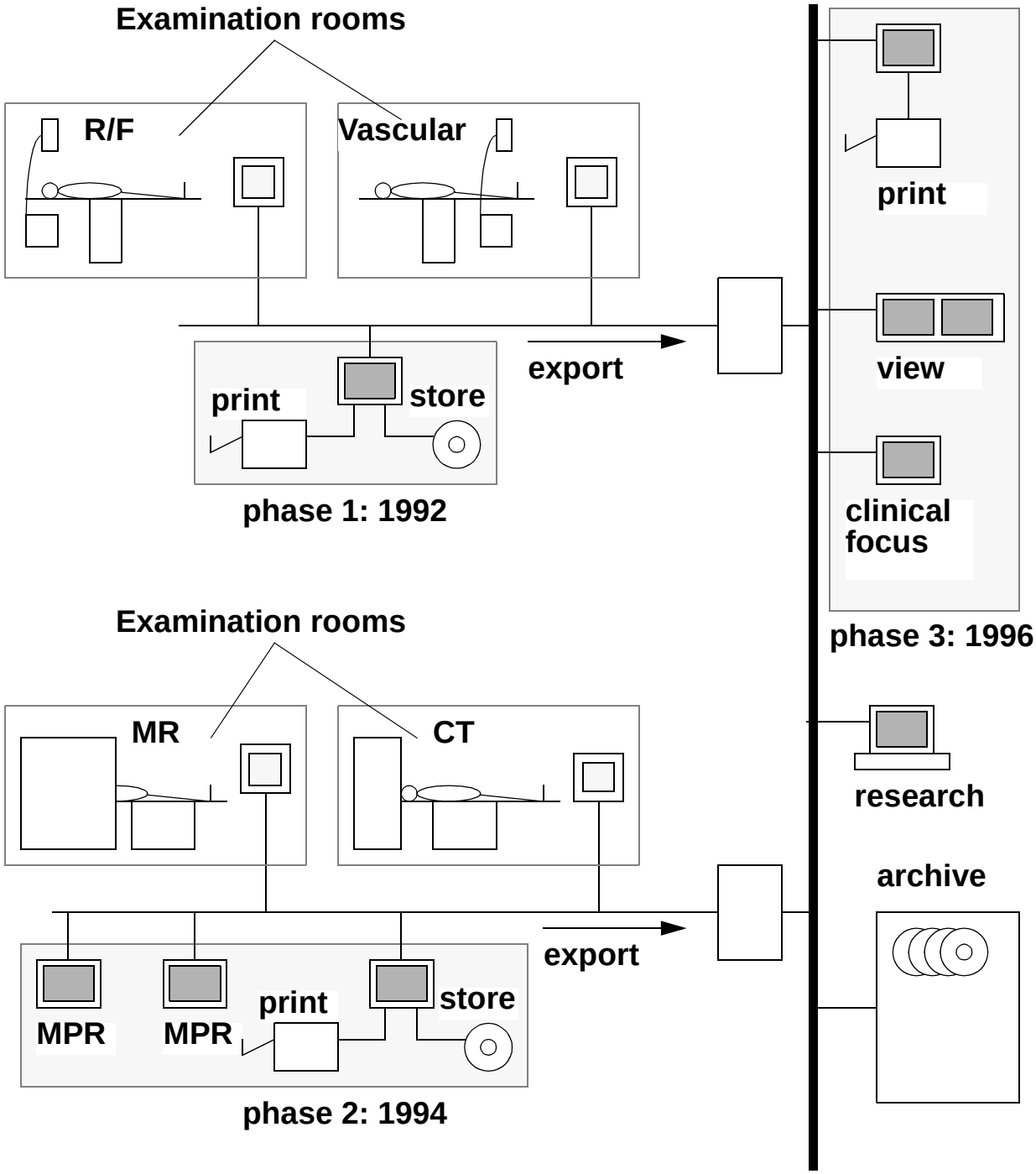
EasyVision family of products



Product types:

- Modality productivity enhancers:
  - + Easyvision R/F
  - + Easyvision RAD
  - + Easyvision CT/MR

street price ca 50 k$, high added clinical value; sales directly related to modality sales

- Clinical Focus:
  - + Neurovision
  - + Image Guided Surgery

street price ca 100 k$, very high added clinical value; sales limited to specialist areas

- “PACS” workstations
  - + Teleradiology Workstation
  - + Critical Care Workstation
  - + Multi modality review station

street price ca 25 k$, low added value, low margin; sales potentially very high

Extrapolation CDS SW.

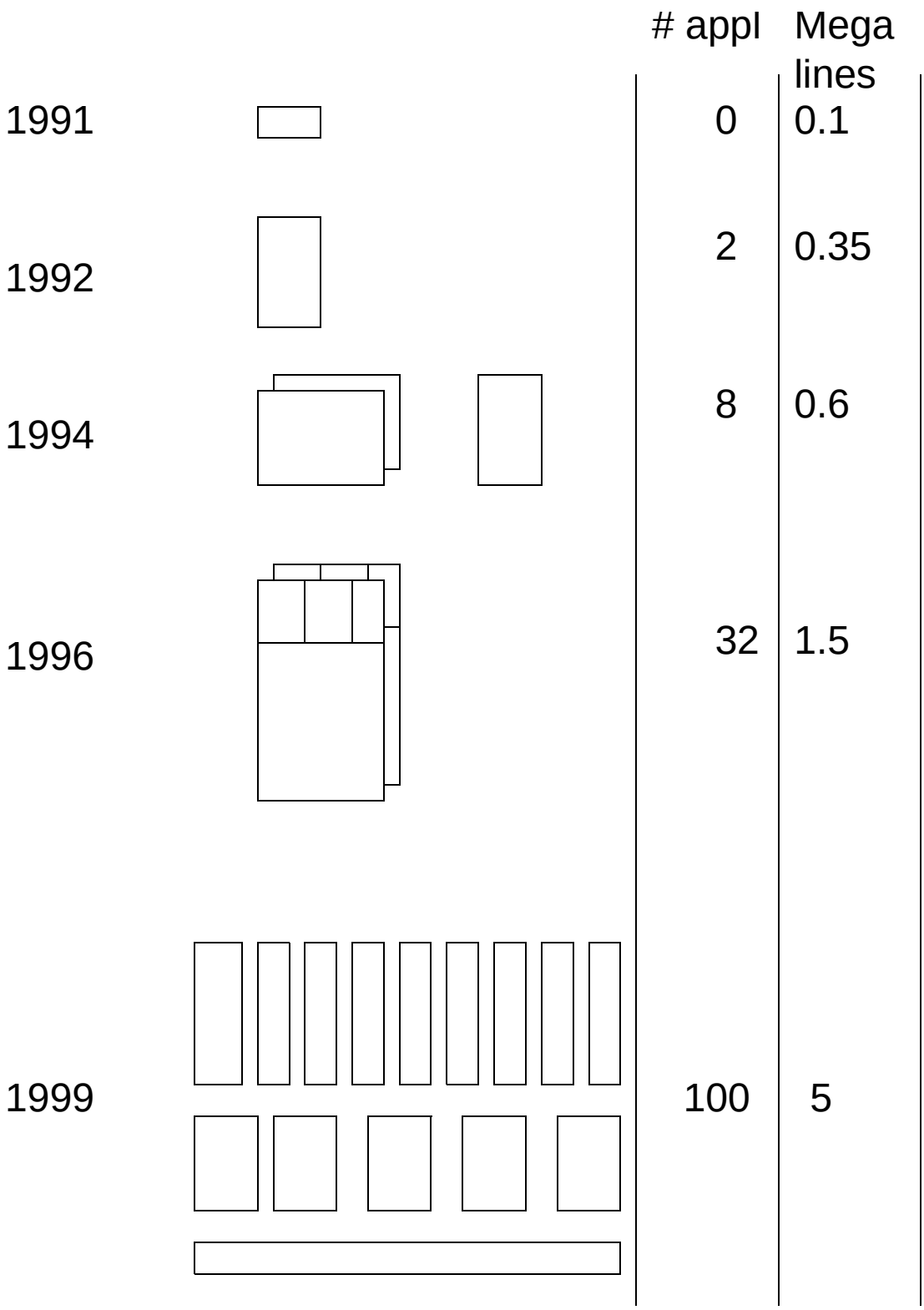


Table 1: Efficiency through re-use

	1992	1993	1994	1995	1996
number applications					
applications	1	4	8	16	32
inputs, a.o. modalities	1	5	10	15	
people					
infrastruc-ture			20+15	21+16	22+16
application			27	35	41
total		52	62	72	79
efficiency					
people per application		13	8	5	3

To OO or not to OO

Characteristics of the Easyvision application are:

- Large variety in input images
  - + 256^2, 480^2, 512^2, 1024^2, non square, etc.
  - + 8, 10, 12 bits
  - + CT, MR, X-ray Image Intensifier
- Large variety in application requirements
- Large variety in use

Easyvision is impossible without OO

Method

Easyvision development method:

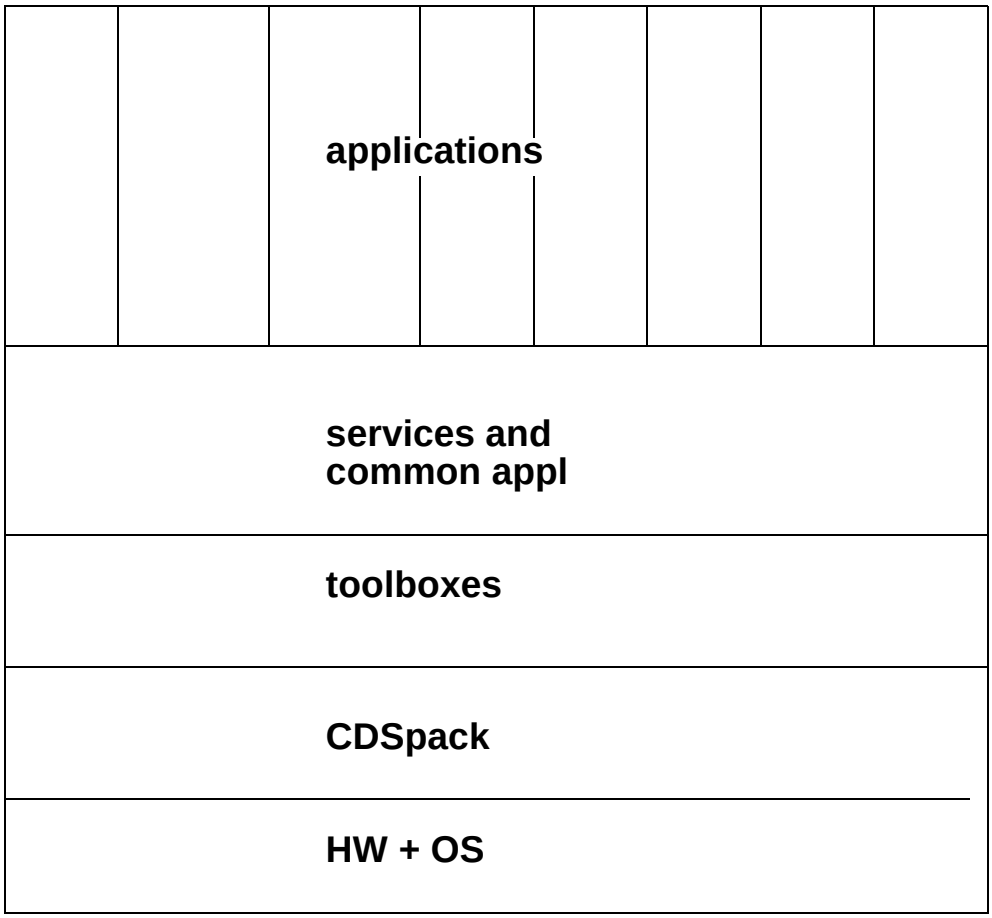
- prototype
- evaluate
- engineering

No formal analysis/design/ documentation method!

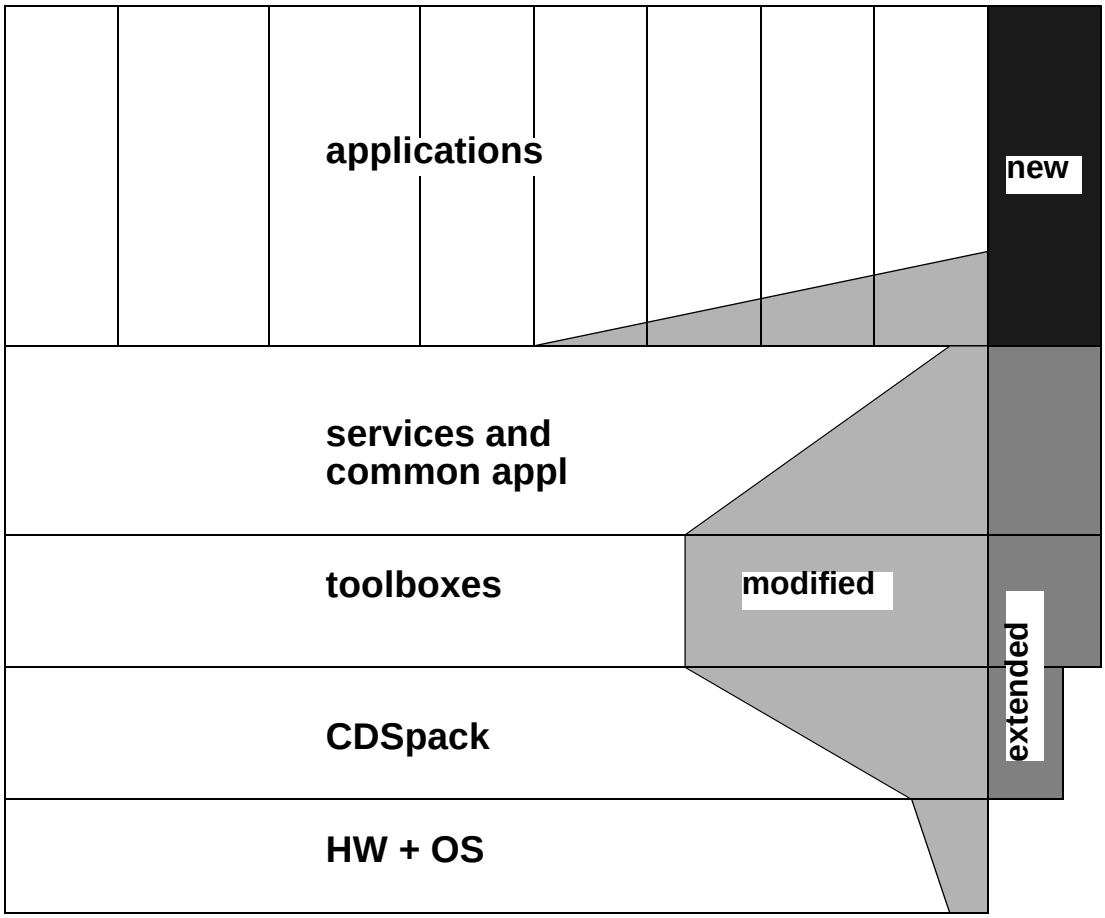
Formal methods:

- work for small projects only
- playground for academics :-)

Simplified layers



Adding an application



Interfaces

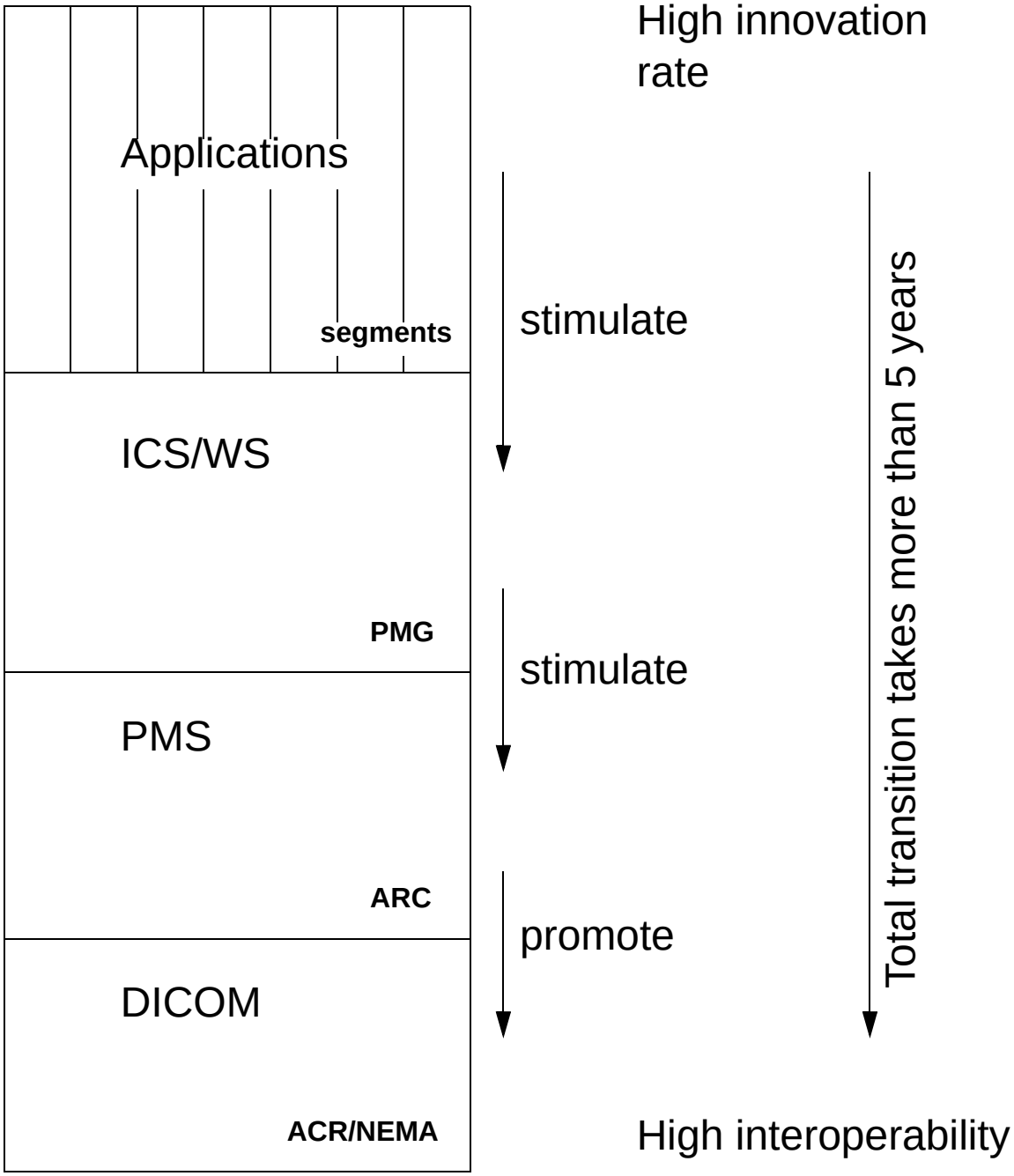
Internal:

- Information model
- Communication mechanism(s):
  - + Database storage and notification support
  - + Connection
  - + In and out streaming support
- API’s to common applications, toolboxes and CDSpack:
  - + Objective-C classes and methods
  - + properties

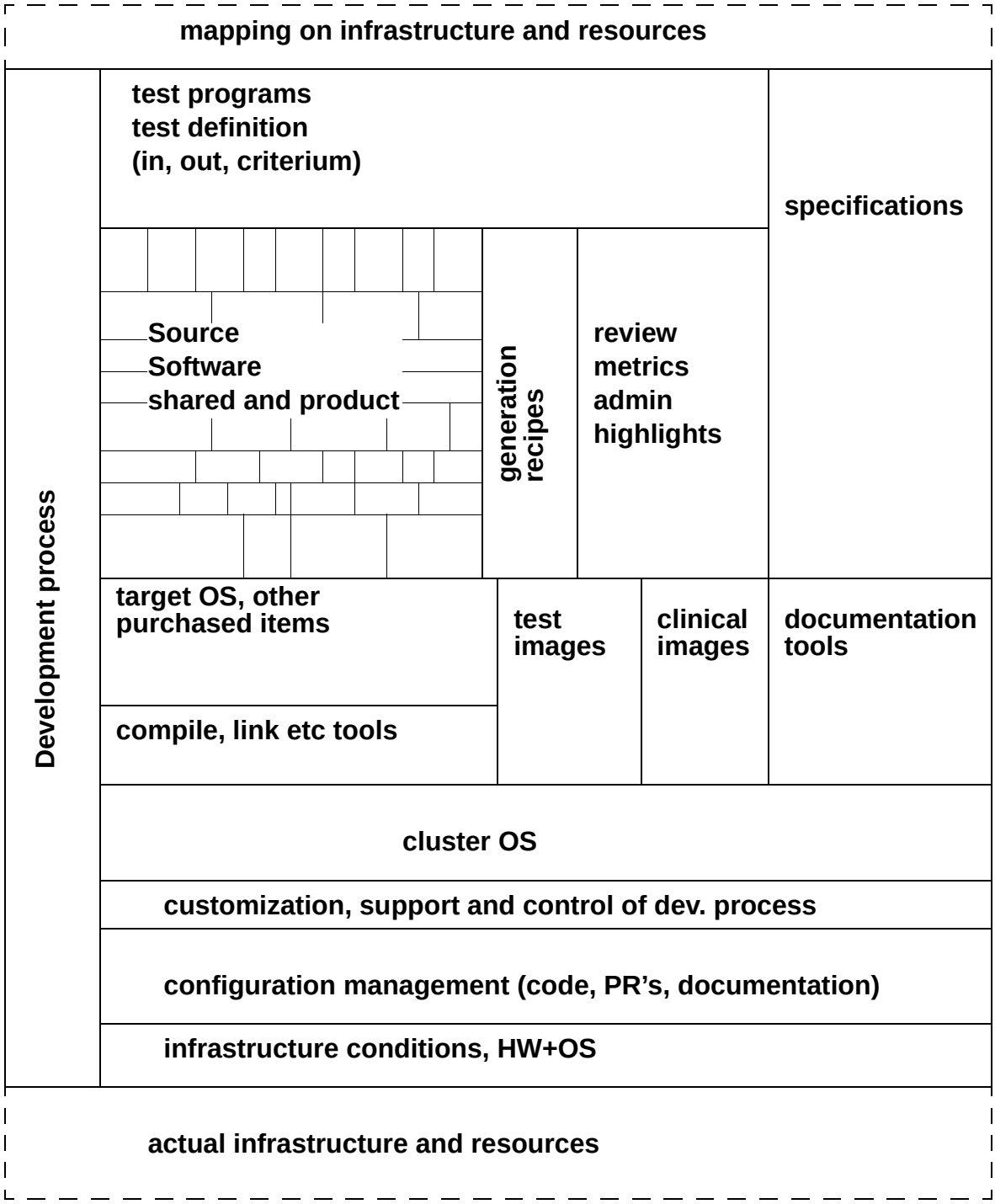
External:

- Dicom + PMS + ICS information model
- DICOM services and mechanisms
- PMS and ICS services

Information model



The platform as deliverable



The Buy myth.

The nineties MBA course teaches:

Thou shall buy....

The poor heathen suffers from NIH

(Not Invented Here) syndrome.

Reality is somewhat more complicated!

Buy, potential components:

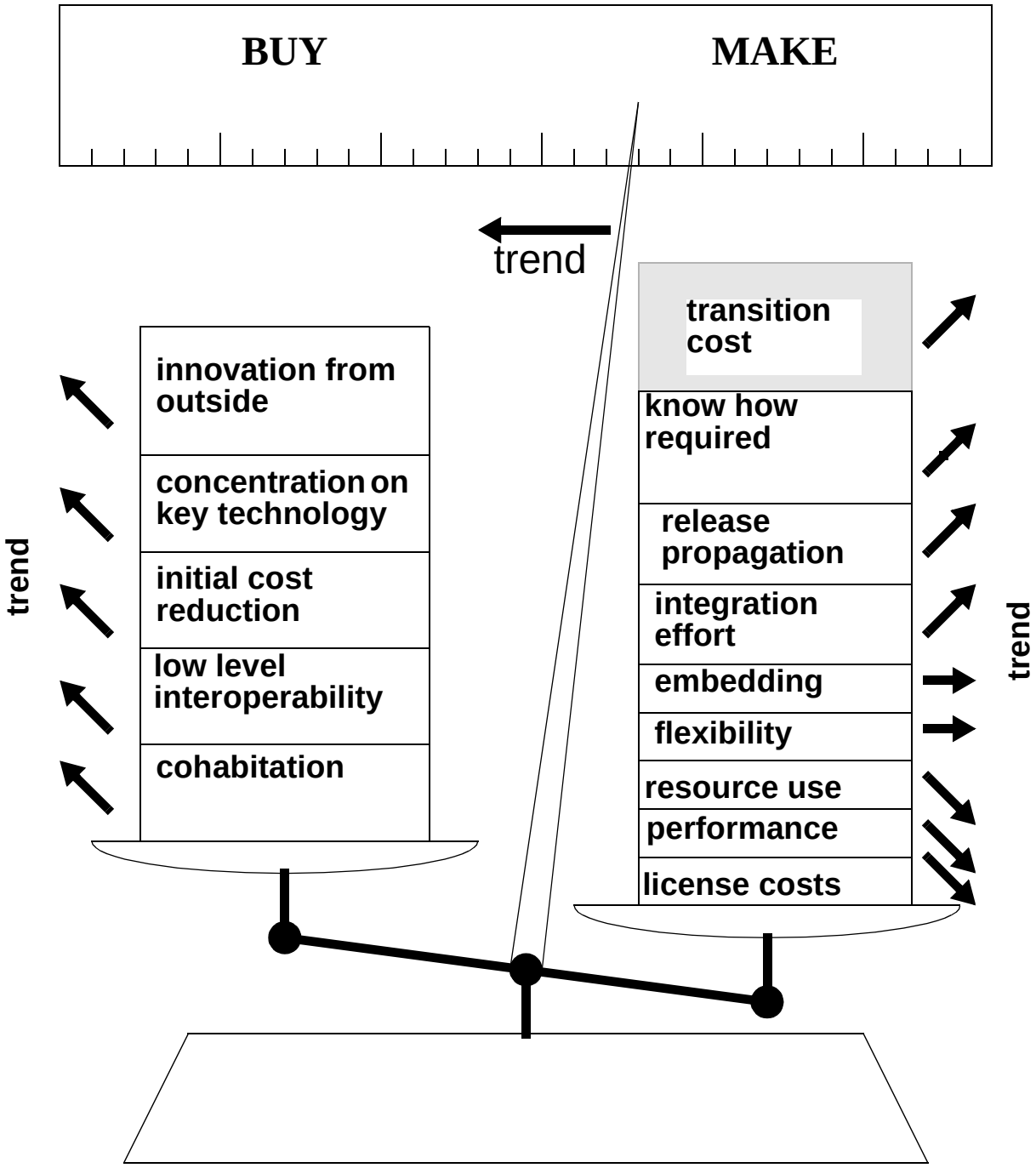
- Operating system
- Communication
- Data base engine
- User interface and related utilities
- Graphics and related utilities
- Image processing
- 3D rendering
- Foundation classes
- Installation
- Licensing, SW keys
- Security, a.o. encryption
- Multi media, virtual reality peripheral support
- etc.

Embedding

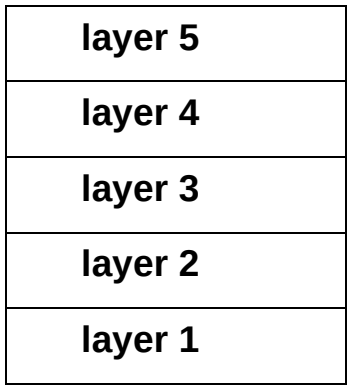
- Installation
- Configuration
- Customization
- Start up, shutdown
- Specifications:
  - + functional
  - + system design
  - + sw design
- Interface to application SW:
  - + add semantics level
  - + use of appropriate low level mechanisms
  - + match to high level mechanisms:
    - - notification, scheduling
    - - job requests, subscriptions

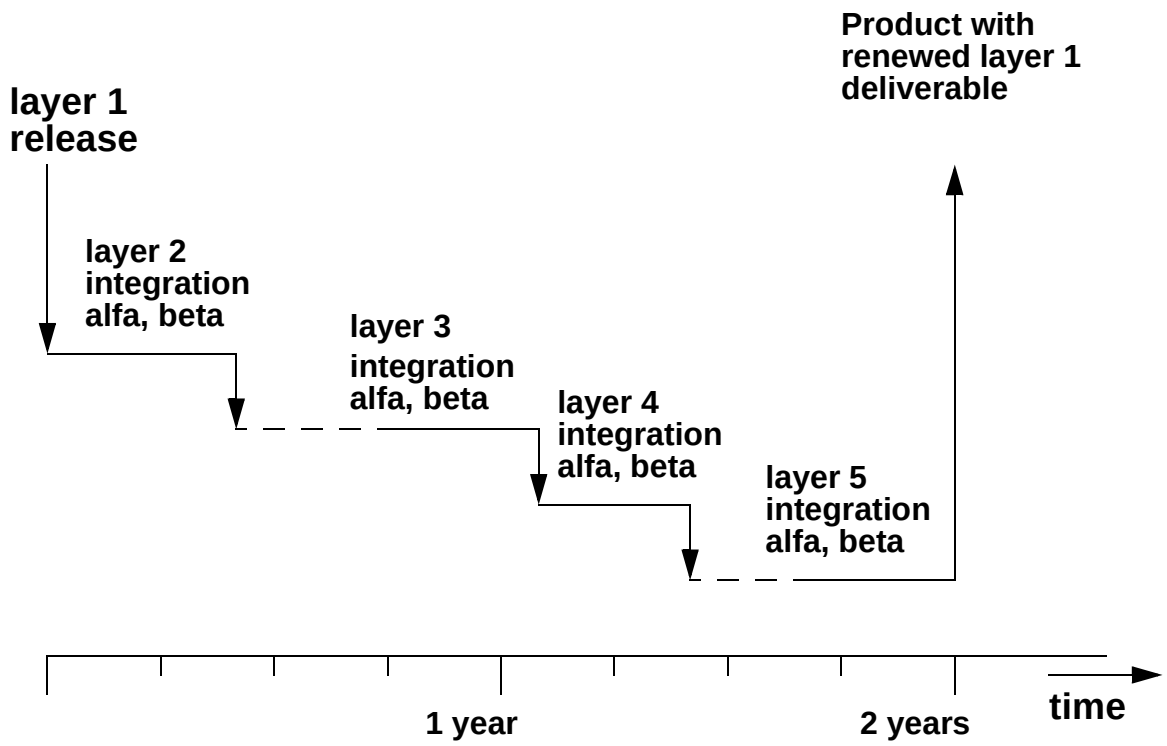
Embedding (continued)

- Exception handling:
  - + System monitor
  - + Error propagation
  - + Logging
- Resource allocation and monitoring provisions
  - + CPU
  - + Memory
  - + Disk
- Resource tuning, see above
- Safety design
- Security design


BUY
MAKE
trend
innovation from outside
concentration on key technology
initial cost reduction
low level interoperability
cohabitation
trend
transition cost
know how required
release propagation
integration effort
embedding
flexibility
resource use
performance
license costs
trend

Release propagation delay





The Buy myth converted in common sense:

The right questions to ask are:

- When to buy?

- How can the design enable buy?

- Which process is needed for buy?

The Re-use myth put into perspective:

The right questions to ask are:

- When to re-use (cost vs. benefit)?

- How can the design enable re-use?

- Which process is needed for re-use?

Pro Re-use

- development cost sharing
- verification cost sharing
- same look and feel
- application developers focus on application
- increased quality, due to repeated use

Contra Re-use

- cost of generalization
- overhead cost
- increase of total complexity
- coupling of lifecycles, products, schedules
- vulnerability (Biological evolution is based on diversity...)

See make vs. buy

Re-use is means not goal.

A look into the future

From box to function:

- customer wants any function on any location/time, not limited by “random” product or box boundaries

In parallel with:

- large number of clinical applications
- integration of health care function
- break down in manageable projects / teams, lifecycle independency
- finite number of skilled development personnel

(R)evolution in 25 years

Table 2:

	1980	1995	2005
integration level	generator stand	depart-ment	health care
time to market	2-5 year	1-2 year	0.5 year
code size complete product	10^4-10^5	10^6	10^7-10^8
memory size	96 kB	96 MB	? GB
CPU power	0.1 MIPS	100 MIPS	? GIPS
dev group size	10-50	50-200	?
of which ASW	2-10	20-60	?

Skills

- increased integration
- increased complexity
- increased abstraction
- increased focus on application
- increased (time) pressure

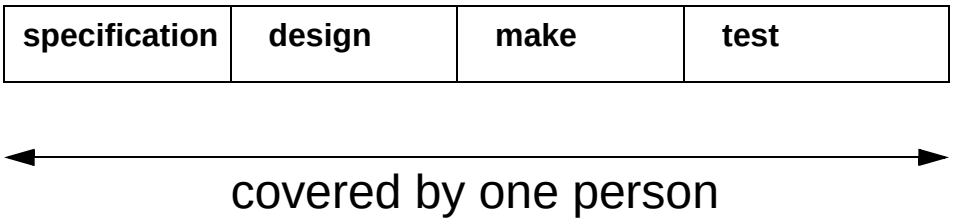
Increased skills required

Profile of 1999 developer

- application oriented
- development process aware
- multi disciplinary
- fluent in software engineering lingo

These people are rare!

Our industry will become skill limited, instead of initial cost limited


specification
design
make
test
covered by one person

Technological changes, opportunities

- Corba, SOM, OLE, ...
- Java, ...
- Windows NT, Windows 95, OS 2
- Taligent, Spring, ...
- SW only products
- Multi media (HW+SW)

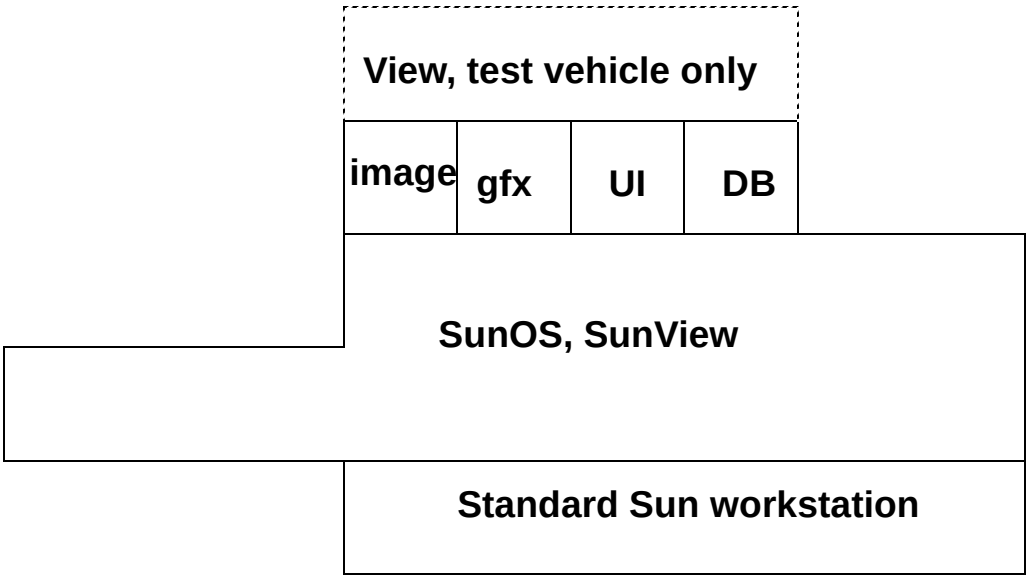
EasyVision in 2000

- More than 100 independent applications
- Interoperating fluently with other EV applications
- Interoperating fluently with other vendors
- Interoperating fluently with other health care applications (Information systems, etc.)
- SW only
- Running on at least UNIX and NT platforms
- Distributed development process
- Consolidation and cross fertilization process
- Platform for innovative applications in image handling, analysis, clinical focus.

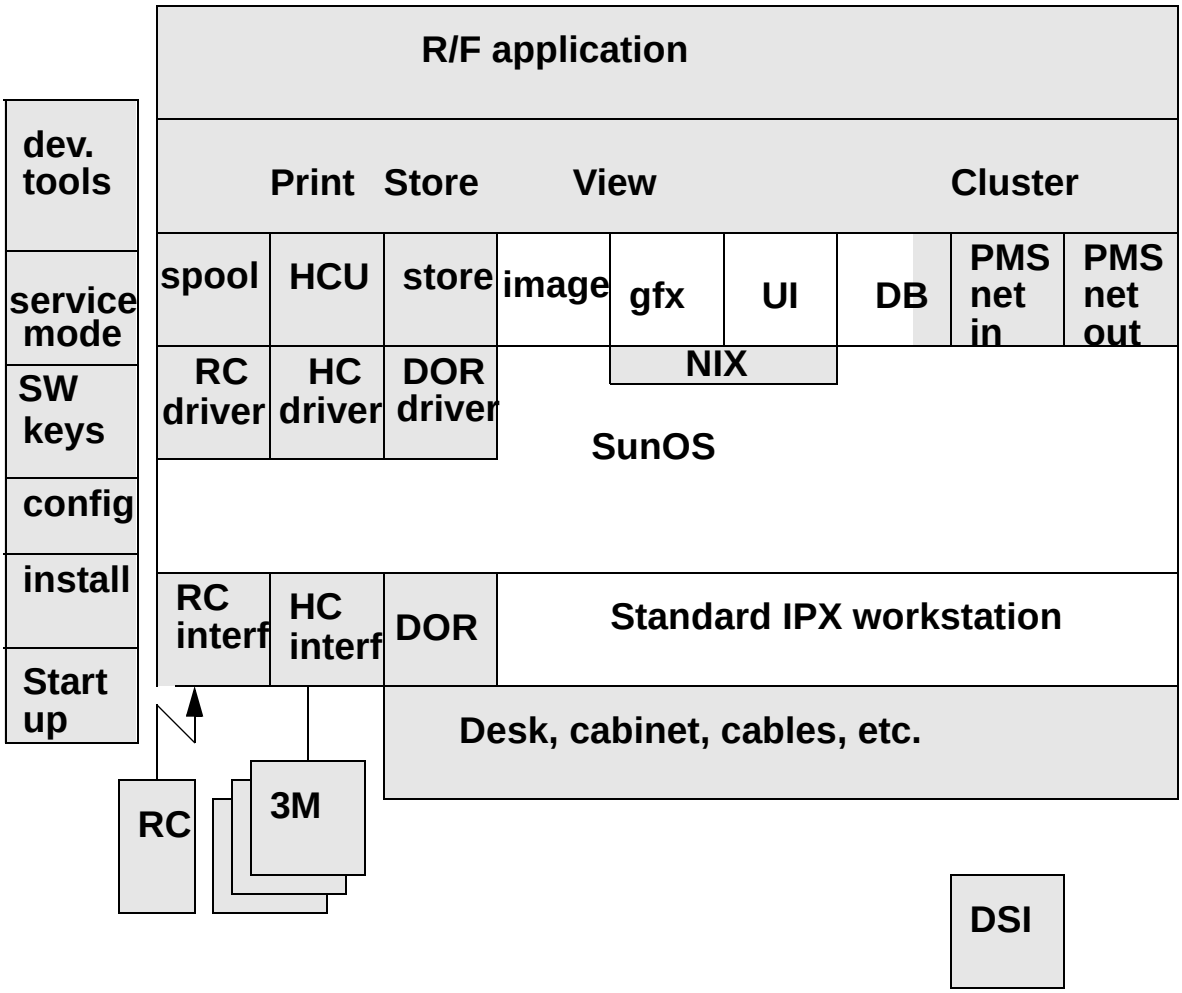
Re-use levels

- Concepts
- Development Process
- Interoperability architecture
- Functional specifications
- User interface
- Algorithms
- Design
- Verification (test suite, spec)
- Skills
- Copy implementation, code
- Implementation, code
- Application modules

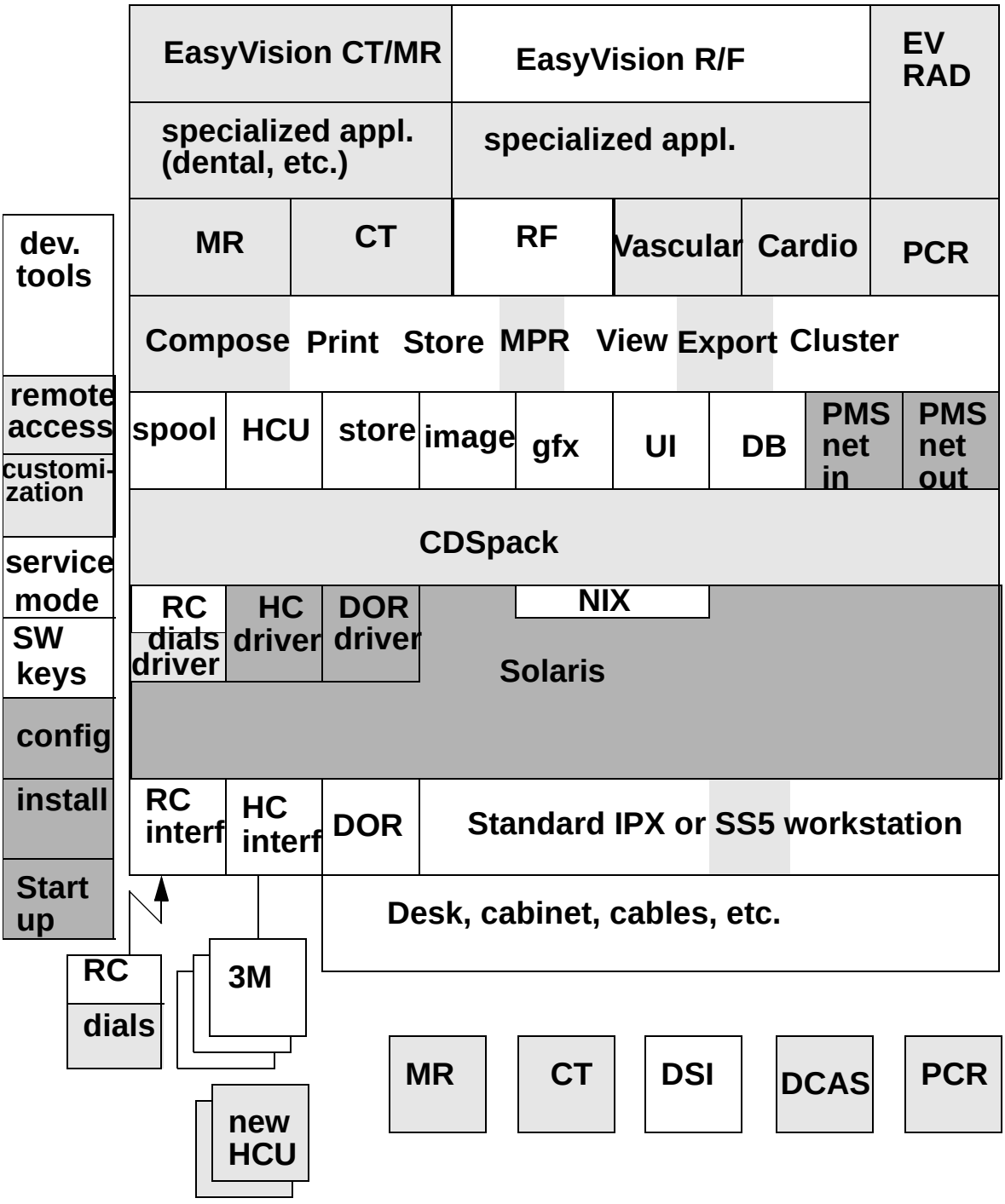
september 1991



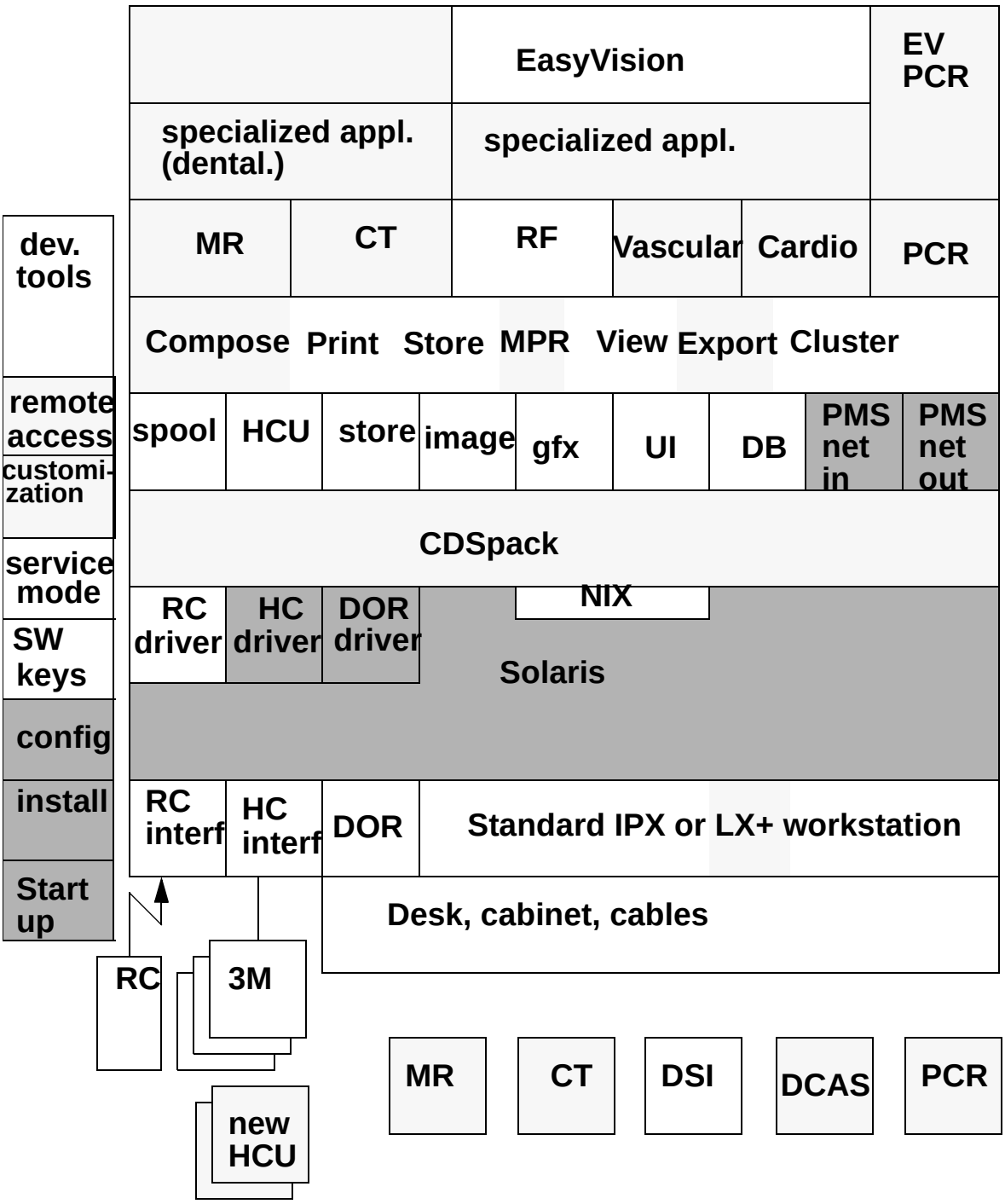
september 1992



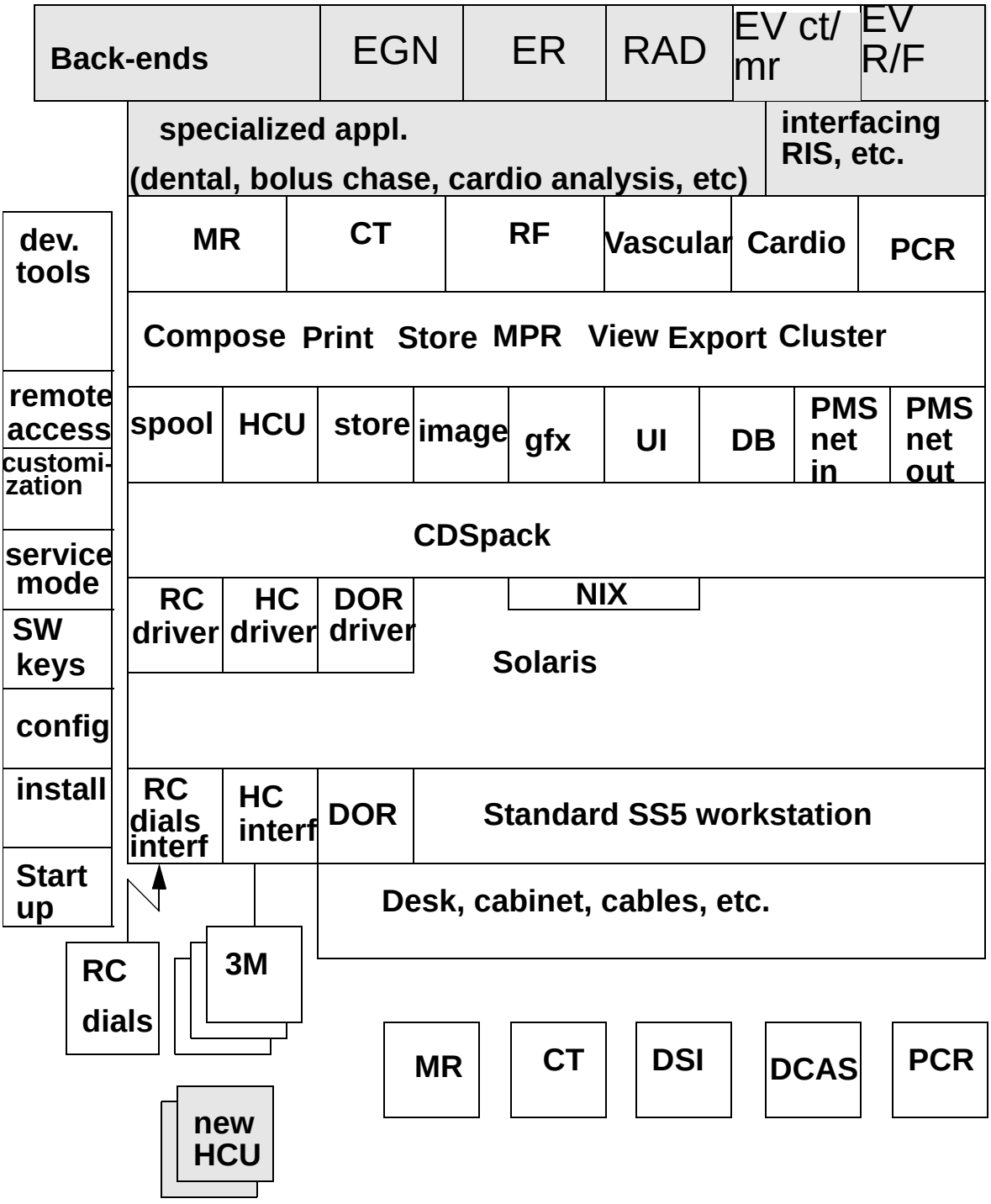
june 1994


EasyVision CT/MR
EasyVision R/F
EV RAD
specialized appl. (dental, etc.)
specialized appl.
dev. tools
MR
CT
RF
Vascular
Cardio
PCR
Compose
Print
Store
MPR
View
Export
Cluster
remote access
customi-zation
service mode
SW keys
config
install
Start up
spool
HCU
store
image
gfx
UI
DB
PMS net in
PMS net out
CDSpack
RC dials driver
HC driver
DOR driver
NIX
Solaris
RC interf
HC interf
DOR
Standard IPX or SS5 workstation
Desk, cabinet, cables, etc.
RC dials
3M
new HCU
MR
CT
DSI
DCAS
PCR

june 1994


EasyVision
EV PCR
specialized appl. (dental.)
specialized appl.
dev. tools
MR
CT
RF
Vascular
Cardio
PCR
Compose
Print
Store
MPR
View
Export
Cluster
remote access
spool
HCU
store
image
gfx
UI
DB
PMS net in
PMS net out
customi-zation
CDSpack
service mode
NIX
RC driver
HC driver
DOR driver
SW keys
Solaris
config
install
RC interf
HC interf
DOR
Standard IPX or LX+ workstation
Start up
Desk, cabinet, cables
RC
3M
new HCU
MR
CT
DSI
DCAS
PCR

1996


Back-ends
EGN
ER
RAD
EV ct/ mr
EV R/F
specialized appl.
(dental, bolus chase, cardio analysis, etc)
interfacing RIS, etc.
dev. tools
MR
CT
RF
Vascular
Cardio
PCR
Compose
Print
Store
MPR
View
Export
Cluster
remote access
customi-zation
spool
HCU
store
image
gfx
UI
DB
PMS net in
PMS net out
CDSpack
service mode
SW keys
RC driver
HC driver
DOR driver
NIX
Solaris
config
install
Start up
RC dials interf
HC interf
DOR
Standard SS5 workstation
Desk, cabinet, cables, etc.
RC dials
3M
new HCU
MR
CT
DSI
DCAS
PCR