I

Requirements for Software Modularity on the Home Gateway

Track	SWEX
Editor:	Andreas Sayegh
Date:	June 2011
Document number:	RD008-R3
Version:	1.0
Status:	Published Version
Keywords:	Software Modularity, Software Management, Service Flexibility, Service Platform, Execution Environment

Abstract

This document contains information regarding modular software deployments onto the home gateway. Operators or third parties may create modular software applications, but only the operator controls the remote deployment onto the devices. Moreover, modular software applications must run in a dedicated virtual execution environment to avoid conflicts and interferences with the natively installed software.

This document reflects generic requirements valid for any modular execution platform as well as technology-specific requirements, currently for OSGi technology.

33

1 Important notice, IPR statement, disclaimer and copyright

The Home Gateway Initiative (HGI) is a non-profit making organization which publishes guidelines, requirements documents, and test plans for broadband equipment and services which are deployed in the connected home. The scope of HGI's work covers Home Gateways, interconnection with the broadband access network, home networks requirements, software requirements, and other technology building blocks required to deliver broadband services.

This document is the output of the Working Groups of the HGI and its members as of the date of release. Readers of this document must be aware that it can be revised, edited or have its status changed according to the HGI working procedures.

The HGI makes no representation or warranty on the contents, completeness and accuracy of this publication.

This document, though formally approved by the HGI member companies, is not binding in any part on the HGI members.

IPRs essential or potentially essential to the present document may have been declared in conformance to the HGI IPR Policy and Statutes available at the HGI website www.homegateway.org.

Any parts of this document may be freely reproduced (for example in RFPs and ITTs) by HGI and non-HGI members subject only to the following:

- HGI Requirement numbers not being changed
- An acknowledgement to the HGI being given in the resulting document.

Trademarks and copyrights mentioned in this document are the property of their respective owners.

The HGI membership list as of the date of the formal review of this document is: Advanced Digital Broadcast, Alcatel-Lucent, Arcadyan, Atheros, AVM, Belgacom, Bouygues Telecom, Broadcom, Broadlight, BT, Celeno, Cisco, Comtrend, Deutsche Telekom, D-Link Corporation, DSP Group, Echelon EMEA, Ericsson AB, Fastweb SpA, France Telecom, Freescale Semiconductor, Hitachi, Huawei, Ikanos, Intel, JDSU, KDDI, KPN, Lantiq, LG-Ericsson Co Ltd, Marvell Semiconductors, Mindspeed, Mitsubishi, NEC Corporation, Netgear, NTT, Ping Communication, Portugal Telecom, ProSyst, Ralink, Sagemcom, Samsung, Sercomm Corp., Shenzhen GongJin Electronics, Sigma, SoftAtHome, Spidcom, Stollmann, Sumitomo, Swisscom AG, Technicolor, Telecom Italia, Telekom Slovenije, Telekomunikacja Polska, TeliaSonera, Telstra, TNO, Trac Telecoms & Radio Ltd, Transwitch, Vodafone, Vtech, Wind River Systems, Zarlink, ZTE, ZyXEL.

2 Acronyms

ACS	Auto-Configuration Server
API	Application Program Interface
ASP	Application Service Provider
BBF	Broadband Forum
BSP	Broadband Service Provider
CDC	Connected Device Configuration
CPU	Central Processing Unit
CWMP	CPE WAN Management Protocol
DHCP	Dynamic Host Configuration Protocol
DSL	Digital Subscriber Line
EE	Execution Environment
FTP	File Transfer Protocol
HG	Home Gateway
HG_API	Home Gateway Application Programming Interface
HG_EE	Home Gateway Execution Environment
HG_SP	Home Gateway Service Platform
HGI	Home Gateway Initiative
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ID	Identifier
IGD	Internet Gateway Device
ISP	Internet Service Provider
J2ME	Java 2 Mobile Edition
JCP	Java Community Process
JRE	Java Runtime Environment
JSR	Java Specification Request
MA	Management Agent
NAT	Network Address Translation
NTP	Network Time Protocol
OS	Operating System
RMS	Remote Management System
RPC	Remote Procedure Calls
SIP	Session Initiation Protocol
SP	Service Platform
UPnP	Universal Plug & Play
URL	Uniform Resource Locator
USB	Universal Serial Bus
WAN	Wide Area Network

3 Scope and Purpose of the Document

Service delivery to residential customers beyond triple play services demands, for the integration of home devices and appliances into WAN located service infrastructures. Especially for retail devices, such integration requires a software-based integration in the home network, and the variety of available technologies makes it very difficult to cope with all of them using standardized functions. So, in order to achieve the next level of service integration, there is a need for software flexibility on the main operator-controlled device in the home: the home gateway.

Trying to accomplish software flexibility needs by iterating the firmware is possible. However, doing this causes significant problems: Each new version must be fully tested, and different versions are required for different application areas. Maintaining different versions of firmware for several HG models would further complicate configuration management.

The solution discussed in this document integrates a software execution platform into the firmware allowing for installing, updating, uninstalling as well as starting and stopping software modules, while the underlying firmware image remains untouched.

This document contains a home gateway software modularity architecture specification based on function blocks, a role and entity model, and derived requirements for the home gateway. Requirements are specified not only for a software execution platform, but also for an API that allows software modules to access the home gateway functions.

The requirements sections are divided into two areas: generic requirements, which apply to any technology used as software execution platform, and specific requirements for selected technologies. Specific requirements are stated only for OSGi technology.

Sections 3 and 4 of this document are informative; sections 5 and 6 are normative.

3.1 Relation to HGI Residential Profile

This document contains requirements over and above the HGI Residential Profile (RD001), in case an HG should run a software module execution environment. All requirements of the HGI Residential Profile still apply.

Some of the functions defined in the HGI Residential Profile may be implemented as modules. These functions have some characteristics in common:

- Control functions rather than data plane transmission functions.
- Functions with easy interfacing from other modules.
- Functions which may be implemented once and run on different HG models
- Functions where flexibility is needed in choosing the appropriate version of that function
- Functions suitable to a multi-service provider model, in which different service providers may provide similar functions on different HGs

The following list serves as examples of modules from the HGI Residential Profile which may be amenable to implementation as modules:

- DHCP Server (R108) – A DHCP server is a fairly stand alone application, handing out local IP addresses and managing their lifetime. The information gathered by the DHCP server can easily be given to other (authorized) modules (MAC addresses, DHCP options).
- UPnP IGD (R127) – A UPnP IGD [6] service for example would be easy to implement on top of an OSGi service platform, because OSGi provides a standardized UPnP stack [4], given that there is an interface to manage the lower functions of the HG.
- Local Management Remote User Interface (R198) – An operator could see some advantage in bringing the same implementation of the LM Remote UI to different models of HGs, following the operator corporate design, and maintaining only one code basis.
- DynDNS Client (R249) – A DynDNS client again is a fairly stand alone application, registering a public IP address at a certain domain server. Having it as a module has the advantage that several registration protocols for different dynamic DNS services can be supported.

- NTP Client (R250) – Needs access to system date and time management of the HG.

There are, conversely, some functions defined in HGI Residential Profile that are not recommended to run as modules, but to implement only in native software.

Some of these functions are the networking features (e.g. routing, bridging, NAT), the basic HG remote management and the firmware upgrade from the RMS. The main factor is that these functionalities must be preserved in the case of an execution environment fault. Besides, these functionalities are generally already offered in native software, and gain from a direct interaction with the hardware.

4 Architecture

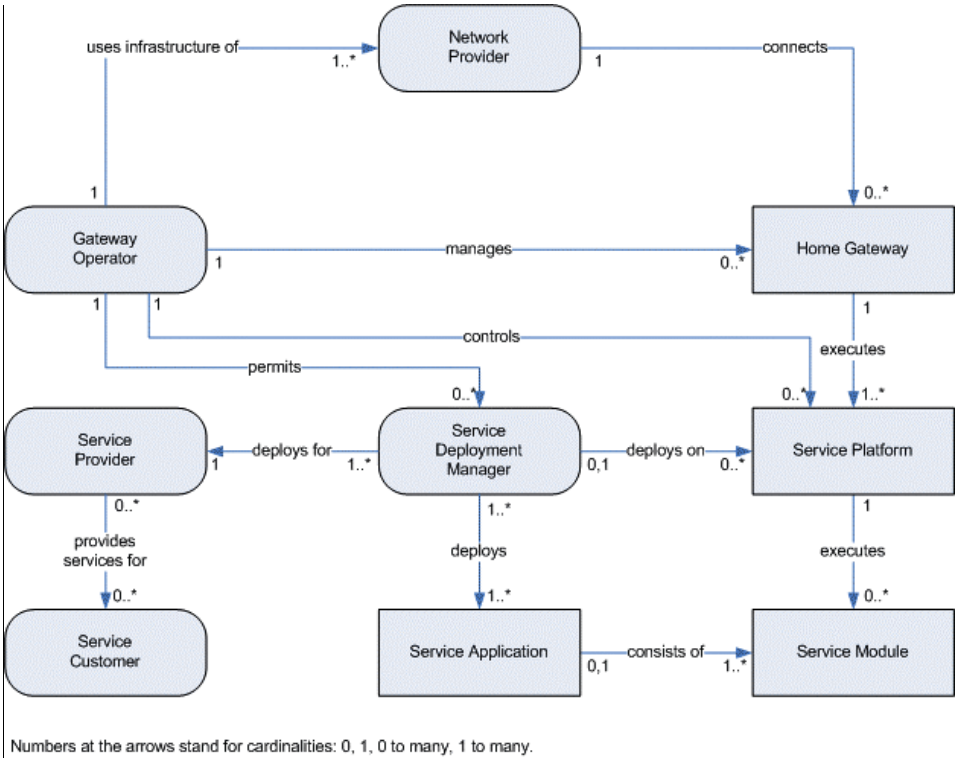
4.1 Roles and Entities

This section defines operational roles and entities for an execution environment on the HG. This model is an adapted version of the business architecture model from the OSGi Alliance, but only contains roles that have meaning in *operating* an execution environment.

The HGI roles of "Broadband Service Provider" (BSP) and "Application Service Provider" (ASP) as defined in [1] may apply to several of the following roles, but surely to the "operator" role. For example, a BSP might take the role of the Service Deployment Manager, but might also chose to delegate this function to another company. However, it's subject to a BSP's business policy which roles to take and which to delegate.

The following figure depicts roles (symbolized through round-corner rectangles) and entities (regular rectangles). Roles are natural or legal persons (e.g. companies), while entities are machines/software.

Fig. 1. Roles and Entities Relations



The various roles and entities are defined as follows:

- **Home Gateway Service Platform (HG_SP)** - The primary function of the Home Gateway Service Platform (short: Service Platform) is to manage the execution lifecycle of Service Modules.

  The Service Platform is capable of dynamically loading, activating, deactivating, updating, and unloading the Service Modules.

  The HG_SP is part of the HG_EE as defined in 4.2.
- **Home Gateway (HG)** - The Home Gateway (HG) hosts one or more Service Platforms.
- **Gateway Operator** (or just operator) - The primary responsibility of the Gateway Operator is to control who is allowed to deploy services to the Service Platform in question i.e. control which Service Deployment Managers are allowed to manage the particular Service Platform.

In addition to this, the Gateway Operator can also manage other functions related to a specific Service Platform instance.

- **Service Module** - A Service Module (or just module) is a downloadable, packaged collection of resources and/or code needed to provide a specific function.
- **Service Application** - A Service Application is a set of modules and configuration that collectively implement a specific function or set of functions, possibly across several Service Platforms.

  The Service Application concept is only known to the Service Deployment Manager.
- **Service Provider** - The Service Provider is a business related entity. The Service Provider supplies the necessary means to provide the business related support of a specific Service Application. The Service Provider is also responsible for delegating the task of service deployment to the Service Deployment Manager.
- **Service Deployment Manager** - The Service Deployment Manager acts on behalf of the Service Provider and deploys Service Modules on the Service Platform.

  The Service Deployment Manager manages all aspects related to the life cycle of Service Applications that are external to the Service Platform,
- **Service Customer** - The Service Customer is the entity that is responsible for subscribing to services and paying charges that are incurred using services.
- **Network Provider** - The Network Provider provides and manages wide area network connectivity between the Service Platform and outside parties. Outside parties include the Gateway Operator and other Service Providers. In the case where the Service Platform is connected via the Internet, the Network Provider is assumed to include the Internet Service Provider (ISP) functionality.

4.2 Entities of the EE-enabled Home Gateway

While the last chapter has defined roles and entities of an end-to-end service delivery architecture to deploy software modules, the following definitions are in the scope of the Home Gateway software stack.

- **HG Core (HG_Core)** – The HG_Core is using a dedicated operating system (OS). On this operating system, the HG native software is running, providing all or parts of the home gateway functions as defined in the HGI Residential Profile [1]. Native drivers give access to hardware modules. The summation of the home gateway hardware, operating system, HG native software and drivers is forming the HG_Core.
- **HG Execution Environment (HG_EE)** – The service platform (as defined in 4.1) is running on the HG_Core, and is used for flexible application software module execution and management. The platform allows life cycle control (install, start, update, stop and uninstall) of provider-managed software modules. The summation of the service platform and the software modules forms the **HG_EE**. There may be more than 1 HG_EE running on top of the HG_Core.
- **HG Application Programming Interface (HG_API)** – The HG_API is provided to modules running in the HG Execution Environment (HG_EE), giving standardized access to residential gateway functions that are defined in the HGI Residential Profile. The HG_API can be implemented as part of the HG_Core and as modules running in an HG_EE. The minimal function set of the HG_API must provide access to the parameters defined in TR-098 [2], profiled for the actual HG features. It is assumed that the HG_API is provided by a module (or a set of modules). The actual implementation these modules point to may be found in HG_Core and/or in other modules. Implementation matters of the HG_API are out of scope for this specification.

  Access to the HG_API must be explicitly granted by the operator for each module on a per function basis.
- **Management Agent (MA)** – A piece of software running on the HG and using CWMP [7] that intermediates between a Remote Management System (RMS) and the HG. It handles

requests from the RMS to set and get parameters from the HG or execute HG functions. Moreover, the MA sends notifications from the HG back to the RMS.

This specification does not make any assumptions about the HG-internal implementation of the MA: It could be part of the HG_Core, or one or more modules running in the HG_EE, or a hybrid implementation.

4.3 Home Gateway Software Architecture

This section describes the software architecture for the integration of a Service Platform into an HG. The architecture is generic in a sense that it should apply to any chosen Service Platform technology.

4.3.1 Essentials

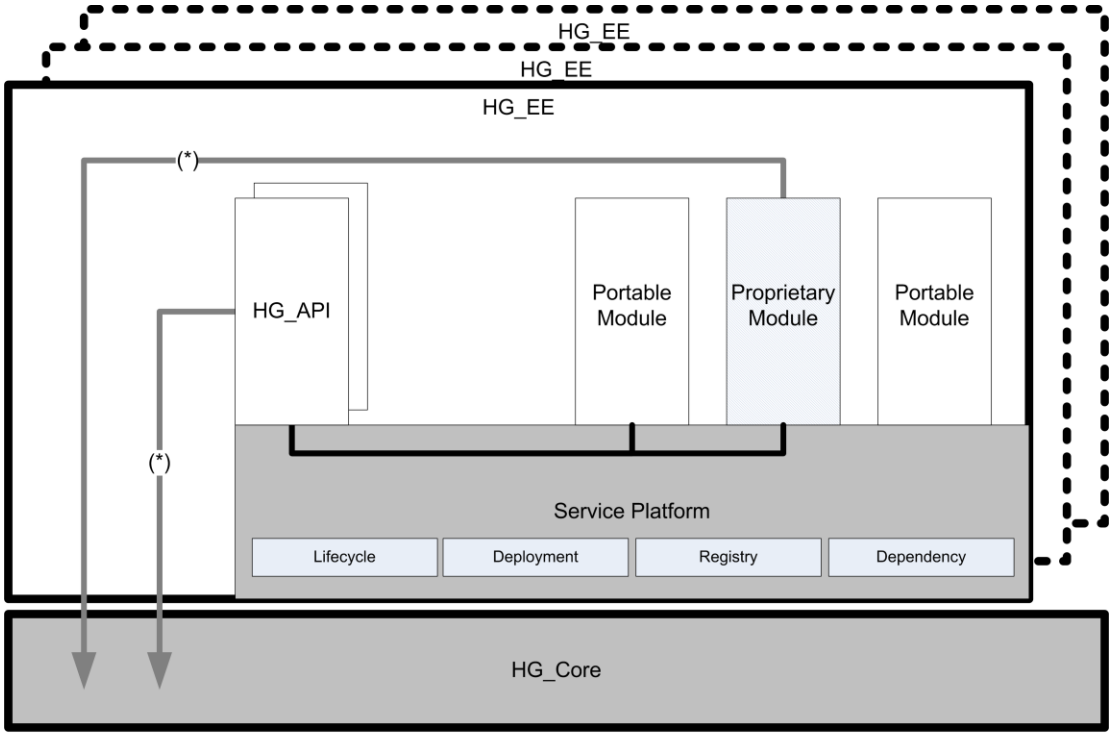
- An HG_API must be defined for any specific Service Platform technology. All HGs using that specific technology must implement the HG_API defined for that technology.
- There are no restrictions on how modules are implemented other than defined by the chosen Service Platform technology.
- Modules that are considered to be portable across different HG products must not access any HG-internal software interface outside of the HG_EE.
- Implementation of the HG_API is vendor-specific and not part of this specification.

4.3.2 Details

The HG_EE runs on top of the HG_Core. Modules run on top of the Service Platform and are able to communicate with each other resp. to use each other's functions, provided that they are authorized to do so.

Using other functions than those provided through mechanisms of the Service Platform is explicitly allowed (e.g. direct access to the HG_Core for vendor-specific modules), although doing so would likely harm the portability of a module.

Fig. 2. Home Gateway Architecture Model



(*) Proprietary

The Service Platform has a number of necessary function blocks:

- **Lifecycle Management** - The Service Platform executes Service Modules and controls their execution. These Service Modules can be started and stopped.
- **Deployment Management** – Modules are packaged in a format defined by the Service Platform technology and can be downloaded from remote repositories accessible through URLs. According to the URL, the proper protocol handler must be available (e.g. HTTP, HTTPS, FTP). Modules can then be installed which may imply unpacking on local storage or other specific one-time tasks.
- **Registry** - Module functions can be exposed through a local repository interface, which is specific to the Service Platform technology. The registry holds up-to-date information on which module is installed, which module is running, etc. Modules can use the registry to lookup functions implemented by other modules.
- **Dependency Management** - A module can depend on other modules to work correctly, e.g. on resources being present or services running. The dependency management function takes care of these relations, resolves dependencies and cleans up when a module is uninstalled.

4.4 Remote Management

Any management of the software configuration of an HG is under control of the Gateway Operator. This includes the approval of any software before being installed onto an execution environment. But even if modules are approved, e.g. by electronically signing them, it becomes too complex to handle software configurations if several parties would install and uninstall modules.

Of course there are operational aspects where an operator wants to allow others to install software, for example when a third party service provider needs the installation of a module to deliver a service. However, these situations must be covered through the operators RMS (e.g.

through a north-bound interface), and not by managing the HG from several points. The RMS then is responsible to figure out any conflicts, lack of resources or other issues.

An execution environment for software modules is just another aspect of an HG to be remotely managed through an RMS. As the HGI has agreed to use CWMP for HG management, the management of an execution environment must also be based on CWMP.

The configuration of service applications can be carried out in any appropriate way, e.g. by a local user interface, a network connection to a third party backend, or through the RMS of the operator. Modules of service applications are not allowed to install or uninstall other software modules.

The RMS has full control and responsibility about which modules to install, uninstall or update. The HG in not able and not in charge to determine the order of install, uninstall and update actions. This definition implies that the RMS must only issue requests to the HG for uniquely identified modules. For example, and RMS shall not issue a single request to the HG to update “all” modules, but must specify each module separately. On the other side, the HG must deny any RMS issued request that cannot be resolved to a well-defined action.

4.4.1 Essentials

- Remote management protocol for HG_EE is CWMP.
- There is only 1 management entity for HG_EE (see role model in 4.1).
- LAN-side management (by e.g. the customer) is not allowed for HG_EE.
- Management/configuration of the service aspect of modules is out of scope for this specification.
- Regular download protocols for modules are HTTP and HTTPS.

4.4.2 Management Agent

- Responsible to translate TR-069 [7]-based RPCs into EE technology specific actions.
- Responsible to translate EE technology specific events into TR-069 based event, including all handling options like passive/active delivery.
- Implementation is vendor-specific. The MA may run in the HG_EE as a module, or in the HG_Core as native code, or as a hybrid set of software components.
- Initial provisioning of MA software or its configuration is out of scope for this specification.

5 Generic Requirements

The following sections show high-level requirements to make an HG execution environment for software modules reliable, manageable and useful to application developers. Any technology used has to meet these requirements.

5.1 HG Requirements

No.	Requirement	
R.1	The HG **MUST** provide an internal system clock.	
R.2	The HG **SHOULD** support the modular installation of USB drivers.	
R.3	The HG **MUST** provide a way to strictly limit the CPU load used by the HG_EE if it would interfere with the HG Core.	
R.4	The HG **MUST** provide a way to strictly limit the runtime memory used by the HG_EE.	

5.2 Reliability and Security

No.	Requirement	
R.5	An HG_EE **MUST NOT** interfere with HG Core functions. Even if an HG_EE collapses, all HG Core functions **MUST** still be able to work.	
R.6	An HG_SP **SHOULD** ensure that any single module can not exceed assigned resources (CPU load, runtime memory).	
R.7	An HG_SP **MUST** ensure that modules do not affect other modules in terms of security and reliability.	
R.8	An HG_SP **MUST** be able to log life cycle operations (install, uninstall, update, start, stop) and resulting errors / aftereffects.	
R.9	An HG_SP **MUST** be able to log any operation not authorized by the gateway operator.	
R.10	An HG_SP **MUST** ensure that only modules authorized by the gateway operator are able to reboot or terminate the HG_EE.	

5.3 Life Cycle Management

Life cycle management refers to the management of modules and their existence on the HG. The life cycle actions for a module are installation, uninstallation, update, start and stop.

5.3.1 General

No.	Requirement	
R.11	An HG_SP **MUST NOT** affect any unrelated service when performing a lifecycle operation. *Note that an unrelated service is defined as a service that doesn't have any dependencies (direct or indirect) to a module.*	

5.3.2 Installation

Installing a module means uploading a module package to the HG from a management backend, or triggering the HG to download a module package from a URL.

No.	Requirement	
R.12	An HG_SP **MUST** be able to install software modules, which are each contained in a single package.	

R.13	An HG_SP **MUST** be able to check the signature integrity of a module. Signature integrity implies the authorization for a module to be installed. If the integrity check fails, the HG_SP **MUST** reject the module before installation.	
R.14	An HG_SP **MUST NOT** start any newly installed module automatically.	

5.3.3 Uninstallation

Uninstalling a module means removing the module from the HG's non-volatile memory.

No.	Requirement	
R.15	Before uninstalling a module, an HG_SP **MUST** inform all depending modules.	
R.16	After uninstalling a software module, an HG_SP **MUST** release all volatile and non-volatile resources of the uninstalled module.	
R.17	Before uninstalling a module an HG_SP **SHOULD** stop the module.	

5.3.4 Update

Updating a module means to replace an existing module on the HG with another module version.

No.	Requirement	
R.18	An HG_SP **MUST** be able to update software modules.	
R.19	If a module was started before its update, the HG_SP **MUST** start the updated module.	
R.20	Before updating a module an HG_SP **SHOULD** stop the module.	

5.3.5 Start

No.	Requirement	
R.21	An HG_SP **MUST** provide a function for starting modules. Starting a module means to assign system resources to a module, invoke an initialisation operation of the module.	
R.22	An HG_SP **MUST** be able to inform dependent modules that another module has been started.	

5.3.6 Stop

No.	Requirement	
R.23	An HG_SP **MUST** provide a function for stopping modules. Stopping a module means to invoke an operation of the module that allows for a graceful release of used resources.	
R.24	After stopping a software module, all assigned dynamic resources of the module **SHOULD** be released by an HG_SP.	
R.25	Before stopping a module, an HG_SP **MUST** be able to inform all depending modules.	

5.4 Remote Management

The HGI operators are already using or intend to use a TR-069 [7]-based remote management system to manage the core functions of a Home Gateway. They would not want to be obliged to use a second management system to manage the lifecycle of software modules on the HG. Thus, software module management should be carried out using TR-069.

No.	Requirement

R.26	Any remote management action on an HG_EE **MUST** be performed atomically. If a remote management action fails, the HG_EE **MUST** rollback to the previous status.	
R.27	An HG_SP **MUST** be able to be managed using CWMP.	
R.28	The HG_EE **MUST** be manageable using the ChangeDUState, `DUStateChangeComplete`, `AutonomousDUStateChangeComplete` RPC methods as defined in TR-069 Amendment 3, and the following restriction applies to the ChangeDUState RPC: ▪ UpdateOpStruct **MUST** always specify the UUID parameter. Any other configuration **MUST** be answered with a FaultStruct message.	
R.29	The HG_EE **MUST** be manageable using the data model component for software modules management as specified in TR-157 Amendment 3, [9]. In particular, the following components **MUST** be supported: ▪ Object `.SoftwareModules.ExecEnv.{i}.` – for remote management of the HG_EE ▪ Objects `.SoftwareModules.DeploymentUnit.{i}.` and `.ManagementServer.DUStateChangeComplPolicy.` - for remote management and monitoring of the installed modular software ▪ Object `.SoftwareModules.ExecutionUnit.{i}.` - for remote management, including starting and stopping, and monitoring of installed applications ▪ Objects `.DeviceInfo.MemoryStatus.`, `.DeviceInfo.ProcessStatus.` - for remote management and monitoring of the whole software environment of the device ▪ Theory of operations for software module management, as defined in Appendix II.	
R.30	For each software module, an HG_SP **MUST** provide a locally unique identifier.	
R.31	An HG_SP **MUST** be able to execute software installation or update requests from an RMS, where software module locations are transmitted as URL.	
R.32	An HG_SP **MUST** support HTTP and HTTPS as software download protocols.	
R.33	An HG_SP **MUST** be able to execute requests from an RMS to uninstall software modules. Requests to uninstall software modules are unconditional.	

5.5 Module Dependencies and Interaction

A module implementation could use functions or data resources of other modules. The HG_SP provides the means to ensure the consistency of a module configuration (no unresolved and stale references).

For the interaction between modules, the HG_SP technology allows modules to register/advertise, retrieve and invoke interfaces.

No.	Requirement	
R.34	An HG_SP **MUST** be able to manage dependencies between modules. Dependencies from one module towards another (resources or interfaces) **SHOULD** include a version number or a version range. If no version is specified, the latest available **MUST** be the default one.	
R.35	An HG_SP **MUST** allow for an interaction of modules through module interfaces.	
R.36	For security reasons, an HG_SP **SHOULD** be able to hide the existence of modules from other modules.	

R.37	An HG_SP **MUST** provide mechanisms for modules to expose their interfaces. Other operations than the advertised ones **MUST** be inaccessible and **SHOULD** be hidden.	
R.38	An HG_SP **MUST** be able to allow and deny access to an interface of a module.	
R.39	An HG_SP **SHOULD** be able to allow and deny access to specific operations of a module interface.	

5.6 Supported Protocol Stacks

Protocol stack support is to avoid that each module bring its own protocol implementation. The requirements do not assume the protocol stacks to be implemented as part of the HG_SP, nor do they specify the type of access provided, since these are technology dependent.

No.	Requirement	
R.40	An HG_SP **MUST** provide a technology-specific standardized API to a UPnP protocol stack for control points and services. The concrete API to use is specified in chapter 6.	
R.41	An HG_SP **MUST** provide technology-specific standardized API to an HTTP and HTTPS client and an HTTP server protocol stack. The concrete API to use is specified in chapter 6.	
R.42	An HG_SP **SHOULD** provide access to an HTTPS server protocol stack.	

5.7 HG Application Programming Interface (HG_API)

No.	Requirement	
R.43	The EE enabled HG **MUST** provide an Application Programming Interface allowing communication between modules and the HG_Core.	
R.44	An HG_API **MUST** provide access to HG functions as specified in the HGI Residential Profile [1].	
R.45	An HG_API **MUST** provide access to management functions of the Internet Gateway Device as specified in TR-098 [2].	
R.46	An HG_API **MUST** provide access to management functions of the Internet Gateway Device as specified in TR-104 [8].	
R.47	If the HG provides SIP functionality, an HG_API **MUST** provide access to the SIP functions as specified in [1] R.260-R.265.	
R.48	An HG_API **SHOULD** provide access to the USB Class Drivers (Printer, Mass Storage, USB Hub, Wireless Controller) functions of the HG.	
R.49	An HG_API **MUST** be able to provide access to file system functions of the HG for internal/external mass storage devices access.	
R.50	An HG_API **SHOULD** provide the means to the HG_EE to get informed of events happening in any implementation of an HG function as specified in the HGI Residential Profile [1], e.g. events related to DHCP.	
R.51	An HG_API **MUST** be able to allow and deny access to specific operations of its interfaces of calling modules.	

6 Technology-specific Requirements

This section specifies details for specific service platform technologies. It currently contains requirements for an OSGi based service platform.

6.1 OSGi Service Platform

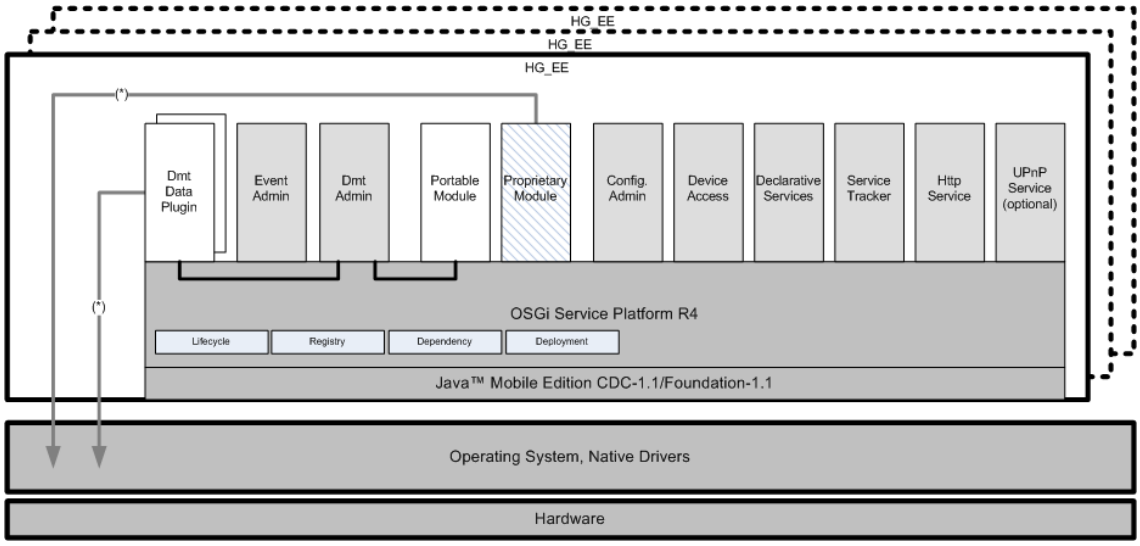
The OSGi Service Platform is a Java based technology that adds a comprehensive component model to Java. Moreover, the OSGi provides specifications of optional services.

This chapter defines a profile of Java and OSGi technology used to implement an HGI compliant execution environment on the HG.

6.1.1 Architecture

The following figure is an instantiation of the generic architecture (Fig. 2) for OSGi technology.

Fig. 3. Component view of an OSGi Service Platform running on the HG



6.1.2 OSGi specific requirements

The following requirements define a Java/OSGi-based execution environment, meeting the generic requirements of chapter 5.

No.	Requirement	
R.52	The HG **MUST** provide a Java Runtime Environment (JRE) following the CDC/Foundation 1.1 [5]. *Note that this requirement does not imply a mandatory certification of the JRE.*	
R.53	The HG JRE **MUST** be able to verify the integrity of bundles signed with certificates issued by an operator-specific list of Certification Authorities.	
R.54	The HG JRE **MUST** be able to verify the integrity of bundles signed by an operator-specific list of self-issued X.509 certificates.	
R.55	The HG JRE class MessageDigest **MUST** support the SHA-1 algorithm to create UUIDs compliant with [7], chapter H.2.	
R.56	The HG **MUST** provide an OSGi Service Platform compliant with [3], including	

	OSGi Security Layer.	

R.57	For unauthorized bundles, the Service Platform and/or the JRE **MUST** deny invocations to operations (in all signature variants) that could compromise the HG_SP. This is a non-exhaustive list.
	
	`System.exit`
	`System.load`
	`System.loadLibrary`
	`System.setSecurityManager`
	`System.setIn`
	`System.setErr`
	`System.setOut`
	`System.clearProperty`
	`System.setProperty`
	`System.setProperties`
	
	`Runtime.getRuntime().halt`
	`Runtime.getRuntime().exec`
	`Runtime.getRuntime().loadLibrary`
	`Runtime.getRuntime().traceInstructions`
	`Runtime.getRuntime().traceMethodCalls`
	`Runtime.getRuntime().addShutdownHook`
	`Runtime.getRuntime().removeShutdownHook`
	
	Any attempt of unauthorized bundles to invoke one the above operations **MUST** result in throwing an exception to the invoking bundle, typically a java.lang.SecurityException.	

R.58	For unauthorized bundles, the Service Platform and/or the JRE **MUST** deny access to classes of the java.reflect package.	
R.59	For unauthorized bundles, the Service Platform and/or the JRE **MUST** deny the installation of native code libraries for the Java Native Interface (JNI). Any attempt to install a bundle that contain native code libraries and is not authorized **MUST** result in throwing an exception to the entity that tried to install the bundle, typically a java.lang.SecurityException.	
R.60	The Service Platform **MUST** include Device Access compliant with [4].	
R.61	The Service Platform **MUST** include Configuration Admin compliant with [4].	
R.62	The Service Platform **MUST** include Declarative Services (also known as Service Component Runtime) compliant with [4].	
R.63	The Service Platform **MUST** include the Tracker (also referred to as Service Tracker) compliant with [4].	
R.64	The Service Platform **MUST** include the DMT Admin Service compliant with [4].	
R.65	The Service Platform **MUST** include the Event Admin compliant with [4].	
R.66	The Service Platform **MUST** include the Log Service compliant with [4].	
R.67	The Service Platform **MUST** include the Http Service compliant with [4].	
R.68	The Service Platform **MUST** include UPnP™ Device Service compliant with [4].	
R.69	The Service Platform **MUST** include the Permission Admin Service compliant with [4].	
R.70	For unauthorized bundles, the Service Platform **MUST** deny invocations to the following Permission Admin Service operations (in all signature variants):	

	`setPermissions()`
	`setDefaultPermissions()`
	
	This is a non-exhaustive list.	

R.71	The Service Platform **MUST** include the Conditional Permission Admin Service compliant with [4].	
R.72	The Service Platform **MUST** include 1 or more data plug-ins for the Dmt Admin Service [4], which **MUST** provide access to the management objects of the HG as specified in TR-098 [2] and TR-104 [8] following these mapping rules for the path of the addressed management entity: ▪ The path starts with ./InternetGatewayDevice. ▪ Any dot ('.') in the TR-069 path is replaced by a slash ('/').	
R.73	The Service Platform **MUST** include 1 or more data plug-ins for the Dmt Admin Service [4], which **MUST** provide access to the management objects of the HG as specified in TR-157 Amendment 3, Appendix II [9] following these mapping rules for the path of the addressed management entity: ▪ The path starts with ./InternetGatewayDevice/SoftwareModules. ▪ Any dot ('.') in the TR-069 path is replaced by a slash ('/').	
R.74	If the HG provides SIP functionality as specified in [1] R.260-R.265, the service platform **MUST** include JSR-180 ("SIP API for J2ME").	

1

7 References

[1] HGI Residential Profile
www.homegateway.org
[2] TR-098 latest amendment, Internet Gateway Device Data Model for TR-069
www.broadband-forum.org
[3] OSGi Service Platform Core Specification, Release 4, Version 4.0 or later,
www.osgi.org
[4] OSGi Service Platform Service Compendium, Release 4, Version 4.0 or later
www.osgi.org
[5] Java ME Foundation Profile
http://java.sun.com/products/foundation/
[6] Internet Gateway Device (IGD) V 1.0
http://www.upnp.org/standardizeddcps/igd.asp
[7] TR-069 Amendment 3, CPE WAN Management Protocol
www.broadband-forum.org
[8] TR-104, DSLHome Provisioning Parameters for VoIP CPE
www.broadband-forum.org
[9] TR-157, Amendment 3 Component Objects for CWMP
www.broadband-forum.org