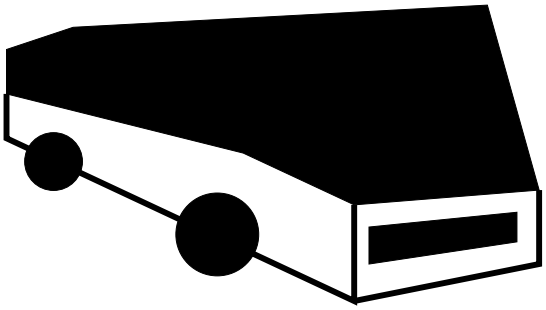
ND-Bus Documentation



Document version 0 (2010-09-08)

Covers ND-Bus version 0

This document is available online as HTML website at http://nd-bus.sourceforge.net and as PDF file at http://nd-bus.sourceforge.net/nd-bus.pdf.

Contents

Chapter 1

ND-Bus

1.1 Table of contents

1.2 What is ND-Bus?

The name *ND-Bus* stands for *Network-enabled Desktop Bus*. ND-Bus is an inter-process communication system which can be used by applications on a single computer and/or within a network to exchange any kind of data, send messages and make remote function calls. Although many kinds of applications may use ND-Bus, it is mainly intended to connect applications with graphical user interfaces on desktop computers and mobile devices like laptops and smartphones. ND-Bus is currently only being developed by a single person but hopefully an international *ND-Bus Community* will soon emerge to realize ND-Bus. The community defines the *ND-Bus Specification* which can be used by everyone who wants to make her own implementations. Furthermore a reference implementation which supports many popular platforms is under development. ND-Bus wants to become the replacement for `D-Bus` which is the defacto standard message bus system for GNU/Linux now, but has no appropriate networking support by design. The overall design of ND-Bus is not only heavily inspired by D-Bus but also includes legacy support for the current

D-Bus protocol. This is first of all because D-Bus is really good but also to simplify the migration from D-Bus to ND-Bus.

1.3 ND-Bus is free software

The ND-Bus Specification is open and freely available for everyone. There are no patent restrictions. All documentation is published under the terms of the GNU Free Documentation License. All sources are published under the terms of the GNU General Public License.

1.4 Features

The list of features is only a first draft. Nothing of this has been realized yet.

- Replacement for D-Bus, but with network support by design
- Legacy support for D-Bus
- Make things like Telepathy less painful for developers
- Transparent signals and slots mechanism across applications on one computer and on a network
- Ability for file transport
- Strong encryption between different machines
- Authentication using certificates
- Optional end-to-end-encryption between applications
- Automatic service discovery
- Service introspection
- Fine grained access rights
- Reference implementation running at least on GNU/Linux, Windows and MacOS
- Later support for other platforms like: FreeBSD, MeeGo, Android, iPhone and Windows phones

1.5 Requirements

The following list states the requirements for running the current reference implementation.

- Platform supported by Qt (see list here), currently only tested on GNU/Linux and Windows
- Installed Qt libraries of version 4.6 or greater

General requirements for fully implementing the current ND-Bus Specification on an arbitrary platform:

- TCP and TLS available

1.6 Development status

Development of ND-Bus has just been started and is currently in the architecture design phase. No code has been released yet.

1.7 FAQ

- Q: Why does ND-Bus beside of running on GNU also aim to run on proprietary platforms like Windows and MacOS instead of concentrating all afforts on supporting only free software?
- A: Firstly ND-Bus is about connecting applications and thus about connecting users. ND-Bus wants to build a bridge between the users of GNU who are free and the people that want to use or for some reasons are forced to use proprietary systems and are thus not free. Secondly if applications on free and proprietary systems are able to talk to each other easily, the barrier to replace existing proprietary systems with free systems is lowered. I think more free software should also run on e.g. Windows as the average user doesn't care what operating system she is using as long as it runs all her applications. If all your applications run on all major platforms you are free to decide if you want to use a free or proprietary platform.
- Q: Why do you start from scratch instead of extending the current D-Bus implementation?
- A: D-Bus is optimized for operating on a single machine. Extending the design to support network transparency would break the implementation in multiple places and there would be a need to introduce incompatible changes to the protocol. The maintainers of D-Bus do not wish to support true network ability anyway and especially don't want to break compatibility. Furthermore D-Bus uses a binary protocol that is platform dependent and thus a problem in mixed environments. The last important point is that D-Bus is written in C and is thus hard to maintain, hard to modularize, error prone and in some ways platform dependent. ND-Bus tries to face these problems by making use of the multiplatform toolkit Qt. Not only does this allow to write ND-Bus in C++, it also abstracts all platform details, resp. operating system details, and so makes it possible to write one source that compiles with no changes on all major platforms, not only Unix-like systems (as D-Bus does) but also on Windows, MacOS and some smartphones. Using Qt makes the development much easier and the produced code becomes stable much faster. Also it is far easier to integrate encryption, network ability and things like regular expressions and state machines. Last but not least: As writing good code with Qt is much easier than with plain C more people with even lower programming skills can actively participate in the development process.
- Q: Don't you think relying on Qt is a drawback? Gnome is based on GTK+ and not on Qt!
- A: I don't like GTK+ and I think Qt is the only serious C++ toolkit available under the GPL today. Qt is the base for KDE, it is the base for MeeGo's GUI and it is used in more and more mobile devices like touchpads, smartphones and soon TVs and car entertainmaint computers. And it runs on Windows and MacOs. If you use Gnome then you'll have to install only a few additional libraries to be able to run Qt based applications.
- Q: But doesn't this mean, that GTK+ applications won't be able to connect to ND-Bus?
- A: No! The core of ND-Bus, the daemon process, will be written in Qt/C++. Applications that want to connect to this daemon can be written in any language and with any toolkit. Of course it will be easier to write applications that connect to ND-Bus when appropriate libraries are available for that language/toolkit. But those 'client' libraries have to be developed for all langauges, even for Qt.
- Q: Telepathy does already everything one can want.

- A: Parts of Telepathy do only exist, because of the missing features in D-Bus. In some places Telepathy abuses D-Bus for things D-Bus was not designed for.
- Q: You're stupid and wasting your time!
- A: We'll see.

1.8 Mailing list

All ND-Bus discussion is currently on nd-bus-interest@lists.sourceforge.net

1.9 Twitter

You can receive news and contact the community on Twitter

1.10 Bugs, patches, feature & support requests

Please use our tracking system: SorgeForge.net trackers

1.11 Developer guidelines

The developer guidelines shall help to keep the community afforts consistent. It is a work in progress, see the current document here: Developer guidelines

1.12 Source code

All sources are being managed with Git.

- Browse the repository online: http://nd-bus.git.sourceforge.net
- Read-only access: git://nd-bus.git.sourceforge.net/gitroot/nd-bus/nd-bus
- Read-write access for developers: ssh://nd-bus.git.sourceforge.net/gitroot/nd-bus/nd-bus

Chapter 2

Developer guidelines

2.1 Table of contents

2.2 Preface

The following paragraphs describe some standard processes, behavior and tools. The coding standard is based on the Kdelibs coding style, the Qt coding style and the GNU coding standards The document is incomplete and everyone has different preferences. If you find something is missing or could be done better then please discuss it on the mailing list.

2.3 Organization

Development of ND-Bus has just been started and the ND-Bus Community currently consists of only one active member - that's me. So all all work is done by - you guessed it right - myself. I hope the community will soon grow. Until then I am the lonely dictator of ND-Bus. As soon as someone else joins the community ND-Bus will get a more democratic form of organization.

2.4 General rules

- Portability comes first. The primary purpose of ND-Bus is to connect applications on very different platforms, especially GNU/Linux, Windows and MacOS.
- Use C++ instead of any other programming language
- Use Qt instead of other toolkits
- When using C++ use Qt
- Avoid arbitrary limits on the length or number of any data structure, including file names, lines, files, and symbols, by allocating all data structures dynamically.
- Check every function call for an error return, unless you know you wish to ignore errors.
- Try to make all functions reentrant and thread safe
- Assume the targeted hardware has multiple CPUs. Try to create parallel algorithms instead of long sequential algorithms. Create multiple threads when it's worth the overall effort.
- A program shall not only write error messages to the terminal but also into a file
- If you need a some functionality and the language/toolkit you are using offers classes etc. to support this: Use them instead of reinventing the wheel.
- Everything has to be put under version control. This applies to source code and documentation. ND-Bus uses Git for this purpose. If you are not already familiar with SCM systems in general and with Git in particular, then please take an evening or evening two and read the Git manual and tutorial.

2.5 Internationalization

- With no exception use American English for every thing in source files and documentation.
- When programming with Qt use its abilities for translation of user interfaces.
- When programming in C use GNU gettext for internationalization if possible.

- It is mandatory to make translations of user interfaces possible, but poviding any actual translation is not a primary task
- Try to integrate unicode support

2.6 Documentation

- First of all: Raw / very short documentation is not good but better than nothing.
- Please don't think 'I can write the documentation for this function later'. As soon as you have a piece of working code and a more or less stable API, then please take the time and write the documentation for it. Otherwise it will become harder and harder to remember all details as days go by and you already have turned your attention to something else.
- I will not integrate any code into ND-Bus that has an incomplete API documentation. (Incomplete documentation of internals is acceptable to some degree).
- Documentation for developers must be supplied in American English
- End user documentation must also be supplied in American English. Translations to other languages are nice to have but not a primary task
- Use `Doxygen` for source code documentation when ever possible. Use Doxygen also for all other kind of documentation. Standard configuration files for Doxygen are provided by the ND-Bus Community.
- ND-Bus provides all documentation at least in HTML and PDF format. PS files and man pages should also be provided when it's not too much work.
- In general, prefer vector graphics to raster graphics.
- It is recommanded to use `xfig` for drawing vector graphics. Save your files always in the FIG-format and export them as PDF files to include them in the documentation.
- It is recommanded to use `GIMP` when you need to create/manipulate raster graphics. Always save your work in the XCF file format and export the image as PNG file to include them in the documentation.
- Don't forget to put your XFIG and XCF files under version control.

2.7 Legal

- All ND-Bus source code must be published under the terms of the current GPL
- All ND-Bus documentation must be published under the terms of the current GFDL
- All files must state an appropriet legal notice

2.8 Identation

- 4 Spaces, no tabs

2.9 General naming conventions

- Names start with a lowercase letter.
- Each new word in a name starts with a capital letter
- Avoid short names and abbreviations like 'randGetVal' or 'snd'
- Exceptions: The meaning of a short name is absolutely obvious or the name is e.g. a counter variable or similar temporary object
- If an abbreviation is part of a name then only the first letter is upper case, eg. getIpAddress and ipIsValid instead of getIPAddress and IPIsValid
- Exception: 'NDBus' must be written like this always.

2.10 Variables

- Each variable declaration on a new line
- Wait with declaring a variable until it is needed

2.11 Functions

- Forward declarations are mendatory
- Try to avoid pointers and use references
- In the parameter list the input-parameters come first, followed by combined input/output-parameters and output-parameters at last
- Try to avoid combined input/output-parameters.
- As a rule of thumb, a function should either return void if it can't fail anyhow, return bool if can fail and no further error handling is needed (false means error) or return int (qint32) where zero means success and anything else is a well documented error code.
- Avoid returning anything else, e.g. strings, without good reason.
- Exception: Class member function that simply return class private data etc.

2.12 Classes

- General naming conventions also apply to class declarations, but with the exception that class names start with a capital letter

2.13 Templates

- Use them

2.14 C++ Exceptions

- Don't use them

2.15 Whitespace

- Use blank lines to group statements
- Use only one empty line
- Use one space after each keyword
- For pointers or references, use a single space before '$*$' or '&', but not after
- No space after a cast

2.16 Types

- Use Qt types like qint32 instead of int where possible
- Avoid C-style casts when possible, use c++ casts
- Use QStrings instead of c-style strings
- Use Qt container classes instead of STL container classes
- Use STL-style iterators for Qt containers, not Java-style

2.17 Braces

- The left curly brace goes on the same line as the start of the statement.
- Use curly braces even when the body of a conditional statement contains only one line.
- Use curly braces when the body of a conditional statement is empty
- Exception: Function implementations, class, struct and namespace declarations always have the opening brace on the start of a line.

2.18 Switch statements

- Case labels are on the same column as the switch
- The default label is mentatory
- Every case must have a break (or return) statement at the end or a comment to indicate that there's intentionally no break

2.19 Parentheses

- Use parentheses to group expressions
- Avoid assignments inside conditions

2.20 Line breaks

- Try to keep lines shorter than 100 characters, inserting line breaks as necessary.
- Commas go at the end of a broken line; operators start at the beginning of the new line.

2.21 Qt Includes

- If you add includes for Qt classes, use both the module and class name. This allows library code to be used by applications without excessive compiler include paths.

2.22 Preprocessor

- General rule: Avoid the use of the c preprocessor when ever possible
- Use enum types and constant types instead of defines
- Try to use if() statements with constant types instead of ifdef with defines when conditional compilation is needed; modern compilers will create exactly the them output but can perform more extensive type checking before.
- Exception: ifndef statements for header file protection are ok.
- Avoid function-like macros

Chapter 3

GNU Free Documentation License

GNU Free Documentation License Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the

copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission. B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement. C. State on the Title page the name of the publisher of the Modified Version, as the publisher. D. Preserve all the copyright notices of the Document. E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices. F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below. G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice. H. Include an unaltered copy of this License. I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence. J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission. K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein. L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles. M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version. N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section. O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first

published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Chapter 4

Overall design

4.1 Glossary

The following terms are defined within the ND-Bus specification:

Domain Abstraction for operating system context.

User Abstraction for what a *user* is in Unix.

Service group Abstraction for what *session bus*, *system bus* and similar are in D-Bus.

Service Abstraction for what a *process* is in Unix.

Instance identifiers Each service group has a name that specifies what this group is good for. There might be multiple service groups for the same purpose, so they both have the same name. These instances can be identified/differentiated by their *instance identifers*. It's the same with services. For example if a user runs 3 gnome sessions in parallel, they all can have the service group name "gnome_session", but their instance identifiers are 0,1 and 2.

Instance contingent The number of instances of a particular service group can be restricted. The total amount is called *instance contingent*. When the maximum number of registered service group instances has been reached we say *the instance contingent is exhausted*. The same applies to services.

Service group master service The service that created a service group.

Service identification Uniquely identifies a service by the set of its corresponding domain, user, service group and service.

Node tree Structured view on the nodes of a service.

Node Abstract container for interfaces and nodes.

Interface Abstraction for what objects are in object oriented programming.

Method Function within an interface. Functions have input and output parameters. Methods are *called* by remote services and *executed* by its interface.

Signal Special function within an interface. Signals have output parameters only. Signals are *emitted* by its containing interface. Remote services can register to a particular signal to *receive* it.

4.2 Hierarchy

The term *service* is of central relevance within the ND-Bus terminology. A service is something that does something, provides some functionality to something else and may make use of the functionality of something else.

ND-Bus is about connecting services.

Todays operating systems encapsulate certain functionality within applications. The term service is more abstract. ND-Bus does not know about applications. So it depends on the implementation what part of computer software offers a ND-Bus service. Normally it is an application that will provide exactely one service, but of course is also possible that an application or that something else, e.g. a simple microcontroller firmware without an OS or the kernel of an OS, offers one or more services.

In ND-Bus services are hierarchically organized. On top is a mesh of *domains*. Domain is an abstraction for an organizational unit. With nowadays computers a domain normally includes exactely one operating system instance. Within a domain the next organizational unit are the *users*. Users can be anything, normally they are the people who use the OS (Unix/Windows user). On the next level there are the *servicegroups* that bundle services on arbitrary criterions. E.g. all services that belong to a particular desktop session are grouped together or all services that belong to the OS kernel. The next level are the services themselves. So: every service belongs to exactely one domain, exectely one user and exectely one servicegroup. If multiple servicegroups bundle the same kind of services those servicegroups can have same name, but they can still be differentiated by their unique *instance identifiers*. This is analogous with service that provide the same functionality.

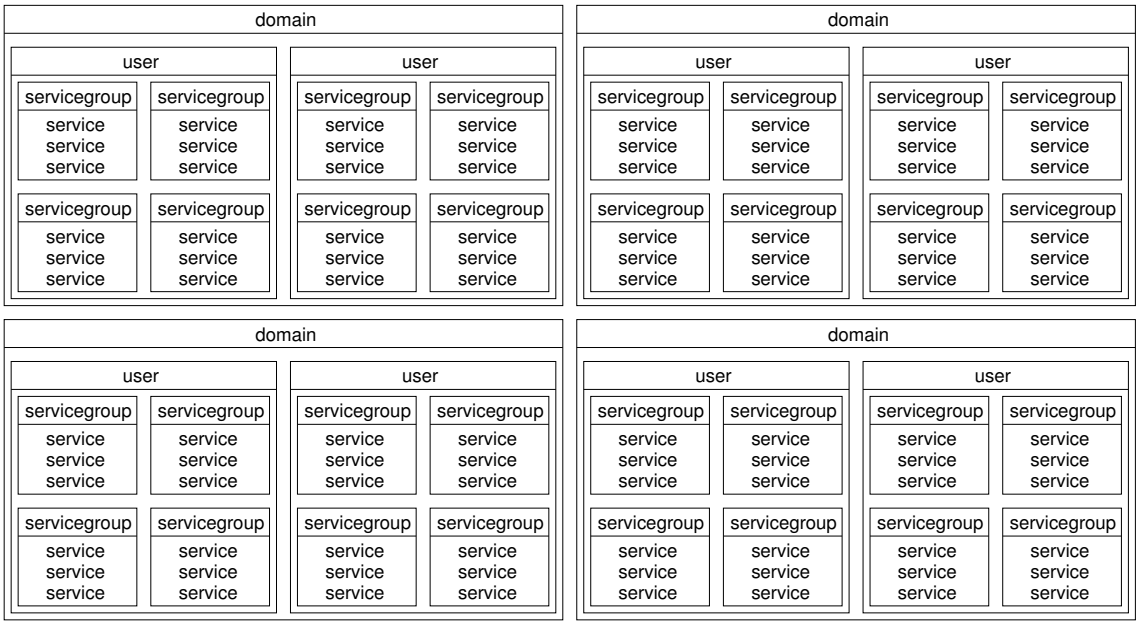


Figure 4.1: Hierarchy (loose notation)

Every service within ND-Bus can play exactely one *role*. An ordinary service (e.g. mediaplayer, terminal emulator, webbrowser) has the role *customer*. But within every domain there must be exectely one service in the role of the *agent*. This one mainly acts as a broker to help the customers to connect to each other. One can see if a service is agent or customer by its name: An agent is always service "org.ndbus.agent" (with instance identifier 0) within user "ndbus" and servicegroup "ndbus" (with instance identifier 0).

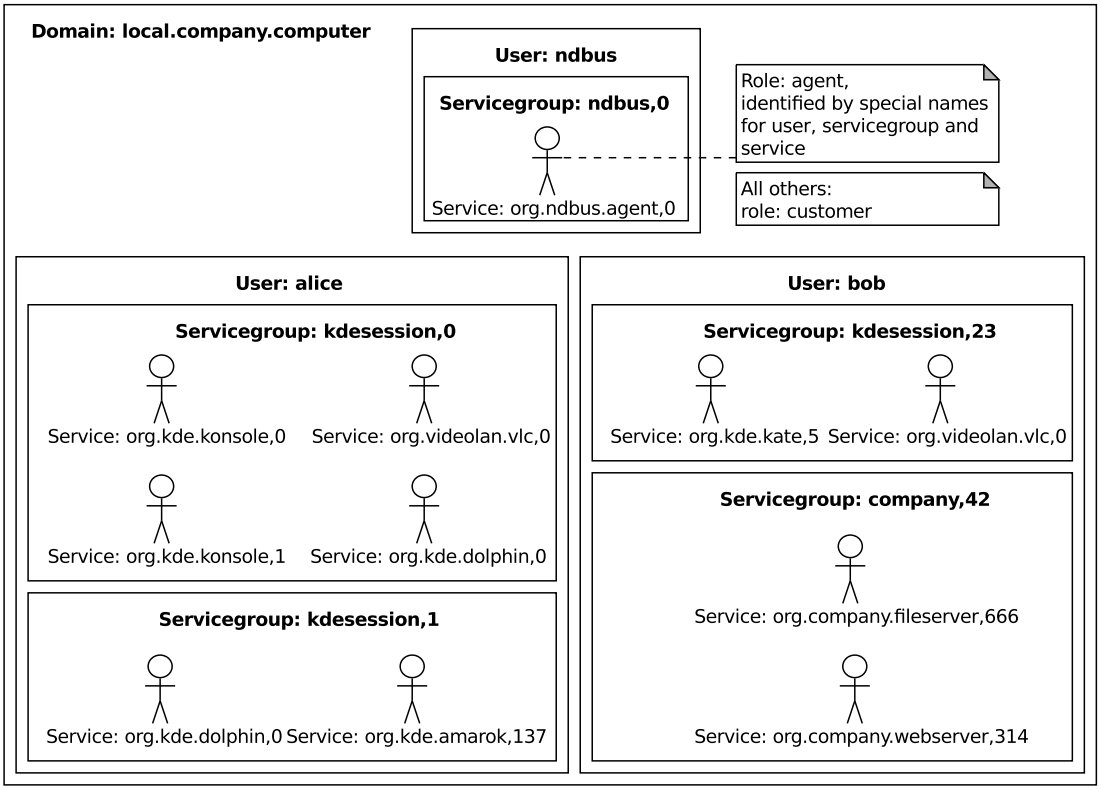


Figure 4.2: Example domain (loose notation)

Like already stated, service provide functionality. This includes *methods*, *signals* and data represented as *files*. All these are again hierarchically organized within a service. This hierarchy is a tree. Within the tree there are containers called *nodes*. A node can again contain nodes and other things: files and *interfaces*. An interface is an abstraction similar to objects in object-oriented programming languages. Interfaces can contain methods and signals.

4.3 Valid names

The various names in ND-Bus have some restrictions:

- All names have a maximum length of 255 characters
- All names are case sensitive
- Names for domain, user, servicegroup, service, file and interface must exactely match: `([A-Za-z_]+[A-Za-z0-9_]*)(\.([A-Za-z_]+[A-Za-z0-9_]*))*`
- Names for nodes, methods and signals must exactely match: `[A-Za-z_]+[A-Za-z0-9_]*`
- Exception: The root node of the tree has the fixed name "/".
- Parameter names of methods and signals must exactely match: `[A-Za-z_]+[A-Za-z0-9_]*`

4.4 Types

In various places the ND-Bus specification refers to particular data types, e.g. in the context of message formats. Especially can methods and signals only use a limited set of specific types as parameters. It is not possible to use other types than listed below, but with `List<Variant>` and `Map<String, Variant>` one can construct arbitrarily complex data structures of the given types.

Conventinal name	Description

4.5 Entities

All the things named before, means: domains, users, servicegroups, services, nodes, files, interfaces, methods and signals, are generally refered to as "entities" to make it easier to talk about them.

An entity is described by a combination of a *classifier*, *name* and *parameters*. The combination is the *signature* of the entity. The signature is a `STRING` formated as: `<classifer>;<name>(;<parameter>)*`. The number and format of the parameters and depend on the classifier. Every kind of entity has a unique classifier.

4.6 Valid classifiers

Classifiers are of type `UINT8` and must be given in their decimal representations with no leading zeros.

Conventional name	Decimal value
DOMAIN	0
USER	1
SERVICEGROUP	2
SERVICE	3
NODE	4
FILE	5
INTERFACE	6
METHOD	7
SIGNAL	8

4.6.1 Valid entity parameters

As stated before, entity signatures may include zero or more parameters. The number of parameters is fixed for a given classifier.

Domain No parameters.

User No parameters.

Servicegroup Has its instance identifer as parameter. It is of type `UINT32` and must be given in its decimal representation with no leading zeros. In some context the the parameter may be left blank (empty string).

Service Has its instance identifer as parameter. It is of type `UINT32` and must be given in its decimal representation with no leading zeros. In some context the the parameter may be left blank (empty string).

Node Has its path as parameter. Thus parameter must be a valid path, means left blank (empty string) when node is "/" or otherwise exactely match: `/|(/<name>)+`, where `<name>` is a valid node name, but not "/".

File No parameters.

Interface No parameters.

Method Has 2 parameters:

- Parameter 0: List of the input parameters of the method
- Parameter 1: List of the output parameters of the method

Both parameters use the same format: `(<type>=<param>)+(,<type>=<param>)*`, where `<type>` is a valid type name and `<param>` is a valid method parameter name. A parameter can be left blank (empty string) to indicate an empty parameter list.

Signal Has as parameter the list of the output parameters of the signal. The parameters uses the format: `(<type>=<param>)+(,<type>=<param>)*`, where `<type>` is a valid type name and `<param>` is a valid method parameter name. The parameter can be left blank (empty string) to indicate an empty parameter list.

4.6.2 Entity identifiers

The combination of domain, user, servicegroup and service uniquely identifies a service within ND-Bus. Nodes, files, interfaces, methods and signals are parts of a particular service and thus are organized in a tree structure. A particular entitiy can be refered to either by its *complete identifier* or by its *short identifier* which omits domain, user, servicegroup and service. Which identifier can be used depends on the context. Such an identifier is a sequence of ":"-separeted signatures defining the position of an entity within a ND-Bus system, starting from the domain.

Examples for complete identifiers:

Node: `0;local.alice:1;peter:2;kdesession;0:3;org.kde.amarok,0:4;control;/app/main`

Root node: `0;local.alice:1;peter:2;kdesession;0:3;org.kde.amarok,0:4;/;`

Method: `0;local.bob:1;peter:2;kde;0:3;org.app,0:4;/;:6;org.nice:7;doIt;;`

Service: `0;local.bob:1;peter:2;terminal;0:3;cc.funny.tool;1337`

Domain: `0;org.ndbus`

Same examples for short identifiers:

Node: `4;control;/app/main`

Root node: `4;/;`

Method: `7;doIt;;`

Service: not applicable

Domain: not applicable

4.7 Transport

Services talk to each other via sockets. The ND-Bus specification describes 3 different kinds of sockets that services may use as transport mechanism:

Local sockets Can only be used for local connections. Implemented on top of Unix domain sockets. Fast and secure, recommanded for local connections. Exeption: On Windows it's fast, too, but insecure, because there it is implemented on top of Named Pipes. If you don't trust your Windows installation (and programs and users) then don't use local sockets.

TCP sockets On systems that don't support local sockets one can use TCP sockets for the local connections. It is also possible to use TCP sockets for the remote connections. TCP sockets are slower than local sockets and insecure. So in general it is not recommanded to use them without reasons.

TLS sockets This is recommanded for remote connections. It is much slower than TCP sockets but secure.

4.8 Message format

All ND-Bus communication is point-to-point so there is no need for addressing information within messages. The message is the smallest logical entity within the protocol. Every message belongs to a single transmission. Within each transmission each message has a unique *message sequence number (MSN)*. Within each connection each transmission has a unique *transmission sequence number (TSN)*. The *message size (MS)* indicated the number of bytes of the whole encoded message (including the size information itself). MS, TSN, MSN and MT form the *message header (MH)*. All contents of a message must be encoded in the *ND-Bus DataStream format*.

Field index	Type	Description
0	`UINT32`	Message size
1	`UINT32`	Transmission sequence number
2	`UINT32`	Message sequence number
3	`UINT32`	Message type

The MH may be followed by the payload of the message. If a message contains any payload and how it is formatted depends on the *message type (MT)*:

Conventional name	Decimal value	Description
`ERROR`	0	Error reply
`METHOD`	1	Method call
`METHOD_RETURN`	2	Method reply with returned data
`FILE`	3	File request
`FILE_PART`	4	Reply to file request with part of file
`FILE_ACK`	5	Reception of file part acknowleged (if part left: ready for next)
`AUTH`	6	Authentication request for previously requested action
`AUTH_RETURN`	7	Authentication reply
`SIGNAL_SUBSCRIBE`	8	Signal subscription request
`SIGNAL_ACK`	9	Signal subscription reply
`SIGNAL`	10	Signal emission

4.8.1 Format of ERROR message payload

Field index	Type	Description
0	`UINT32`	Error code (depends on context)
1	`STRING`	Error message (depends on context)

4.8.2 Format of METHOD message payload

Field index	Type	Description
0	STRING	Node
1	STRING	Interface
2	STRING	Method
3	VARIANTLIST	Input

4.8.3 Format of METHOD_RETURN message payload

Field index	Type	Description
0	VARIANTLIST	Output

4.8.4 Format of FILE message payload

Field index	Type	Description
0	STRING	Node
1	STRING	File

4.8.5 Format of FILE_PART message payload

Field index	Type	Description
0	UINT32	Parts left (last part: 0)
1	BYTEARRAY	Part of file

4.8.6 Format of FILE_ACK message payload

This message type contains no payload.

4.8.7 Format of AUTH message payload

Field index	Type	Description
0	BYTEARRAY	Chunck as challenge for authentication procedure

4.8.8 Format of AUTH_RETURN message payload

Field index	Type	Description
0	BYTEARRAY	Returned chunck of authentication procedure

4.8.9 Format of SIGNAL_SUBSCRIBE message payload

Field index	Type	Description
0	STRING	Node
1	STRING	Interface
2	STRING	Signal

4.8.10 Format of SIGNAL_ACK message payload

This message type contains no payload.

4.8.11 Format of SIGNAL message payload

Field index	Type	Description
0	`STRING`	Node
1	`STRING`	Interface
2	`STRING`	Signal
3	`VARIANTLIST`	Output

4.9 Connecting daemon to daemon

Daemon A of local.alice wants to connect to daemon B of local.bob.

- A looks up the information to connect to B
- A establishes network connection to B
- A calls `/:org.ndbus:hello` on B
- A calls `/:org.ndbus:authenticate` on B
- A calls `/:org.ndbus.daemon.daemons:connectDomains` on B
- A and B are now connected.

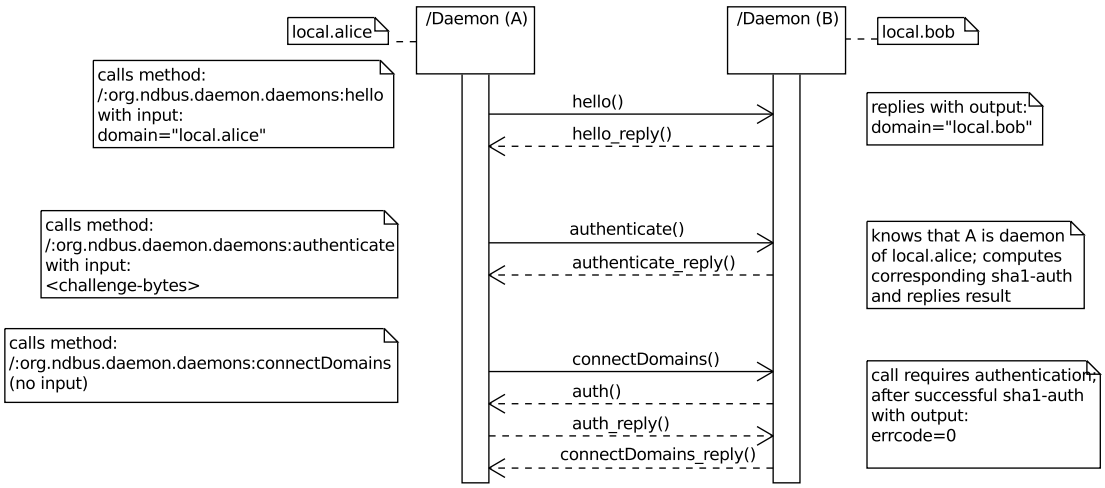


Figure 4.3: Connecting daemon to daemon

4.9.1 Connecting service to domain

Process P wants to connect to daemon A of local.alice A and become a service.

- P looks up the information to connect to A
- P establishes network connection to A
- P calls `/:org.ndbus:hello` on A
- P calls `/:org.ndbus:authenticate` on A
- P calls `/:org.ndbus.daemon.services:joinDomain` on A
- P calls `/:org.ndbus.daemon.services:joinUser` on A
- P calls `/:org.ndbus.daemon.services:joinServiceGroup` on A
- P calls `/:org.ndbus.daemon.services:addService` on A
- P and A are now connected, P is service in desired servicegroup. P can now go on and adds its nodes, interfaces etc.

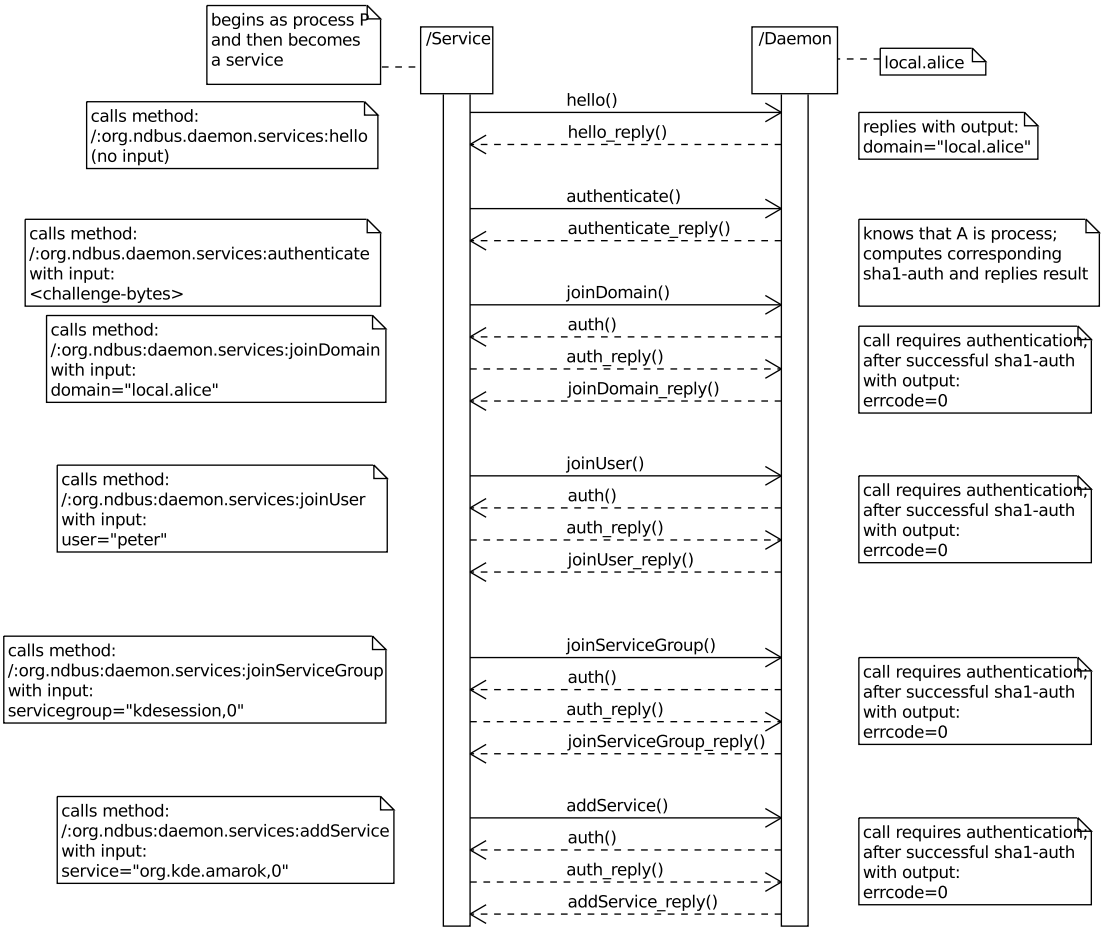


Figure 4.4: Connecting service to daemon

4.9.2 Connecting service to domain

Service A at daemon DA on local.alice wants to connect to service B at daemon DB on local.bob.

- A calls `/:org.ndbus.daemon.services:broker` on DA (and thus asks for e.g. domain="local.bob", user="system",servicegroup="system",service="org.kernel.acpi")
- DA looks up A, checks its policy and may deny the request
- DA calls `/:org.ndbus.daemon.daemons:broker` on DB
- B checks its policy and may deny the request
- DB return session_cookie(X,Y) S to DA
- DA returns S to A
- A establishes network connection to B
- A calls `/:org.ndbus:hello` on B (and thus identifies itself e.g. as domain="local.alice", user="barne",servicegroup="kdesession,0",service="cc.chubbyblackcat.funnyremotetool")
- A calls `/:org.ndbus.authenticate` on B
- B calls `/:org.ndbus.daemon.services:broker` on DB
- DB returns S to B
- B uses S to authenticate itself against A
- A calls `/:org.ndbus.service.services:connectServices` on B (as this commands needs authentication, on success A and B now both are authenticated against each other (using S).
- X and Y are now connected

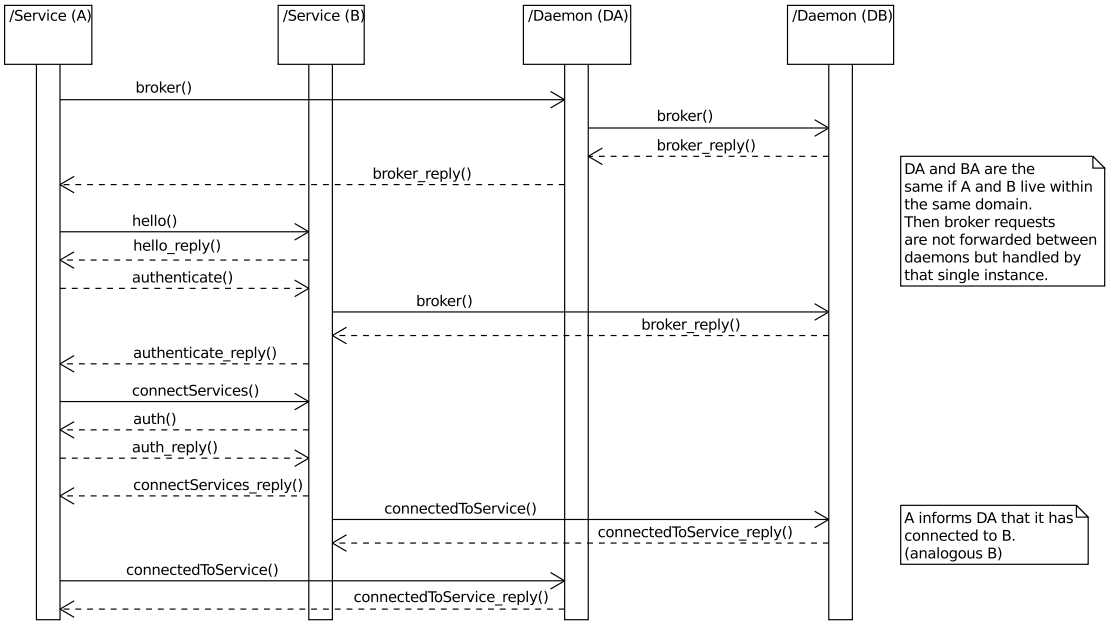


Figure 4.5: Connecting service to service

Chapter 5

Standard interfaces

5.1 org.ndbus.introspector

5.1.1 Method: identify

Direction	Field index	Type	Name
out	0	STRING	service

5.1.2 Method: nodeNames

Direction	Field index	Type	Description
in	0	STRING	node
out	0	STRINGLIST	nodeNames

5.1.3 Method: files

Direction	Field index	Type	Description
in	0	STRING	node
out	0	STRINGLIST	files
out	1	INT32	errcode

5.1.4 Method: interfaces

Direction	Field index	Type	Description
in	0	STRING	node
out	0	STRINGLIST	interfaces
out	1	INT32	errcode

5.1.5 Method: methods

Direction	Field index	Type	Description
in	0	STRING	node
in	1	STRING	interface
out	0	STRINGLIST	methods
out	1	INT32	errcode

5.1.6 Method: signals

Direction	Field index	Type	Description
in	0	STRING	node
in	1	STRING	interface
out	0	STRINGLIST	signals
out	1	INT32	errcode

5.1.7 Signal: nodeAdded

Field index	Type	Description
0	STRING	parentNode
1	STRING	node

5.1.8 Signal: nodeDeleted

Field index	Type	Description
0	STRING	parentNode
1	STRING	node

5.1.9 Signal: fileAdded

Field index	Type	Description
0	STRING	parentNode
1	STRING	file

5.1.10 Signal: fileDeleted

Field index	Type	Description
0	STRING	parentNode
1	STRING	file

5.1.11 Signal: fileChanged

Field index	Type	Description
0	STRING	parentNode
1	STRING	file
2	STRING	parentNodeNew
3	STRING	fileNew

5.1.12 Signal: interfaceAdded

Field index	Type	Description
0	STRING	parentNode
1	STRING	interface

5.1.13 Signal: interfaceDeleted

Field index	Type	Description
0	STRING	parentNode
1	STRING	interface

5.1.14 Signal: methodAdded

Field index	Type	Description
0	STRING	parentNode
1	STRING	interface
2	STRING	method

5.1.15 Signal: methodDeleted

Field index	Type	Description
0	STRING	parentNode
1	STRING	interface
2	STRING	method

5.1.16 Signal: signalAdded

Field index	Type	Description
0	STRING	parentNode
1	STRING	interface
2	STRING	signal

5.1.17 Signal: signalDeleted

Field index	Type	Description
0	STRING	parentNode
1	STRING	interface
2	STRING	signal

5.2 org.ndbus.daemon.introspector

5.2.1 Method: domains

Direction	Field index	Type	Description
out	0	STRINGLIST	domains
out	1	INT32	errcode

5.2.2 Method: users

Direction	Field index	Type	Description
in	0	STRING	domain
out	0	STRINGLIST	users
out	1	INT32	errcode

5.2.3 Method: serviceGroups

Direction	Field index	Type	Description
in	0	STRING	domain
in	1	STRING	user
out	0	STRINGLIST	servicegroups
out	1	INT32	errcode

5.2.4 Method: services

Direction	Field index	Type	Description
in	0	STRING	domain
in	1	STRING	user
in	2	STRING	servicegroup
out	0	STRINGLIST	services
out	1	INT32	errcode

5.2.5 Method: nodes

Direction	Field index	Type	Description
in	0	STRING	domain
in	1	STRING	user
in	2	STRING	servicegroup
in	3	STRING	service
in	4	STRING	node
out	0	STRINGLIST	nodes
out	1	INT32	errcode

5.2.6 Method: files

Direction	Field index	Type	Description
in	0	STRING	domain
in	1	STRING	user
in	2	STRING	servicegroup
in	3	STRING	service
in	4	STRING	node
out	0	STRINGLIST	files
out	1	INT32	errcode

5.2.7 Method: interfaces

Direction	Field index	Type	Description
in	0	STRING	domain
in	1	STRING	user
in	2	STRING	servicegroup
in	3	STRING	service
in	4	STRING	node
out	0	STRINGLIST	interfaces
out	1	INT32	errcode

5.2.8 Method: methods

Direction	Field index	Type	Description
in	0	STRING	domain
in	1	STRING	user
in	2	STRING	servicegroup
in	3	STRING	service
in	4	STRING	node
in	5	STRING	interface
out	0	STRINGLIST	methods
out	1	INT32	errcode

5.2.9 Method: signals

Direction	Field index	Type	Description
in	0	STRING	domain
in	1	STRING	user
in	2	STRING	servicegroup
in	3	STRING	service
in	4	STRING	node
in	5	STRING	interface
out	0	STRINGLIST	signals
out	1	INT32	errcode

5.2.10 Signal: domainAdded

Field index	Type	Description
0	STRING	domain

5.2.11 Signal: domainDeleted

Field index	Type	Description
0	STRING	domain

5.2.12 Signal: userAdded

Field index	Type	Description
0	STRING	domain
1	STRING	user

5.2.13 Signal: userDeleted

Field index	Type	Description
0	STRING	domain
1	STRING	user

5.2.14 Signal: serviceGroupAdded

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup

5.2.15 Signal: serviceGroupDeleted

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup

5.2.16 Signal: serviceAdded

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service

5.2.17 Signal: serviceDeleted

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service

5.2.18 Signal: nodeAdded

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	parentnode
5	STRING	node

5.2.19 Signal: nodeDeleted

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	parentnode
5	STRING	node

5.2.20 Signal: fileAdded

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	file

5.2.21 Signal: fileDeleted

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	file

5.2.22 Signal: fileChanged

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	file
6	STRING	nodeNew
7	STRING	fileNew

5.2.23 Signal: interfaceAdded

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	interface

5.2.24 Signal: interfaceDeleted

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	interface

5.2.25 Signal: methodAdded

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	interface
6	STRING	method

5.2.26 Signal: methodDeleted

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	interface
6	STRING	method

5.2.27 Signal: signalAdded

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	interface
6	STRING	signal

5.2.28 Signal: signalDeleted

Field index	Type	Description
0	STRING	domain
1	STRING	user
2	STRING	servicegroup
3	STRING	service
4	STRING	node
5	STRING	interface
6	STRING	signal

5.3 org.ndbus.service.daemons

5.3.1 Method: disconnect

This method has no parameters.

5.3.2 Method: disconnectFromService

Direction	Field index	Type	Description
in	0	STRING	domain
in	1	STRING	user
in	2	STRING	servicegroup
in	3	STRING	service
out	0	INT32	errcode

5.3.3 Method: connectedServices

out	0	STRINGLIST	services
out	1	INT32	errcode

5.3.4 Signal: serviceConnected

0	STRING	service

5.3.5 Signal: serviceDisconnected

0	STRING	service

/org.ndbus.daemon.daemons method hello i: string domain, string user, string servicegroup, string service o: string domain, string user, string servicegroup, string service, int32 errcode method authenticate i: bytearray challenge o: bytearray replay, int32 errcode method connect i: o: int32 errcode method connectionRequest i: string user, string servicegroup, string service, string user, string servicegroup, string service o: bytearray sessionCookie, int32 errcode

/org.ndbus.daemon.services method hello i: string domain, string user, string servicegroup, string service o: string domain, string user, string servicegroup, string service, int32 errcode method authenticate i: bytearray challenge o: bytearray replay, int32 errcode method connect i: o: int32 errcode method joinDomain i: string domain, bytearray password o: int32 errcode method joinUser i: string user, bytearray password o: int32 errcode method joinServiceGroupAddService i: string servicegroup, bool addIfDoesNotExist, bytearray servicegroup_password, string service, bytearray service_password o: string servicegroup, string service, int32 errcode method addServiceGroupAddService i: string servicegroup, bool joinIfDoesExist, bytearray servicegroup_password, string service, bytearray service_password o: string servicegroup, string service, int32 errcode method addService i: string service, bytearray password o: string service, int32 errcode method addNode i: string node, string newNode o: int32 errcode method addInterface i: string node, string interfaces o: int32 errcode method addMethod i: string node, string interface, string name, string parameters o: int32 errcode method addSignal i: string node, string interface, string name, string parameters o: int32 errcode method connectionRequest i: string domain, string user, string servicegroup, string service o: bytearray sessionCookie, int32 errcode

/org.ndbus.service.services method hello i: string domain, string user, string servicegroup, string service o: string domain, string user, string servicegroup, string service, int32 errcode method authenticate i: bytearray challenge o: bytearray replay, int32 errcode method connect i: string domain, bytearray password o: int32 errcode