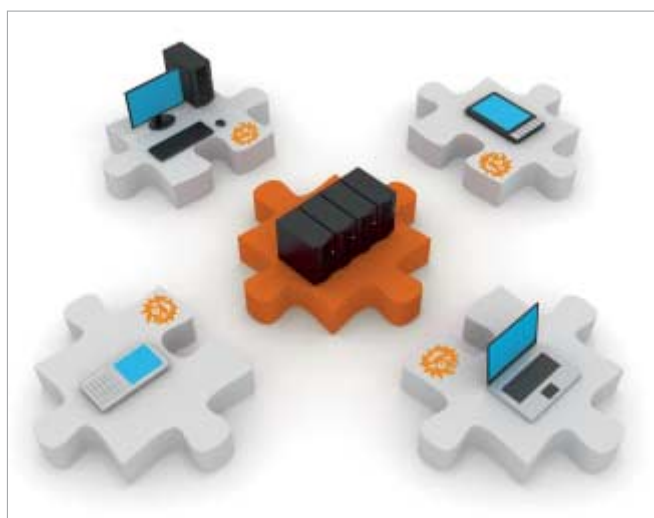


# Client Framework for Building Rich Multimedia Applications

MONSTER – Multimedia Open InterNet Services and Telecommunication EnviRonment



The Fraunhofer Institute FOKUS launches MONSTER at the Mobile World Congress 2009 – a client framework prototype which is an extendible plug-and-play service platform providing developers with a unified communication API for developing mash-up Internet and telecommunication services such as telephony, chat, messaging, social networking and contacts. The framework enables the concept to “develop once and run on many platforms” and can be downloaded at: [www.monster-the-client.org](http://www.monster-the-client.org).

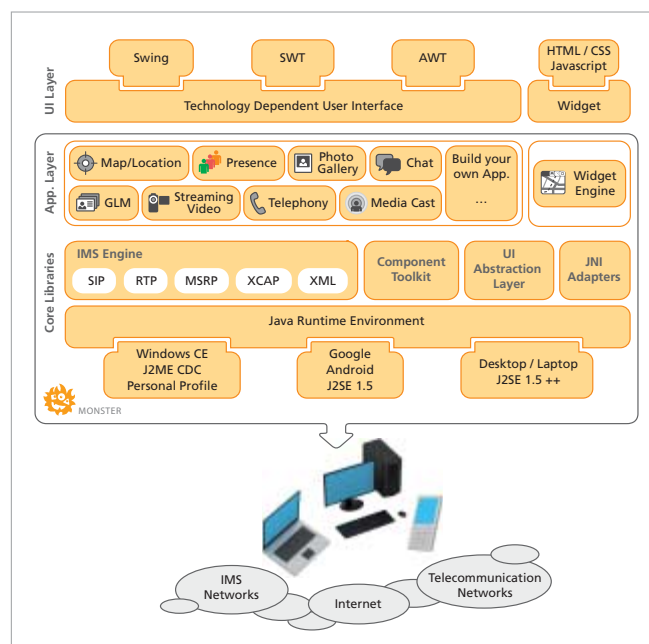


Extendible Plug-and-play Framework for Multiple Platforms

## Conceptual Architecture

In recent years there has been a proliferation of mobile and fixed devices matched by a growing number of services to be accessed from them. For service developers this often means that an application written for one platform must be rewritten repeatedly for other platforms, a problem that increases time to market, reduces market size

and hinders uptake of services. MONSTER seeks to address this situation by offering a middleware platform that can run on multiple devices and that offers a framework for hosting convergent rich multimedia applications that make use of telecommunication features. We have used the Java programming language to develop this framework because of its ready availability on multiple platforms.



MONSTER Architecture

The middleware layer of MONSTER consists of a set of *core libraries* that provide framework functionalities, and an application layer where basic applications are deployed and on which developers can program and host their own applications. The core libraries consist of the following components

The **IMS Engine** is an implementation of the JSR 281 specification for IP Multimedia Subsystem (IMS) service developments on the client side. Application developers can use this library to create IMS based applications

without having to deal with the complexity of IMS protocols and standards. The IMS protocols supported include

- A Session Initiation Protocol (SIP) stack and IMS service APIs for SIP signaling for creating multimedia sessions, handling event notifications and registering on the IMS network
- A Real-Time Protocol (RTP) stack for real time audio and video streaming
- A Message Session Relay Protocol (MSRP) stack for messaging and chat services
- An Extensible Markup Language (XML) Configuration Access Protocol (XCAP) API that allows applications to read, write and modify application configuration data, stored in XML format on a server which can be directly accessed by HTTP
- XML handlers for decoding XML files into program objects and encoding them back into XML files for storage

The **Component Toolkit** implements several design patterns and framework classes which developers can use to develop service oriented applications. This toolkit glues the framework services together by injecting inter-service dependencies, and life cycle and resource management of the applications.

The **UI Abstraction Layer** decouples the application logic from the UI presentation layer, thus enabling the same services to be presented using different presentation layers tailored to the target device platform.

The **Java Native Interface Adapters** implement Java native wrappers to access the device's native service API. This enables developers to mash their applications with services offered on the native device's API.

The *application layer* contains basic services which are deployed together with the framework. The underlying

core libraries expose service APIs which developers can use to implement their own services on this layer. The basic multimedia applications include soft telephony, video streaming, group list management and presence, chat, media cast, and photo gallery. On the application layer the framework also provides a Widget Engine on which web developers can implement widgets that consume the services from the application layer through JavaScript APIs.

### Key Benefits and Advantages

---

- Shortens development time
- Decouples application development from the user presentation layer which also facilitates branding of applications
- Runs on multiple platforms (Google Android, Windows Mobile, Linux, Windows, Mac OS)
- Allows easy application update with OMA DM
- MONSTER builds on open standards – the IMS Engine aligns with the 3GPP TS 24.229 (Rel. 7) and JSR 281 specification, the widget engine aligns to W3C standards for widget resource packaging and dynamic deployment
- The free download and Open API concept makes MONSTER flexible and extendible so that own applications can be plugged in

### Free Download from [www.monster-the-client.org](http://www.monster-the-client.org)

---

For developers who want to work with the framework to build their own applications, the project website offers download links for various platforms, developer documentation as well as a user forum.

Contact: [info@monster-the-client.org](mailto:info@monster-the-client.org)