Problems

- \( \lim_{t \to \infty} \frac{\text{packagers}(t)}{\text{packages}(t)} = 0 \)

- Sometimes we need unpackaged software even in largest distributions

- Packaging is a hard process

- Too many diverse and incompatible packaging systems, it’s hard to choose the distribution for entire life

- How many *BSD/Debian/Gentoo/Fedora/Arch Linux developers are in this room? ;-) 

- What is the reason to package the same software again and again for hundreds of Linux distributions, NetBSD, Solaris, and others after all?
Possible solutions

- Open Build Service (OBS). Unfortunately it is Linux-specific.
- ABF from ROSA. The same problem as above.
- Korinf. Does AltLinux/Etersoft care about non-Linux portability?
Cross-platform packaging system. PkgSrc!
PkgSrc disadvantage (minor)

- PkgSrc is a parallel world. It has its own package format and binary packages managers, therefore pkgdb.
Package format. Package managers. Build specifications.

<table>
<thead>
<tr>
<th>format</th>
<th>build spec</th>
<th>low-level mngr</th>
<th>high-level mngr</th>
</tr>
</thead>
<tbody>
<tr>
<td>.rpm</td>
<td>RPM spec</td>
<td>rpm(8)</td>
<td>yum, zypper, apt</td>
</tr>
<tr>
<td>.deb</td>
<td>debian/rules</td>
<td>dpkg(8)</td>
<td>apt, aptitude</td>
</tr>
<tr>
<td>.tgz (NetBSD)</td>
<td>pkgsrc Makefiles</td>
<td>pkg_*(8)</td>
<td>nih, pkgin</td>
</tr>
</tbody>
</table>

What if we separate package build mechanism from package format and package management?
pkgsrc4unix components

- Build mechanism: pkgsrc (including bulk builds)
- Package format: native
- Low-level and high-level package management: native (rpm/yum, dpkg/apt etc.)
- Package convertor: EPM (http://epmhome.org)
pkgsrc4unix features

- PkgSrc infrastructure is used for package build
- PkgSrc tools are used for bulk builds
- `epm(1): .tgz -> .rpm`
- `createrepo(8): .rpm -> yum repository`
- `pkgsrc` way is used everywhere: handling configuration files, registering info files, fonts, daemon startup scripts etc.
- `pkgsrc4unix` doesn’t conflict with existing repositories (repoforge, epel, elrepo etc.) because every package has a prefix "nb-" in its name and all files are installed to "/opt/pkgsrc4unix".
System: RHEL-6, .rpm, rpm(8), yum(8)

```
# cat /etc/redhat-release
Scientific Linux release 6.4 (Carbon)
# cd /etc/yum.repos.d/
# wget http://pkgsrc4unix.mova.org/packages/
  RHEL/6/x86_64/pkgsrc4unix.repo
# yum install nb-mk-configure
...
```

<table>
<thead>
<tr>
<th>Package</th>
<th>Arch</th>
<th>Version</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installing:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>nb-mk-configure</td>
<td>x86_64</td>
<td>0.24.0-1</td>
<td>pkgsrc4unix</td>
</tr>
<tr>
<td>Installing for dependencies:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>nb-bmake</td>
<td>x86_64</td>
<td>20110606-1</td>
<td>pkgsrc4unix</td>
</tr>
<tr>
<td>nb-bootstrap-mk-files</td>
<td>x86_64</td>
<td>20120415-0</td>
<td>pkgsrc4unix</td>
</tr>
</tbody>
</table>

... Complete!
```
System: RHEL-6, .rpm, rpm(8), yum(8)

```
# rpm -qa | grep ^nb-
  nb-bmake-20110606-1.x86_64
  nb-bootstrap-mk-files-20120415-0.x86_64
  nb-mk-configure-0.24.0-1.x86_64

# rpm -qi nb-mk-configure
Name                  : nb-mk-configure
Version               : 0.24.0
Release               : 1
Vendor                : pkgsrc4unix
...                   
Packager              : Aleksey Cheusov <cheusov@NetBSD.org>
Summary               : Lightweight but powerful replacement
                        for GNU autotools
Description:
mk-configure is a lightweight replacement for GNU autoconf,
written in and for bmake (NetBSD make).
...                   
#```
pkgsrc4unix prototype (pre-alpha stage of development!!!)

System: RHEL-6, .rpm, rpm(8), yum(8)

```
# rpm -ql nb-mk-configure | head -18
/opt/pkgsrc4unix/bin
/opt/pkgsrc4unix/bin/mkc_check_common.sh
/opt/pkgsrc4unix/bin/mkc_check_compiler
/opt/pkgsrc4unix/bin/mkc_check_custom
/opt/pkgsrc4unix/bin/mkc_check_decl
/opt/pkgsrc4unix/bin/mkc_check_funclib
/opt/pkgsrc4unix/bin/mkc_check_header
/opt/pkgsrc4unix/bin/mkc_check_prog
/opt/pkgsrc4unix/bin/mkc_check_sizeof
/opt/pkgsrc4unix/bin/mkc_check_version
/opt/pkgsrc4unix/bin/mkc_install
/opt/pkgsrc4unix/bin/mkc_test_helper
/opt/pkgsrc4unix/bin/mkc_which
/opt/pkgsrc4unix/bin/mkcmake
/opt/pkgsrc4unix/man
/opt/pkgsrc4unix/man/man1
/opt/pkgsrc4unix/man/man1/mkc_check_custom.1
#
```
# export PATH=/opt/pkgsrc4unix/bin:$PATH
#