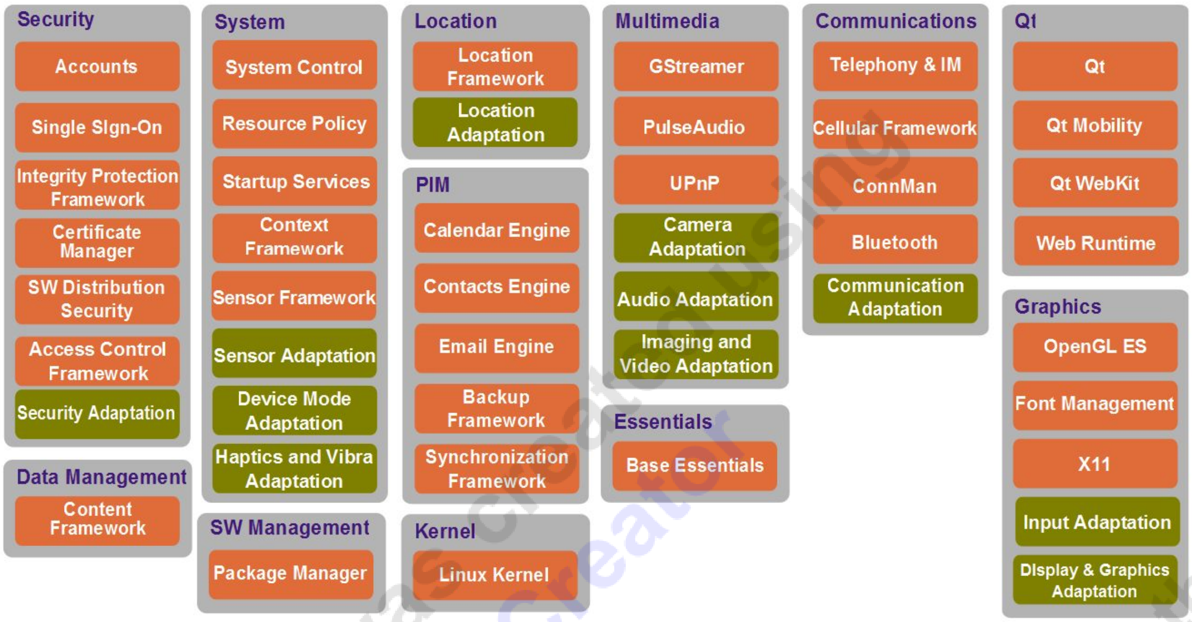
User Experience
Netbook Application Framework
Clutter and MX Libraries
Handset Application Framework
MeeGo Touch Framework
Other Application Framework
MeeGo APIs
Core OS
Security
Security Frameworks and Enablers
System
Device State and Resource Policy mgmt, Sensor, Context
Essentials
Base Essentials
Communications
Telephony, IM, Connection Management, Bluetooth
Data Management
Metadata Storage
Location
Location Framework
Multimedia
Multimedia related enables and drivers
Qt
Qt, QtWRT, Qt Mobility
SW Management
Package management and Software Lifecycle
Graphics
X11, Open GL, Input, and Display Drivers
Personal Info Mgmt
Calendar, Contacts, Backup and Sync
Kernel
Linux Kernel and core drivers
Hardware Adaptation

MeeGo Core OS architecture is grouped into *domains*, based on functionality in that area:

- ✓ **Security** - Security framework and enablers
- ✓ **Data Management** - Meta-data storage
- ✓ **Software Management** - Package Management and software lifecycle
- ✓ **System** - Device State and Resource Policy Managmeent, Sensor, Context
- ✓ **Location** - Location Framework
- ✓ **Graphics** - X11, OpenGL, input and Display drivers
- ✓ **Essentials** - System essential libraries
- ✓ **Multimedia** - Multimedia related enablers and drivers
- ✓ **Personal Information Management** - Calendar, Contacts, Backup, and Sync
- ✓ **Communication** - VOIP, IM, Presence, Cellular Telephony, and IP Connectivity
- ✓ **Qt** - Qt, QtWRT, Qt Mobility
- ✓ **Kernel** - Linux Kernel and core drivers

Domain View

shows the grouping of subsystems into architecture domains, based on similarities in technology and functionality



The Domain view expands each domain and details the subsystems required to provide that functionality.

Security:

Security domain is responsible of security deployment across the system. It provides enablers for platform security and user identity.

- ✓ **Accounts** - Provides a storage solution for user accounts. Applications which need to store and access user settings for the service they provide over a user account will use the Accounts API. Instant messaging, e-mail, calendar, and sharing are examples of such applications.
- ✓ **Single Sign-On** - Responsible for providing secure storage for credentials and framework for authentication plugins to different services
- ✓ **Integrity Protection Framework** - Integrity protection of executables, configuration, and data files.

- ✓ **Certfificate Manager** - Services for storing and validation of security certificates for various purposes (such as email, wifi, and browsing).
- ✓ **Software Distribution Security** - Security aspects of software distribution including new application installations and updates.
- ✓ **Access Control Framework** - Access control enforcement and access control policy for the device
- ✓ **Security Adaptation** - Platform specific abstraction of security and crypto services.

Data Management:

Data Management domain provides services for extracting and managing file meta-data (for example to support extracting and searching metadata for media files). The Data Management domain includes the following subsystem:

- ✓ **Content Framework** - Tracker provides indexing, meta-data extraction, and search capabilities for a variety of data types, including media files, and documents.

Software Management:

Software Management domain is responsible for package manager and its backend functionality.

- ✓ **Package Manager** - PackageKit uses distribution package management tools to make installing and updating software on devices easier. It is a system activated daemon meaning that it is only run when the user is using the tools, and quits when it is no longer used.

System:

System domain is responsible for device state/mode handling, time management, policy control, startup services, and sensor abstraction.

- ✓ **System Control** - Device state and time management
- ✓ **Resource Policy** - Plugin based framework for audio, video, and system policy management.
- ✓ **Startup Services** - Components related to system startup.
- ✓ **Context Framework** - High level API to numerous context properties of the device.
- ✓ **Sensor Framework** - Provides an interface to hardware sensors through logical sensors.
- ✓ **Sensor Adaptation** - Sensor specific plugins for sensor framework
- ✓ **Device Mode Adaptation** - Hardware abstraction layer for device mode related information (such as watchdogs, temperature sensors)
- ✓ **Haptics and Vibra Adaptation** - Hardware abstraction layer for vibra and haptics devices

Location:

Location domain provides location services.

- ✓ **Location Framework** - GeoClue provides location data combined from number of sources, such as GPS, GSM cell, or wifi network.
- ✓ **Location Adaptation** - Hardware abstraction layer for location source devices such as GPS

Kernel:

Kernel domain contains Linux kernel and device drivers.

- ✓ **Linux Kernel** - Linux kernel 2.6.35 or newer.

Personal Information Management:

Personal Information Management domain enables managing user data on the device, including managing calendar, contacts, tasks, and retrieving data about the device context (such as device position, cable status). The domain includes the following subsystems:

- ✓ **Calendar Engine** - Calendar engine provides an interface for accessing calendar data.
- ✓ **Contacts Engine** - Contacts engine provides an interface for accessing contact data.
- ✓ **Email Engine** - Email engine provides an interface for accessing emails.
- ✓ **Backup Framework (*)**
- ✓ **Syncronization Framework** - Synchronizing calendar, email, and contacts data between different devices via various transport layers like USB and Bluetooth

Multimedia:

Multimedia domain provides audio and video playback, streaming, and imaging functionality to the system. In general, the domain takes care of the actual audio and video data handling (retrieval, demuxing, decoding and encoding, seeking, etc.). The domain includes the following subsystems:

- **Imaging and Video Adaptation** - Platform specific codecs and containers for GStreamer
- **Camera Adaptation** - Platform specific codecs and containers for GStreamer. Adaptation interface is CameraBin.
- **UPnP** - Universal Plug and Play provides a UPnP stack, the UPnP profile for audio and video.
- **Gstreamer** - GStreamer, through its plug-ins, provides playback, streaming, and imaging functionality to the system.
- **Audio Adaptation** - Platform specific modules for Pulse Audio
- **Pulse Audio** - The audio subsystem handles audio inputs, post and pre processing, and outputs in a system. The purpose is to provide a proxy between audio applications and audio hardware.

Essentials:

Essentials domain provides all system essential packages.

- **Base Essentials** - Fundamental system tools and libraries

Communications:

Communications domain provides Cellular and IP Telephony, Instant Messaging, Presence, Bluetooth, and Internet Connectivity services.

- **IP Telephony, Instant Messaging and Presence** - Telepathy is a modular communications framework that enables real-time communication via pluggable protocol backends.

- ✓ **Cellular Framework** - oFono provides cellular telephony stack and services in MeeGo. Plugin based architecture supports multiple platforms and modems.
- ✓ **ConnMan** - Connection Manager provides services for managing internet connections.
- ✓ **Bluetooth** - The Bluetooth subsystem consists of the Linux Bluetooth stack BlueZ, as well as related extensions.
- ✓ **Communication Adaptation** - Platform specific modules for WiFi and Bluetooth devices and oFono plugins for different platforms and modems.

Qt:

Qt domain contains cross platform toolkits such as Qt, Qt Mobility, Qt WebKit, and Qt WebRuntime.

- ✓ **Qt** - Qt application and UI toolkit.
- ✓ **Qt Mobility** - Qt Mobility APIs for MeeGo.
- ✓ **Qt Webkit** - MeeGo provides Qt Webkit as a layout engine. It renders web content (HTML, XML, XHTML, SVG, CSS, JavaScript, etc.) for on-screen display within applications.
- ✓ **Web Runtime** - Provides an execution environment for Web Widgets and extends the standard JavaScript environment with device-specific APIs providing access to other subsystems.

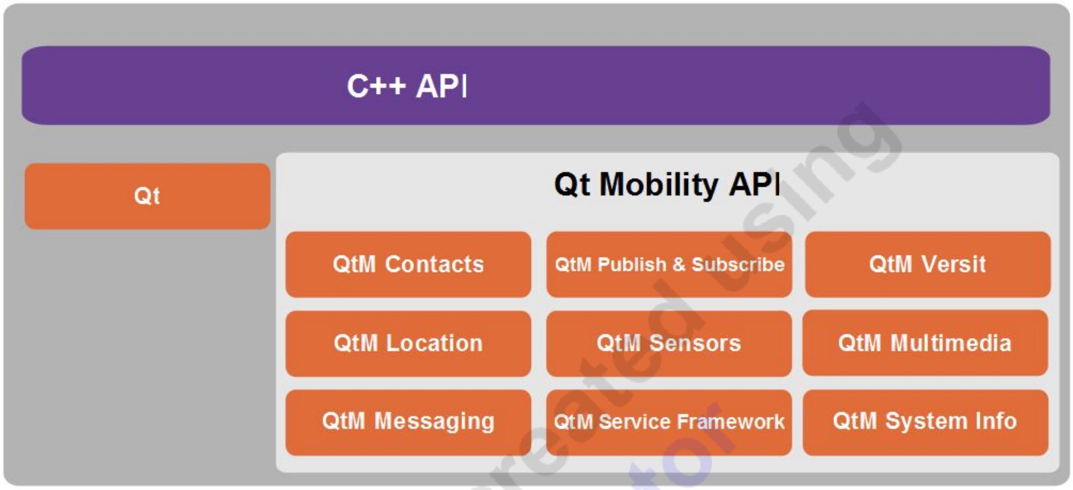
Graphics:

Graphics domain enables the core 2D and 3D graphics capabilities for the platform, including support for rendering internationalized text and taking advantage of underlying hardware platform acceleration for graphics. The Graphics domain includes the following subsystems:

- ✓ **Font Management** - Service to locate fonts within the system and select them according to requirements specified by applications
- ✓ **Input Adaptation** - Input adaptation abstracts the hardware behind drivers and exposes an input event interface for user space. Hardware buttons, qwerty keyboard, and touch screen are provided as input devices. Typical HW buttons are: power button, camera, volume up, and volume down.
- ✓ **X11** - Implementation of the X11 Window system with architecture specific drivers, patches and configuration.
- ✓ **OpenGL ES** - Provides Khronos interfaces and implementation of OpenGL, OpenGLES and EGL. Includes also platform specific implementation of GL/ES driver and libraries.
- ✓ **Display and Graphics Adaptation** - Framebuffer and display panel related platform specific abstraction.

API view:

This shows the grouping of MeeGo API into functional areas.



The MeeGo API is based on Qt and Qt Mobility.

Qt:

Qt provides application developers with the functionality to build applications with state-of-the-art graphical user interfaces. Qt is fully object-oriented, easily extensible, and allows true component programming.

Upcoming APIs:

→MeeGo Touch Framework

The MeeGo Touch Framework provides the features needed for developers creating applications for touch-enabled devices. Features include standardized window navigation, list and other widget behavior, and common theming for components.

→MeeGo Web Runtime→Web Runtime (WRT) allows web developers to use standard web languages (HTML, CSS, and JavaScript) to create applications for mobile devices. WRT exposes the features of the underlying platform so that applications can interact with device data and combine location-based context with web information.

→Qt mobility

Qt Mobility extends Qt with libraries providing additional features for applications targeting mobile platforms. However, these APIs allow the developer to, with ease, use features from one framework and apply them to phones, netbooks, and non-mobile personal computers.

HARDWARE ADAPTATION SOFTWARE

There are multiple software components that a hardware vendor must provide for MeeGo to run successfully on their platform architecture, including platform kernel drivers, core architecture additions, kernel configuration, X software additions and configuration, modem support, and hardware specific media components. You can learn more about the hardware enabling process here. These specific software components are called the hardware adaptation software and are detailed below.

The MeeGo Core OS defines interfaces for platform dependent hardware. It's the responsibility of a chipset's hardware adaptation software to implement these interfaces. The hardware adaptation software is divided into to the following *adaptation subsystems*:

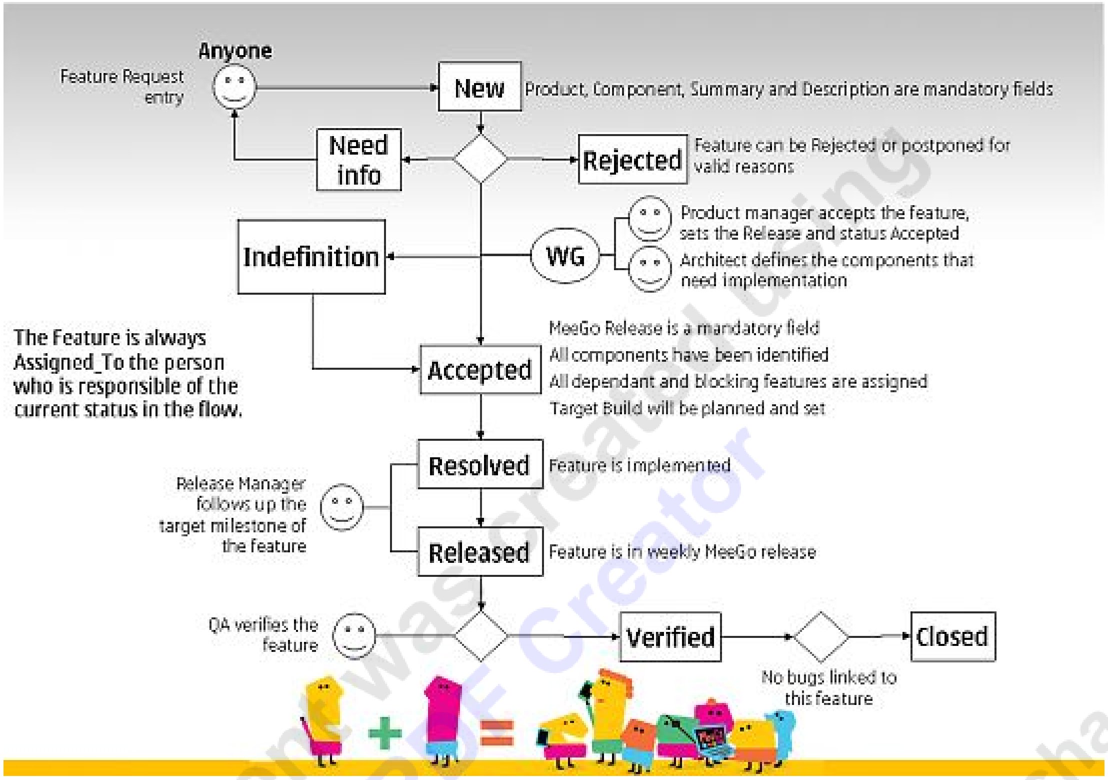
- **Security**
- **Sensor**
- **Device Mode**
- **Haptics and Vibra**
- **Audio**
- **Camera**
- **Imaging and Video**

- **Location**
- **Cellular**
- **Connectivity**
- **Input**
- **Display and Graphics**

REQUIREMENTS

MeeGo requirements are defined and managed in a public process and toolchain based on common Bugzilla. The toolchain provides a closed-loop lifecycle management from initiation of a Feature Request, to planning of requirements, implementation, integration, and verification of it in the same tool throughout the whole lifecycle.

Requirements are initiated as Feature Requests using the MeeGo Featurezilla tool at bugs.meego.com. The product manager is responsible for facilitating this front-end and for further preparation the Feature Requests for decision making in the Working Group. Working Groups, and the MeeGo Technical Steering Group for Core, approve the Features for a specific MeeGo Release and publish this in the form of a Roadmap and approved Features in Featurezilla. Furthermore, the approved features are tuned to finer detail of implementable engineering requirements. Requirements are then used to create a Release Plan, detailing which requirements are planned to be integrated in every weekly MeeGo release. Once a requirement is integrated, MeeGo QA verifies it by running the corresponding test cases. Bugs are filed against the requirements and marked with a dependency. Eventually, once there are no longer bugs preventing closing the feature, it is closed in Featurezilla. Requirements lifecycle management is defined as:


Anyone
Feature Request entry
New
Product, Component, Summary and Description are mandatory fields
Need info
Rejected
Feature can be Rejected or postponed for valid reasons
Indefinition
WG
Product manager accepts the feature, sets the Release and status Accepted
Architect defines the components that need implementation
The Feature is always Assigned_To the person who is responsible of the current status in the flow.
Accepted
MeeGo Release is a mandatory field
All components have been identified
All dependant and blocking features are assigned
Target Build will be planned and set
Resolved
Feature is implemented
Release Manager follows up the target milestone of the feature
Released
Feature is in weekly MeeGo release
QA verifies the feature
Verified
Closed
No bugs linked to this feature

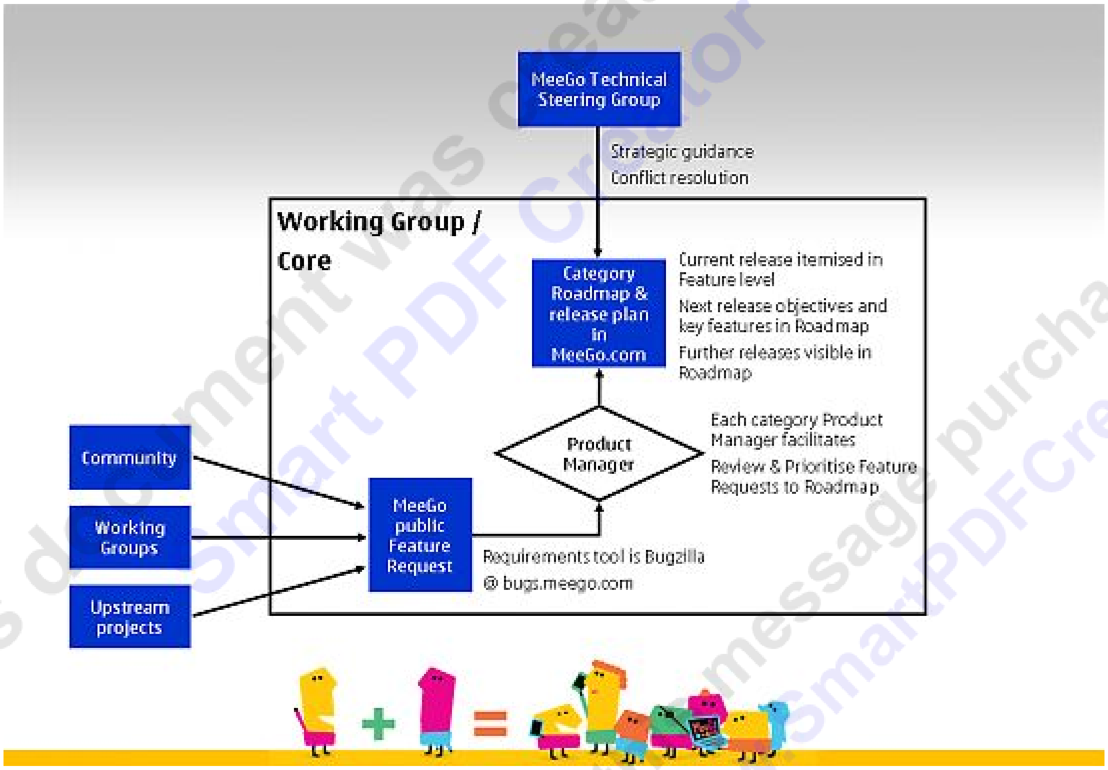
WHAT MAKES MEEGO UNIQUE?

- Broad industry support and adoption in all categories
- Great UX
- Great OS
- Multi-category
- Multi-HW
- Active community
- Open governance model
- Close relation with upstream projects
- A partnership project
- Compliance program
- Successful app store story

- Ability to provide areas for differentiation and support for proprietary add-ons

MEEGO ROADMAP

The MeeGo Roadmap will be developed with an open process that is being set up currently.


MeeGo Technical Steering Group
Strategic guidance
Conflict resolution
Working Group / Core
Category Roadmap & release plan in MeeGo.com
Current release itemised in Feature level
Next release objectives and key features in Roadmap
Further releases visible in Roadmap
Product Manager
Each category Product Manager facilitates
Review & Prioritise Feature Requests to Roadmap
Community
Working Groups
Upstream projects
MeeGo public Feature Request
Requirements tool is Bugzilla @ bugs.meego.com

The principal idea is to have category-specific working groups responsible for defining the roadmap for their corresponding areas, like netbook and handset. Similarly, the Core Program is responsible for developing the Core Roadmap.

The initial roadmap is outlined for MeeGo releases for year 2010. The MeeGo 1.0 release focuses on providing the core OS components and a rich user experience for netbooks. MeeGo 1.1 further develops the core functionality and focuses specifically on the Handset User Experience.

Nokia's First MeeGo Smartphone is **Nokia N950**. It supports 4G technology and powered by MeeGo platform and runs on 1GHz processor. The Nokia N950 has a display type 4.0 inches large full capacitive toch screen display. This will run on the MeeGo platform and will operate on Quad band GSM frequency. This model comes with 8.0 Mega pixel camera with auto focus, Flash and digital zoom. Other expected feature of this cell phone will be Wi-Fi, Bluetooth, 4G connectivity; GPS with A-GPS, MicroSD card support has GPRS.

BENEFITS OF THE MEEGO SOFTWARE PLATFORM:

The MeeGo open source project is unique in that it offers benefits to everyone in the ecosystem starting from the developer all the way up to the operator and the industry as a whole. Meego allows participants to get involved and contribute to an industry-wide evolution towards richer devices, to rapidly address opportunities and to focus on differentiation in their target markets.

Benefits to Open Source Developers:

The MeeGo project is a true open source project hosted by the Linux Foundation and governed by best practices of open source development. From meego.com, as an open source developer, you have access to tools, mailing lists, discussion forum, accessibility to technical meetings, and multiple options to make your voice heard over technical and non-technical MeeGo related topics. Furthermore, all source code contributions needed for MeeGo will be submitted to the upstream open source projects from which MeeGo will be built.

Benefits to Application Developers

As an application developer, MeeGo significantly expands the market opportunities for you being the only open source software platform that supports deployments across many computing device types. MeeGo offers Qt and Web runtime for application development, cross platform environments, so application developers can write their applications once and deploy easily on many types of MeeGo devices or even on other platforms supporting the same development environment.

Benefits to Device Manufacturers

MeeGo helps accelerate time to market using an off-the-shelf, open source and optimized software stack targeted for the specific hardware architecture the device manufacturer is supporting. From a device manufacturer perspective, MeeGo lowers complexities involved in targeting multiple device segments by allowing the use of the same software platform for different client devices. In addition, as an open source project, MeeGo enables device manufacturers to participate in the evolution of the software platform and build their own assets for it through the open development model.

Benefits to Operators:

For operators, MeeGo enables differentiation through user interface customization. Although many devices can be running the same base software platform, they can all have different user experiences. Furthermore, it provides a single platform for multitude of devices, minimizing the efforts needed by the operators in training their teams and allows their subscribers to be familiar with the experience common to many device types.

Benefits to the Linux Platform

In addition, MeeGo is helpful for Linux as a platform as it combines mobile development resources that were recently split in the Maemo and Moblin projects into one well-supported, well-designed project that addresses cross-platform, cross-device and cross-architecture development. One major benefit from the MeeGo project is that all other Linux mobile and desktop efforts that use the components as MeeGo will benefit from the increased engineering efforts on those components. This is the power of the open source development model.

DIFFERENCE BETWEEN MEEGO AND ANDROID

1) MeeGo and Android both are open sourced Linux based smart phone operating system whereas Meego is a coordinated effort by the Linux community but Android not.
2) Android project is currently carried out by Google and MeeGo is carried out by Intel and Nokia as a joint venture.
3) Android already have a huge application market having more than 200,000 applications whereas in MeeGo is currently under development. But some of the Nokia OVI apps and Intel AppUp will run on MeeGo.
4) Android already captured considerable mobile market and MeeGo is still under development and not released for commercial use.
5) Android OS is application based, whereas the MeeGo OS is a handheld computer OS while Android's Java based applcations are easy to build for beginners, Maemo & MeeGo's development environments allow for much richer applications.

CONCLUSION:

Factors which makes MeeGo Succeed:

- Great UX –Nokia and Intel making joint strategic investment to make this happen.
- Great OS –Nokia and Intel today make most of the client Linux development invesment. Joint talent pool of 1000+ best Linux platform developers in the planet
- Multi-category –Handset, Tablet, Automotive, TV, Netbook, …
- Multi-HW –ARM, Intel, multi-vendor
- Multi-lateral –Not a monopoly business platform but a partnership for technical de-facto standard

MeeGo phone will push the boundaries of processing power. It will be a very high end product. This is an opportunity to create something well beyond what others are doing. This phone will be something everyone wants to own. It will try to satisfy all needs into one plate. MeeGo phone will be capacitive, more stylish and more beautiful with simpler UI that average users will find as equally compelling as tech leaders. It has potential to be something deeper than typical Smartphone. All future flagships will be built on MeeGo. The phone will be on Nseries (not rumored S series). MeeGo will have higher experience level than Symbian

Summary of Key Messages:

- Smartphones have become Consumer Computers
- Linux will take over the Smartphone market
- MeeGo is challenging Android as the Smartphone

BIBLIOGRAPHY

Reference:

http://meego.com

Architecture:

http://meego.com/developers/meego-architecture

SDK download:

http://wiki.meego.com/Getting_started_with_the_MeeGo_SDK_for_Linux

MeeGo API:

http://meego.com/developers/meego-api

Developer Guides:

http://wiki.meego.com/Developer_Guide